Towards Workload-aware Cloud Efficiency: A Large-scale Empirical Study of Cloud Workload Characteristics

Anjaly Parayil
aparayil@microsoft.com
Microsoft
Bangalore, India

Jue Zhang
Xiaoting Qin
jue.zhang@microsoft.com
xiaotingqin@microsoft.com
Microsoft
Beijing, China

Íñigo Goiri
inigog@microsoft.com
Microsoft
Redmond, USA

Lexiang Huang[**]
Timothy Zhu
lexiangh@psu.edu
tuz68@psu.edu
The Pennsylvania State University
University Park, USA

Chetan Bansal
chetanb@microsoft.com
Microsoft
Redmond, USA

Abstract

Cloud providers introduce features and optimizations to improve efficiency and reliability, such as Spot VMs, Harvest VMs, oversubscription, and auto-scaling. To use these effectively, it's important to understand workload characteristics. However, workload characterization can be complex and difficult to scale manually due to multiple signals involved. In this study, we conduct the first large-scale empirical study of first-party workloads at Microsoft to understand their characteristics. Through this empirical study, we aim to answer the following questions: (1) What are the critical workload characteristics that impact efficiency and reliability on cloud platforms? (2) How do these characteristics vary across different workloads? (3) How can cloud platforms leverage these insights to efficiently characterize all workloads at scale? This study provides a deeper understanding of workload characteristics and their impact on cloud performance, which can aid in optimizing cloud services and identifies potential areas for future research.

CCS Concepts

• **General and reference** → **Empirical studies**.

Keywords

Cloud platform, Workload characteristics, Empirical study, Workload awareness, Efficiency, Performance and Reliability

ACM Reference Format:
Anjaly Parayil, Jue Zhang, Xiaoting Qin, Íñigo Goiri, Lexiang Huang, Timothy Zhu, and Chetan Bansal. 2025. Towards Workload-aware Cloud Efficiency: A Large-scale Empirical Study of Cloud Workload Characteristics. In *Proceedings of the 16th ACM/SPEC International Conference on Performance Engineering (ICPE '25), May 5–9, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3676151.3722008

[*]Lexiang Huang was an intern at Microsoft.

1 Introduction

Motivation. Cloud platforms are a key component of the global IT infrastructure, with major players like Amazon Web Services [3], Microsoft Azure [39], and Google Cloud [25] providing a wide range of cloud services to individuals and organizations around the world. With the increase in cloud adoption, these providers are constantly offering new features and services, as well as introducing various optimizations to increase the efficiency and reliability of the platforms. These include new VM types (Spot VMs [4, 9, 16], Harvest VMs [7], Burstable VMs [53]), dedicated interfaces (*e.g.*, auto-scaling [38]), and optimizations that are internal to the cloud (*e.g.*, oversubscription [31], pre-provisioning [58]). To make the most of new features and cloud optimizations, understanding the characteristics of cloud workloads is crucial. This information can help cloud platforms determine which workloads are best suited for specific features or optimizations.

Challenges. Previous work in resource management, workload profiling, and Machine Learning approaches has limitations in providing complete information on workload requirements [2, 11, 13, 18, 21, 28, 29, 35, 60]. These approaches rely on VM or container level telemetry, which may not provide insights on crucial characteristics such as delay sensitivity [27]. Asking the workload owner for this information is not scalable due to the large number of cloud workloads. It is also important to understand which workloads benefit from each cloud optimization, as well as whether the workload users care about certain performance factors. For example, overclocking the CPU running a memory-bound workload will not result in much performance improvement [27] and some users prioritize cost efficiency even if it takes longer to complete the job.

Our work. We first identify the characteristics (*e.g.*, preemptibility or delay tolerance) required for various common cloud platform optimizations. Given that these characteristics are often not readily available, we conducted a survey of 188 first-party (*i.e.*, internal) workloads comprising over 100,000 VMs in Microsoft, a major public cloud provider. Through the empirical study, we categorized the workloads into six distinct classes based on their functionality and studied their performance, reliability, geographical, and scalability requirements. We then mapped each workload class to different cloud optimizations based on their suitability and found that cloud providers can utilize the unique profiles of workload class to enhance their efficiency and reliability. Our findings can help reduce costs while achieving desired performance for cloud users. This methodology and insights generalize to other external workloads as well, paving the way for an automatic characterization of cloud workloads at scale.

Contributions. In this paper, we make the following contributions:

- We present the first large-scale empirical study and characterization of real-world cloud workloads, with a focus on the relationship between workload characteristics and cloud optimizations.
- We derive insights on the workload characteristics and estimate opportunities for various cloud optimizations.
- We open-source the survey form used to collect information from the workload owners and aim to publish the raw data (pending the necessary privacy and security approvals).

The rest of the paper is organized as follows: Section 2 overviews some common cloud platform optimizations. Section 3 discusses the methodology of the survey. Section 4 analyzes workload characteristics and presents insights. Section 5 explores the implications derived from the study. Section 6 provides an overview of the related research. Section 7 presents our conclusions.

2 Cloud platform optimizations

We define a workload as a collection of applications, services, and data that support a specific process. A clear understanding of the unique requirements of each workload is crucial to optimize cloud platforms. Through an extensive literature survey and discussions with domain experts at Microsoft, we identify ten popular cloud optimizations and their necessary workload characteristics. Table 1 provides a summary of the workload characteristics required for each optimization.

Auto-scaling: Cloud providers offer an auto-scaling feature to dynamically adjust the number of VMs based on load, allowing users to not always provision VMs for the peak load [38]. Workloads designed to support scale in and scale out are required for this feature. Providers offer auto-scaling as a separate service ([6, 40], where users define an auto-scaling policy for their VMs. Workloads that are stateless or delay tolerant and have relaxed deployment time requirements are suitable for auto-scaling.

Spot VMs: Cloud providers offer discounted Spot VMs with relaxed SLOs to monetize unallocated capacity [4, 9, 16]. These low-priority VMs may be evicted if their resources are needed by on-demand VMs. Providers offer Spot VMs through deployment flags and new VM types, and some use dynamic pricing to decide which to evict first. Preemptible workloads are suitable for this optimization [5, 8].

Harvest VMs: Harvest VMs dynamically grow and shrink a VM's size to utilize spare CPU, memory, and storage in the server, building on top of Spot VMs [7, 23, 44, 54]. Providers offer harvesting as a fixed VM type or a deployment flag specifying the amount of resources to harvest. These VMs are ideal for workloads that are both preemptible and delay tolerant.

Overclocking: Cloud providers can improve workload performance by increasing the CPU frequency of VMs, but this approach should consider trade-offs with reliability and power budget [27]. Overclocking is suitable for workloads that are both delay-sensitive and non-preemptible with high CPU utilization periods. However, not all applications benefit from faster CPUs. Cloud providers offer dedicated VM types that allow higher frequencies and provide interfaces for processor power states, such as EC2 frequency control ([27].

Underclocking: On the other hand, underclocking reduces the CPU frequency and decreases the power requirements at the cost of performance reduction. A workload is suitable for underclocking if it is either delay-tolerant, non-user-facing, or highly preemptible. Cloud providers offer dedicated VM types that allow for accounting of CO_2 and energy savings.

VM pre-provisioning: Providers can reduce VM creation time by pre-provisioning VMs and instantiating them as needed [59]. This approach is suitable for workloads with tight delay requirements and works well with auto-scaling. However, pre-provisioned VMs can result in wasted resources and increased costs, as they are allocated a fixed amount of resources regardless of usage. Disabling pre-provisioning when not needed can save costs with little impact on performance.

Region-agnostic placement: To reduce cost and CO_2 emissions, workloads can run on cheaper and greener regions. This is suitable for workloads without latency or data-locality requirements that constrain them to a specific region. While there are proposals to do this semi-automatically [1, 49], there are currently no commercial solutions and workload owners need to manually specify the region.

VM oversubscription: Platforms can increase server utilization by oversubscribing servers with more VMs [56]. This optimization relies on statistical multiplexing of different workloads, but the platform throttles the least critical VMs if all VMs spike in load simultaneously. Oversubscription is suitable for workloads with varying usage over time, particularly those that are either delay-tolerant or non-user-facing with a 95^{th} percentile CPU utilization of less than 65% [18]. Cloud platforms typically determine which VMs can be oversubscribed and by how much [18].

VM rightsizing: The cloud platform can identify underutilized VMs and recommend a smaller VM type similar to VM oversubscription [10]. This proactive adjustment based on workload characteristics is particularly useful for workloads that can scale down and have lower utilization (*e.g.*, <50%), allowing for a transition to a smaller VM size (typically half size).

Multi-availability datacenters (MA DCs): Cloud providers can reduce costs by removing redundant infrastructure for power delivery and cooling. During infrastructure failures or maintenance events, the cloud platform may need to throttle or selectively turn off servers [60]. Multi-availability DCs leverage workloads that require low availability, and the cloud platform currently infers which VMs are less critical and throttles them down or evicts them if necessary.

Cloud optimization	Required workload characteristics
Auto-scaling	(Stateless $\vee$ Delay tolerant) $\wedge$ Relaxed deployment time
Spot VMs	Preemptible
Harvest VMs	Preemptible $\wedge$ Delay tolerant
Overclock	Delay sensitive $\wedge$ Non preemptible $\wedge$ P95 Max % CPU > 40
Underclock	Delay tolerant $\vee$ Non user-facing $\vee$ Preemptible
Pre-provision	Stringent deployment time
Region-agnostic	Geographical requirements
VM oversubscription	(Delay tolerant $\vee$ Non user-facing) $\wedge$ (P95 Max % CPU < 65)
VM rightsizing	P95 Avg % CPU < 25 $\wedge$ P95 Avg % mem < 25 $\wedge$ P95 Max % CPU < 50 $\wedge$ P95 Max % mem < 50
Multi-availability DC	Low availability

Table 1: Cloud platform optimizations and workload characteristics ($\vee$: or, $\wedge$: and, Px Max: x^{th} percentile of maximum utilization, Px Avg: x^{th} percentile of average utilization).

2.1 Relevant workload characteristics

There are many ways to characterize workloads, but only a subset relates to cloud optimizations. We identify the workload characteristics needed by cloud optimizations and categorize them into: (1) performance and reliability, (2) geographical, and (3) scalability.

Performance. Workloads have different performance requirements.
User-facing: Whether the workload is serving users and shows a periodic pattern.
Delay tolerance: Whether the workload tolerates delays (including deadlines). This can go from tail-latency SLOs ($< 200ms$) to batch job deadlines (finish a job before noon).

Reliability. Related to the workload availability requirements.

Availability: How much the workload allows a VM to not be available. This is usually measured in number of nines.

Preemptibility: How well the workload handles losing VMs. Most cloud workloads are built with fault tolerance in mind. However, there are different degrees and not all workloads support losing 50% of their VMs.

Geographical. Related to the workload requirement to run in a specific location or region.

Proximity and locality: Workloads may have location constraints related to latency (proximity to users) or data locality (bandwidth requirements to access data in other locations), indicating whether VMs running the workload can change locations at runtime.

Scalability. Scalability refers to a workload's ability to handle increased demand while maintaining consistent Quality-of-Service. A scalable workload can efficiently handle more resources to scale in/out or scale up/down [38]. Requirements for deploying additional resources are also crucial for better scaling.

Scale up/down: Whether the workload can adjust to changing the size of the VM. For example, switching to a VM with more or fewer cores or memory. This also applies to changes in CPU frequency.

Scale out/in: Whether the workload can adjust to the addition or removal of VMs.

Deployment time: The time that a workload requires to setup after getting a new VM. This relates to the scale out component.

3 Survey methodology

To understand the characteristics of workloads running in the cloud, we conduct a large-scale survey of first-party workloads at Microsoft. We design this survey based on the characteristics that existing cloud platform optimizations require to operate. This is the largest empirical study of the characteristics of workloads running in the cloud to date.

Scope. Microsoft has first-party workloads that include internal services for research and development, infrastructure management, and first-party services offered to third-party customers like communication, gaming, and data management. These workloads are tracked in an internal directory categorized into 14 divisions based on the organizational hierarchy. Division 1 is the largest division in terms of VMs, cores, and deployed regions, mainly encompassing internal workloads for research and development, as well as tools and platforms for cloud services offered by Microsoft. Division 2 creates innovative products and services for people and organizations, including web search, collaboration and productivity suites, and real-time communication services. This study targets Division 2 due to easier access to workload owners and representative features. Relevant metrics from all 14 divisions can be found in Table 10 of the Appendix for the week of June 20th-27th 2022 for all workloads with non-zero core usage.
Overall, we consider 1034 workloads among which the largest one employs over 100k VMs. These workloads are deployed to 49 regions across the world and used by hundreds of millions of users.

Process. We conducted three in-person interviews to refine survey questions. Then, we shared an online form with 27 workload owners selected via weighted random sampling based on their core usage. After refining the survey based on the first 30 workloads, we sent the online form to the remaining 1004 services. The final response rate was 19%, covering 188 workloads (24.3% of the total cores).

Questions. We formulate 21 questions to collect information on workload characteristics and requirements based on the optimizations described in Section 2. We begin by asking for a high-level description, including the main functionality, whether the workload is composed of other workloads, and whether it is user-facing. Then, we move into specific questions in the three categories identified in Section 2.1. A summary of the questions in each category is provided, and interested readers may refer to **??** for the complete list of survey questions.

Performance: We ask if the workload is tolerant to delays and if they have some performance expectation (*e.g.*, latency). We use the expected latency values provided by the workload owners to verify their response to delay tolerance.

Reliability: We ask for availability and fault tolerance requirements.

Geographical: We ask whether the workload can be migrated to other regions. We also ask for the factors restricting cross geo-migrations if their workloads cannot be migrated to other regions.

Scalability: We ask for the potential to run in a different VM configuration (*e.g.*, running in more/less VMs). We also ask if the workload is stateless as stateless workloads are potential candidates for easy scaling due to their input-independent control flows [61].

Free-form text responses. We use free-form text response questions in the survey and use the open coding approach to categorize the responses. The data is randomized and split into three sets: the taxonomy set (30%), the validation set (20%), and the label set (remaining 50%). Two annotators, who are authors of this work with extensive background in systems design and modeling, label the sets independently and then discuss to reach a consensus. Finally, they annotate the label set and compute the inter-annotator agreement score using Cohen's kappa [17] to create a systematically labeled dataset for further analysis.

4 Survey results

In this section, we present the results from the survey. We start by describing the workloads at a high level and move into the detailed results for each of the workload characteristics categories. Unless stated otherwise, when talking about percentages, we are referring to CPU core percentages.

4.1 Workload classes

To provide high-level intuitions about the results, we categorize the workloads into major classes. We correlate the descriptions provided by the workload owners in the survey with their internal directory information and use open-coding to group these descriptions into six classes. The annotators have near-perfect agreement (*i.e.*, Cohen's kappa score of 0.965). Based on these classes, we study their popularity and core usage. "Web Apps" and "Big Data" constitute 50% and 18% of all workloads, respectively. "RTC" and "ML Inference" workloads have lower frequency but higher average core usage due to their compute-intensive nature. We include an "ML Inference" category in the taxonomy, as opposed to "ML Training" which is performed offline in dedicated GPU clusters. See Table 2 for more details.

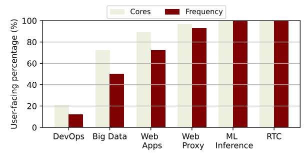
To validate the generality of the survey, we also manually labeled the remaining workloads from the division. Table 3 displays the distribution of workload classes in both the survey and the entire division. The distributions exhibit similar patterns, with "Web Apps" being the most common class, followed by "Big Data" and "DevOps".

Diverse workloads can be categorized into six main classes, out of which "Big Data", "Web Apps", and "RTC" accounts for the majority of core usage and thus should be the primary targets for implementing cloud optimizations.

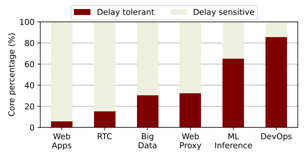
4.2 Performance requirements

User-facing. These are workloads which serve real-time traffic which expects a live human interaction (*e.g.*, web applications composed of front-end web servers and databases). These workloads typically exhibit periodic utilization patterns, with high activity during the day and low activity at night. They are commonly referred to as performance-critical workloads [31].

Figure 1a shows the distribution of user-facing nature with respect to workload class in terms of their frequency and percentage core usage. The "RTC" and "ML Inference" classes are completely user-facing. This is intuitive as "RTC" handles mostly online meetings and the majority of the "ML Inference" workloads support interactive functions (*e.g.*, text prediction, smart reply). As expected, the percentage of user-facing workloads is also high among "Web Apps" and "Web Proxy" classes.



(a) User-facing nature.



(b) Delay tolerance requirements.

Figure 1: Performance requirements per workload class.

"DevOps" contains the lowest number of user-facing workloads, which is also highly intuitive as they are mainly used for software development and IT operations. "Big Data" contains a similar portion of user-facing and non-user-facing workloads. This is due to the presence of diverse workloads such as data analytics, hot data-stores, and event hub workloads.

Considering the user-facing nature, "RTC" and "ML Inference" have high performance requirements while "DevOps" workloads are usually less critical.

Delay tolerance. Delay tolerant workloads have the flexibility to tolerate delays within a specified deadline, allowing for better resource management at the cost of increased service delays [55]. Around 56.4% of the workloads are delay sensitive and the remaining 43.6% are delay tolerant. However, delay-sensitive workloads account for the majority of the cores (75.5%). Workloads do not usually leverage delay-sensitivity information, resulting in users spending more money and cloud platforms dedicating more resources than necessary. Figure 1b shows the delay tolerance across workload classes. "DevOps" has the highest percentage of delay-tolerant cores, while "RTC" and "Web Apps" have a substantial number of delay-sensitive cores. This is expected as they are mostly user-facing workloads. For these workloads, the expected latency values are typically in the range of milliseconds to a few seconds. Workloads like "Big Data" require more time to process and analyze large amounts of data, while "ML Inference" has a large fraction of delay-tolerant cores (over 60%) due to user-delight features.

There is a significant opportunity to optimize the execution of predominantly delay-tolerant workloads (*e.g.*, "DevOps") for both workload owners and the cloud platform, such as through the use of Harvest VMs.

4.3 Reliability requirements

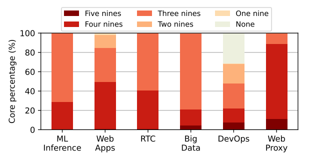
Availability. We first study the availability requirements [48, 52]. A highly available workload remains operational with minimal downtime in the event of a disruption (e.g., hardware failure, networking problems). An availability requirement of "five nines", "four nines", "three nines", "two nines", and "one nine" represent a downtime of 5

Class	Description	Cores	Frequency
Big Data	Database management operations (*e.g.*, ingestion, update) and various analytical (*e.g.*, descriptive, diagnostic, predictive, prescriptive, and cognitive) tasks over large or complex databases. It includes pipelines and event hubs that receive and process a large number of events per second.	32.4%	18.1%
Web Apps	Applications with a front-end, services with external APIs, and backend infrastructure that supports other applications. For example, a product offered by the cloud provider using an online interface.	27.3%	50.0%
Real-Time Comm (RTC)	Audio and video workloads (*e.g.*, online meetings) that support real-time communication.	24.1%	6.4%
ML Inference	Inference of machine learning models.	11.0%	4.8%
Web Proxy	Intermediary between a client application and the server. Examples are relay, proxy, and gateway.	3.9%	7.4%
DevOps	Tools (not part of final offering) for software development and IT operations. It covers code development, phased deployment (*e.g.*, staging, pre-production), and pipelines for continuous integration. Examples are internal workloads for testing and tools to accelerate engineers' debugging.	1.3%	13.3%

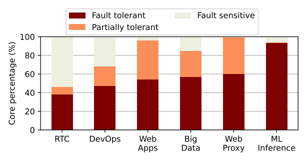
Table 2: Workload class descriptions with their core usage and frequencies.

	Web Apps	Big Data	DevOps	Web Proxy	RTC	ML inference
Survey	50.0%	18.0%	13.3%	7.4%	6.4%	4.8%
Division	50.9%	19.1%	17.0%	6.0%	4.2%	2.7%

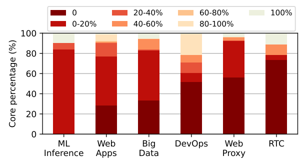
Table 3: Generality for the workload classes.



(a) Availability.



(b) Fault tolerance.



(c) Preemptibility.

Figure 2: Reliability requirements per workload class.

	Five	Four	Three	Two	One	None
Cores	2.4%	34.5%	58.0%	4.0%	0.5%	0.4%
Frequency	10.3%	25.3%	35.6%	14.4%	6.2%	8.2%

Table 4: Availability requirements.

minutes, 52 minutes, 8.77 hours, 3.65 days, and 36.53 days per year, respectively [12].

Table 4 shows that most workloads require an availability of "three" or "four nines". Only 10% of these workloads expect an availability of "five nines", with their daily core usage accounting for only 2.4% of the total daily usage. Availability is an important consideration for most workloads, given that only 8.2% of the workloads have no availability requirements.

Figure 2a shows the availability requirements for each workload class. The cores from the "Web Proxy", "ML Inference", and "RTC" classes expect an availability of at least "three nines". This could be attributed to the higher percentage of user-facing workloads in these classes (cf. Figure 1a). This is intuitive as "Web Proxy" workloads are often critical as they serve intermediate components between the client and server. The requirements for "DevOps" range from "one nine" to "five nines", and there is a significant percentage without any availability requirements (31.4%). This class usually contains workloads for software development and IT operations that are less critical to the functioning of the overall system. "DevOps" workloads may be more amenable to optimization and resource sharing than other types of workloads, since there is more flexibility in terms of when and how these tasks are run.

While a significant percentage of cores (95%) requires an availability greater or equal than "three nines", only a limited percentage require "five nines". "DevOps" workloads offer more flexibility in terms of when and how to run those workloads.

Fault tolerance. A fault tolerant workload can handle VM failures without impacting the service [32, 52]. A workload is fault-tolerant if its performance is insensitive to system failures (*e.g.*, a VM failure does not impact the operation or latency of a workload) [32]. Partially fault-tolerant workloads contain certain components that are tolerant to faults.

More than 50% of the workloads are completely fault tolerant(55.7% of total cores and 53.7% of total frequency). Specifically, "ML Inference" seems to be mostly fault tolerant (93.7%). This is because most of these workloads use frameworks that handle fault tolerance and re-route queries. Workloads that are partially fault tolerant contribute to 24.2% of the total core usage (23.9% of total frequency). Figure 2b shows the breakdown of fault tolerant cores with respect to workloads classes. Workloads that are sensitive to faults account for only 20.2% of the total core usage. "RTC" (53.7%) and "DevOps" workloads (31.8%) use a significant percentage of fault-sensitive cores. "RTC" workloads are part of communication system, which is latency sensitive, and this could be a potential reason for the presence of a significant percentage of fault sensitive cores.

Cores from the "ML Inference" class are highly fault tolerant, and this workload can remain operational with minimal downtime or data loss in the event of a disruption, whereas the

	0%	0-20%	20-40%	40-60%	60-80%	80-100%	100%
Cores	39.3%	41.1%	4.8%	6.5%	0.3%	1.8%	6.1%
Freq.	26.3%	22.2%	12.4%	16%	5.2%	4.6%	13.4%

Table 5: Preemptibility requirements.

"RTC" class contains the most percentage of fault-sensitive cores.

Preemptibility. While availability and fault tolerance relate to failures, preemptions are triggered by the cloud platform for efficiency reasons, so we consider preemptibility separately [19]. This refers to the ability to handle the loss of VM instances under normal operating conditions. For example, some workloads require all VMs to be up all the time (*i.e.*, 0% preemptibility) while others may allow 60% of the VMs to be running.

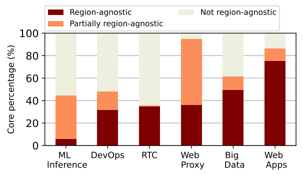
Table 5 shows the distribution of preemptible workloads. Around 60% of the workloads are preemptible by at least 0-20%. In addition, there is a small percentage of workloads that are 100% preemptible (13.4%). Highly preemptible services use significantly fewer cores than services with lower preemptibility. This suggests that these services are less complex or have fewer data dependencies.

Figure 2c shows the preemptibility requirements with respect to the workloads classes. "RTC" contains the highest number of cores (73.6%) with 0% preemptibility requirements. 84.1% of the "ML Inference" cores are at least 0-20% preemptible. On the other hand, "DevOps" shows diverse levels of preemptibility requirements that span from 0% to 80-100%. The higher percentage of "DevOps" cores with no preemptibility is due to their need to persist data as a part of streamlined packages and environments. Similarly, for "Web Proxy", most of the cores show preemptibility levels of 0% and 0-20%. This might be because these workloads serve as intermediate components between client and server and are often critical. "Web Apps" have a heterogeneous mix as it has a relatively high percentage of services with a preemptibility level of 0-20%, but also a relatively high percentage of services with a preemptibility level of 80% to 100%. This suggests that the preemtibility of "Web Apps" depends on the specific workload.

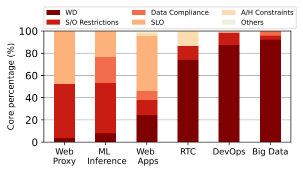
Workload preemtibility varies substantially between specific workloads. A good amount of "DevOps" and "ML Inference" workloads are suitable candidates to deploy on preemptible cores, while "Web Proxy" and "RTC" workloads are the last choice for preemptible cores.

4.4 Geographical requirements

We analyze the geographical requirements of various workloads and the factors restricting cross geo-migrations (*i.e.*, migrations spanning geographical regions). Region-agnostic workloads (RAW) can be deployed or migrated at least to any other region within a certain geo-locale without any negative impact on its operation. A workload is partially RAW if a certain percentage can be migrated or contains migratable components. These workloads are potential candidates for geographical load balancing during capacity shortages [1, 49]. Workloads that are fully RAW (*i.e.*, without any proximity and location constraints) represent over 50% of the workloads (47.5% of total cores and 58% of total frequency). Workloads that are not RAW contribute to 38.6% of total cores (22.9% of total frequency). Fully and partially RAWs contribute to more than 60% of the cores.



(a) Core usage of region-agnostic workloads.



(b) Factors affecting cross geo-migrations.

Figure 3: Location constraints per workload class.

Factors limiting cross-region migration We use open coding to categorize factors and free-text responses from workload owners. The annotators showed near-perfect agreement with a Cohen's kappa score of 0.93. Any disagreements were due to unclear responses from workload owners, which were resolved by reviewing available documentation and adopting appropriate labels. Table 6 summarizes the five classes obtained from open-coding and the associated definitions of each class. Some of the workloads are restricted by a single factor whereas others are restricted by a combination of them. The most frequent combinations are ("Workload Dependencies", "Security/Offering restrictions") and ("Workload Dependencies", "Architecture/Hardware"). To better understand the influence of each category, we consider each category independently, perform normalization, and calculate their core distribution and frequency. "Workload Dependencies" (51.9%) and "Security/Offering Restrictions" (21.3%) are the most common factors restricting cross geo-migrations. "Others" includes unclear responses and accounts for only 0.2% of the cores.

Figure 3 shows the region dependency for each workload class as well as the factors restricting them from cross geo-migrations. Some observations are as follows:

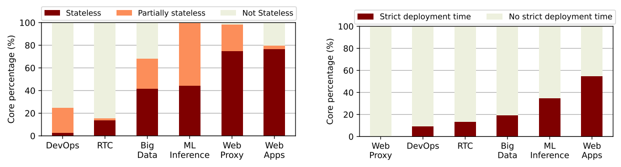
Web Apps dominate the region-agnostic cores (75.4%). This is due to their inherent nature of responding to ad-hoc requests from end users, without the need to store request state. They also rely on components, such as web servers, application servers, and databases, that can be distributed across multiple geographic locations to improve performance and availability. Due to the user-facing nature of these workloads, the major factor restriction is due to "SLO" (*e.g.*, round trip time and latency).

Big Data contains nearly an equal amount of region-agnostic, partial, and region-affine cores. Certain components of "Big Data" workloads such as data storage or processing components are more sensitive to geographic location than others and require low latency or high bandwidth connections to other components. The cross geo-migration here mainly depends on "Workload Dependency" as they often have data dependencies.

Web Proxy comprises mostly fully or partially region-agnostic cores. This is because although proxies are user-facing with "SLO" restrictions and data privacy concerns, a portion of its cores can be migrated due to their stateless and lack of dependency on local resources (c.f. Figure 4a).

Categories	Description	Cores	Frequency
Workload Dependencies (WD)	Upstream and downstream dependencies associated with a workload, local cache requirements or storage affinities.	51.9%	48.2%
Security/Offering (S/O)	Related to security or due to differences in features offered in different regions. Examples include restrictions on IP ranges.	21.3%	25.9%
Service Level Objectives (SLO)	Restrictions such as latency, RTT requirements, and performance requirements.	14.5%	8.2%
Data Compliance (DC)	Region-specific data handling standards. Examples include EU DB restrictions.	7.3%	10.6%
Architecture/Hardware (A/H)	Imposed by underlying fabric and deployment framework.	4.7%	4.7%
Others	Factors not belonging to any of the defined classes and unclear responses.	0.2%	2.35%

Table 6: Factors limiting cross geo-region migrations and their descriptions.



(a) Stateless vs. stateful.

(b) Deployment time requirements.

Figure 4: Scalability requirements per workload class.

RTC contains the highest percentage of region-affine cores and the major restricting factors are "Workload Dependencies" and "A/H constraints". These workloads are generally composed of multiple microservices and are compute heavy.

DevOps is restricted because of "Workload Dependencies" and "S/O restrictions" (similar to "RTC"). These restrictions are mainly due to the packages and environments often needed for "DevOps" workloads.

ML Inference contains the lowest percentage of region-agnostic cores. The major restricting factors are related to data privacy and security ("S/O" and "Data Compliance"). They also often offer interactive inference with location specific features and the cost of spinning them up in a new region is often too high.

A major portion of cores (61.4%) could be completely or partially deployed to other geo-locations. "Web Apps" workloads are potential candidates for cross geo-migrations. Overall, besides "SLO", "Workload dependency" and "Security/Offering" are more dominant restricting factors for cross geo-migrations.

4.5 Scalability requirements

Stateless vs stateful Stateless workloads can be easily scaled in/out as they require no state to be persisted. Over 50% of the workloads are stateless and their core usage amounts to 45.5%, where as stateful workloads constitute 37.1% of total core usage and 32.4% of total frequency. The breakdown of statefulness for each workload class is shown in Figure 4a. "Web Apps" and "Web Proxy" constitute most of the stateless cores, while the presence of stateless cores among "DevOps" and "RTC" workloads is significantly lower.

Stateless cores are prevalent in "Web Apps" and "Web Proxy" workloads, making them more flexible for scaling in/out. On the contrary, "DevOps" and "RTC" workloads are mostly stateful.

Deployment time. We define the deployment time requirements of a workload as strict if the VMs need to be deployed in less than one minute. Requirements greater than a minute are considered relaxed. Most workloads (71.5% of total core usage and 79.3% of total frequency) have relaxed deployment time requirements. Figure 4b shows the breakdown of deployment time requirements by workload class. Most workload classes, except for "Web Apps", have a low percentage of cores with strict deployment time requirements. The "Web Proxy" class contains the lowest percentage of cores with strict deployment time requirements (0.3%). "Web Proxy" workloads exhibit a better ability to handle an increase in demand with consistent QoS due to their stateless nature and relaxed deployment time requirements.

While most of the workloads don't have strict deployment time requirements, the majority of cores of the "Web Apps" class have a strict deployment time requirement, which should be carefully met when scaling out on demand.

5 Implications

5.1 Opportunities for cloud optimizations

Relaxed requirements. In this section, we use Table 1 to identify platform optimizations for different workloads. We focus on those with relaxed requirements, enabling specific optimizations to enhance performance. Table 7 shows the percentage of cores with relaxed characteristics per workload class, revealing unique profiles. For example, "Web Apps" are stateless, delay-sensitive, low in preemptibility, flexible for cross geo-migrations, and have strict deployment time requirements.

Results. We analyzed cloud optimization techniques to determine the percentage of suitable cores for each workload class. Our analysis revealed significant opportunities for "Auto-scaling," "Spot VMs," "Underclocking," "Overclocking," "Region-agnostic" deployments, and "MA datacenters." By leveraging these optimizations on applicable workload classes, cloud providers can monetize their unallocated capacity, reduce costs, and lower CO_2 emissions. However, some workload classes show lower opportunities than others for certain cloud optimizations due to the dominance of delay-sensitive cores. Making workloads less delay-sensitive can expand opportunities for "Harvest VMs" and "VM Oversubscription".
We discuss the particularities for each class in more detail.

Relaxed requirements	Web Apps	Big Data	DevOps	Web Proxy	RTC	ML Inference	Overall
User-facing	89.1%	72.3%	20.8%	96.4%	100%	100%	86.9%
No proximity & location constraints	75.3%	49.6%	31.6%	36.1%	34.7%	6.2%	47.5%
Delay tolerant	5.7%	30.2%	85.6%	32.5%	15.1%	65.2%	24.5%
Low availability (< Four nines)	50.4%	79.2%	78.1%	11.1%	59.3%	71.3%	63.0%
High preemptibility (> 20%)	23.1%	17.0%	39.5%	7.6%	21.6%	15.9%	19.6%
Fault tolerant	54.3%	57.0%	47.4%	60.0%	37.8%	93.8%	55.7%
Stateless	76.4%	43.1%	2.9%	74.8%	3.7%	44.5%	45.5%
No deployment time requirements	45.1%	74.4%	98.5%	99.6%	86.2%	63.0%	68.8%

Table 7: Percentage of cores with relaxed workload characteristics for each class.

	Web Apps	Big Data	DevOps	Web Proxy	RTC	ML Inference	Overall
Auto-scaling	**26.2%**	**34.4%**	**88.1%**	**95.6%**	13.0%	**63.0%**	**33.1%**
Spot VMs	**23.1%**	**22.0%**	**39.6%**	7.6%	**22.0%**	19.0%	**21.6%**
Harvest VMs	3.4%	2.8%	**27.9%**	0.6%	10.5%	12.0%	6.4%
Overclocking	**26.6%**	**45.4%**	0.3%	**56.0%**	**72.0%**	0.0%	**41.0%**
Underclocking	**34.4%**	**27.9%**	**99.0%**	**39.8%**	**26.9%**	**70.0%**	**36.0%**
Non pre-provision	**45.1%**	**74.4%**	**98.5%**	**99.6%**	**86.2%**	**63.0%**	**68.8%**
Region-agnostic	**75.3%**	**31.7%**	**31.1%**	**36.4%**	**35.2%**	6.94%	**43.0%**
VM oversubscription	3.3%	1.6%	19.3%	**24.2%**	13.1%	12.2%	7.6%
MA datacenters	**51.0%**	**72.0%**	**85.1%**	11.2%	**59.0%**	**70.0%**	**59.6%**
VM rightsizing	0.74%	0.0%	0.0%	**23.1%**	0.0%	7.3%	2.1%

Table 8: Percentage of cores suitable for each cloud platform optimization. Bold and underlined values indicate percentage cores greater than 20 and 50, respectively.

Web Apps Most of these workloads can be deployed across multiple regions and take advantage of lower cost resources, reduce CO_2 emissions, and the impact of regional outages. They can be throttled during infrastructure failures or maintenance events (*i.e.*, "MA DC"). A significant percentage can be deployed on cost-effective "Spot VMs" for lower costs. "Auto-scaling" is another feature that can be used to optimize the its efficiency. "Overclocking" and "Underclocking" can also be used to improve workload performance and reliability.

Big Data Most of these workloads do not require "Pre-provisioning" (~ 74%) and over 70% of cores can enable "MA DCs". These workloads can also leverage "Underclocking/Overclocking". Similarly, 30% of the cores are suitable for "Region-agnostic" deployments.

DevOps A significant portion of these workloads can benefit from "Auto-scaling", "MA DCs", and do not require the "Pre-provisioning" of resources. Additionally, "Underclocking" can help reduce power consumption and save costs, which is especially important for "DevOps" teams that manage large-scale infrastructure. **Web Proxy** Cloud providers can enable "Auto-scaling" for most of these workloads to reduce over-provisioning of VMs without the need for "Pre-provisioning". "Overclocking" and "Underclocking" can also be used to improve workload performance and reliability. These workloads are also a good target for "Rightsizing". **RTC** "Overclocking" can enhance the performance and reliability of these workloads. Even small improvements in performance can have a significant impact on user experience. These workloads do not require pre-provisioning and can benefit from the use of "MA DCs" to improve reliability and reduce the impact of regional outages or performance issues.

ML Inference A substantial portion of the cores from these workloads are suitable for "Auto-scaling" and do not need "Pre-provision". These workloads also enable early throttling in the event of failures with "MA DCs" and can benefit from "Underclocking".

Since different workload classes benefit from different sets of cloud optimizations, identifying the workload class is a good start at determining what optimizations to enable. This helps achieve greater cloud efficiency and reliability while minimizing users' costs.

5.2 Cloud platform adoption

To validate our proposed cloud platform optimizations, we analyzed one representative workload per class and contacted its owner to verify the recommendations' appropriateness and effectiveness. Table 9 summarizes, for each workload class, the recommended platform optimizations, their current implementation status (enabled or not), and the owner's assessment of their readiness. For example, we found that the chosen "Web Proxy" workload is well-suited for "Overclocking," accounting for 56% of total core usage among its peers. Although this workload currently does not use "Overclocking," its owner noted that enabling it could reduce the overall core count and better handle peak loads with fewer physical cores.

While many suggested optimizations were accepted by workload owners, most remain disabled. Enabling them would benefit both cloud users and providers; however, practical constraints—such as limits on the number of simultaneous optimizations and compatibility issues—may apply. Future work will quantify these benefits to further validate the approach.

Workload class	Suggested optimization	Enabled	Ready to adopt
Web Proxy	Overclocking	×	✓
	Disable pre-provision	×	☐
Big Data	Region-agnostic	✓	✓
	Auto-scaling	×	✓
	Disable pre-provision	✓	✓
RTC	Overclocking	×	✓
	Disable pre-provision	×	✓
	MA datacenters	×	☐
DevOps	Spot VMs	×	☐
	Underclocking	×	☐
	Disable pre-provision	×	☐
	MA datacenters	×	☐
	Harvest VMs	×	✓
	Auto-scaling	✓	✓
	Region-agnostic	×	✓
ML Inference	Auto-scaling	✓	✓
	Disable pre-provision	×	✓
	Underclocking	×	✓
	MA datacenters	×	☐
Web Apps	Underclocking	×	✓
	Auto-scaling	×	✓
	Disable pre-provision	×	✓
	Spot VMs	×	☐
	Harvest VMs	×	☐

Table 9: Case studies evaluating suitability of optimizations. Box symbol (☐) indicates recommended optimizations that are beneficial but not ready to implement due to limitations such as engineering efforts.

6 Related Work

Resource management and ML-centric approaches. Some works optimize resource utilization while meeting performance using limited workload information such as QoS constraints or resource usage [2, 20, 21, 28, 63]. Other works [31, 38, 60] propose optimizations that assume knowledge of the workload characteristics. There are also ML-centric approaches exploring how and where ML should be infused in cloud platforms [11, 18, 54]. Our work is complimentary to these efforts as our insights enable these approaches and motivate future research into improving existing resource management/optimizations.

Workload profiling and characterization. Related works in this area often study workload characteristics related to resource utilization and workload deployment, providing insights on the heterogeneity and disparity in those characteristics [13–15, 18, 22, 24, 29, 33–35, 35–37, 41, 42, 45–47, 62]. Other studies explore workload characteristics including failure distribution, correlation, arrival rate, interference between resources, volatility of resource demand, usage by region and customer segments, and microservice dependencies [26, 30, 36, 50, 51]. The closest literature to this work are studies that adopt a data-driven approach to identify opportunities and design challenges to enable ML inference locally on smartphones and other edge platforms [43, 57]. However, these studies often focus on workload signals or characteristics directly derived from a single workload signal/telemetry, while this work focuses on uncovering fundamental workload characteristics and deriving insights for reliability and efficiency improvements.

7 Conclusion and Future Work

In this work, we conduct the first large-scale study of first-party workloads at Microsoft. This study provides a comprehensive understanding of the characteristics of first-party workloads and their variations across multiple workloads. The study also identifies unique workload characteristics and suitable workloads for each cloud platform optimization. The findings of this study can be leveraged to improve the efficiency and reliability of cloud platforms. These findings can also assist cloud users in decreasing the overall cost and enhancing performance. Overall, this study contributes to the ongoing effort to optimize cloud services by providing a deeper understanding of workload characteristics and their impact on cloud performance.

References

[1] Muhammad Abdullah Adnan, Ryo Sugihara, and Rajesh K Gupta. 2012. Energy efficient geographical load balancing via dynamic deferral of workload. In *CLOUD*.

[2] Omid Alipourfard, Hongqiang Harry Liu, Jianshu Chen, Shivaram Venkataraman, Minlan Yu, and Ming Zhang. 2017. CherryPick: Adaptively Unearthing the Best Cloud Configurations for Big Data Analytics. In *NSDI*.

[3] Amazon. 2023. Amazon Web Services. https://aws.amazon.com/.

[4] Amazon Elastic Compute Cloud. 2019. Amazon EC2 Spot Instances. https://aws.amazon.com/ec2/spot/.

[5] Amazon Elastic Compute Cloud. 2020. Running batch jobs at scale for less. https://aws.amazon.com/getting-started/hands-on/run-batch-jobs-at-scale-with-ec2-spot/.

[6] Amazon Web Services. 2023. Amazon EC2 Auto Scaling. https://aws.amazon.com/ec2/autoscaling/.

[7] Pradeep Ambati, Íñigo Goiri, Felipe Frujeri, Alper Gun, Ke Wang, Brian Dolan, Brian Corell, Sekhar Pasupuleti, Thomas Moscibroda, Sameh Elnikety, Marcus Fontoura, and Ricardo Bianchini. 2020. Providing SLOs for Resource-Harvesting VMs in Cloud Platforms. In *OSDI*.

[8] Microsoft Azure. 2019. Use low-priority VMs with Batch. https://docs.microsoft.com/en-us/azure/batch/batch-low-pri-vms.

[9] Microsoft Azure. 2020. Azure Spot Virtual Machines. https://azure.microsoft.com/en-us/pricing/spot.

[10] Ataollah Fatahi Baarzi and George Kesidis. 2021. Showar: Right-sizing and efficient scheduling of microservices. In *SoCC*.

[11] Ricardo Bianchini, Marcus Fontoura, Eli Cortez, Anand Bonde, Alexandre Muzio, Ana-Maria Constantin, Thomas Moscibroda, Gabriel Magalhaes, Girish Bablani, and Mark Russinovich. 2020. Toward ML-Centric Cloud Platforms. *Commun. ACM* (2020).

[12] BMC Blogs. 2020. Service Availability. https://www.bmc.com/blogs/service-availability-calculation-metrics/.

[13] Wenyan Chen, Kejiang Ye, Yang Wang, Guoyao Xu, and Cheng-Zhong Xu. 2018. How does the workload look like in production cloud? Analysis and clustering of workloads on alibaba cluster trace. In *ICPADS*.

[14] Yanpei Chen, Archana Sulochana Ganapathi, Rean Griffith, and Randy H Katz. 2010. Analysis and Lessons from a Publicly Available Google Cluster Trace. *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2010-95* 94 (2010).

[15] Yue Cheng, Zheng Chai, and Ali Anwar. 2018. Characterizing co-located datacenter workloads: An alibaba case study. In *Asia-Pacific Workshop on Systems*.

[16] Google Cloud. 2020. Preemptible VM Instances. https://cloud.google.com/compute/docs/instances/preemptible.

[17] Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement* 20, 1 (1960), 37–46.

[18] Eli Cortez, Anand Bonde, Alexandre Muzio, Mark Russinovich, Marcus Fontoura, and Ricardo Bianchini. 2017. Resource Central: Understanding and predicting workloads for improved resource management in large cloud platforms. In *SOSP*.

[19] Konrad Cłapa and Brian Gerrard. 2021. *FProfessional Cloud Architect Google Cloud Certification Guide*. Packt Publishing.

[20] Christina Delimitrou and Christos Kozyrakis. 2013. Paragon: QoS-aware scheduling for heterogeneous datacenters. In *ASPLOS*.

[21] Christina Delimitrou and Christos Kozyrakis. 2014. Quasar: Resource-efficient and qos-aware cluster management. In *ASPLOS*.

[22] Sheng Di, Derrick Kondo, and Franck Cappello. 2013. Characterizing cloud applications on a Google data center. In *ICPP*.

[23] Alexander Fuerst, Stanko Novaković, Íñigo Goiri, Gohar Irfan Chaudhry, Prateek Sharma, Kapil Arya, Kevin Broas, Eugene Bak, Mehmet Iyigun, and Ricardo Bianchini. 2022. Memory-harvesting VMs in Cloud Platforms. In *ASPLOS*.

Division	Workloads*	#VMs*	Cores*	Regions
1	15.93%	82.91%	79.85%	55
2	12.21%	11.04%	13.52%	49
3	3.76%	5.15%	5.76%	46
4	1.91%	0.62%	0.57%	31
5	0.82%	0.14%	0.16%	22
6	1.57%	0.08%	0.03%	35
7	1.73%	0.02%	0.03%	28
8	0.84%	0.03%	0.06%	27
9	0.76%	0.01%	0.01%	9
10	0.22%	0.01%	0.01%	34
Rest (4)	0.91%	0.01%	0.01%	30

Table 10: Characteristics of the workloads across divisions, with values marked with an asterisk (*) indicating the percentage values relative to all of Microsoft.

[24] Peter Garraghan, Paul Townend, and Jie Xu. 2013. An Analysis of the Server Characteristics and Resource Utilization in Google Cloud. In *IC2E*. IEEE.

[25] Google. 2023. Google Cloud. https://cloud.google.com/.

[26] Jing Guo, Zihao Chang, Sa Wang, Haiyang Ding, Yihui Feng, Liang Mao, and Yungang Bao. 2019. Who limits the resource efficiency of my datacenter: An analysis of Alibaba datacenter traces. In *IWQoS*.

[27] Majid Jalili, Ioannis Manousakis, Íñigo Goiri, Pulkit A Misra, Ashish Raniwala, Husam Alissa, Bharath Ramakrishnan, Phillip Tuma, Christian Belady, Marcus Fontoura, et al. 2021. Cost-efficient overclocking in immersion-cooled datacenters. In *ISCA*.

[28] Seyyed Ahmad Javadi, Amoghavarsha Suresh, Muhammad Wajahat, and Anshul Gandhi. 2019. Scavenger: A black-box batch workload resource manager for improving utilization in cloud environments. In *SoCC*.

[29] Congfeng Jiang, Guangjie Han, Jiangbin Lin, Gangyong Jia, Weisong Shi, and Jian Wan. 2019. Characteristics of co-allocated online services and batch jobs in internet data centers: a case study from Alibaba cloud. *IEEE Access* 7 (2019), 22495–22508.

[30] Cinar Kilcioglu, Justin M Rao, Aadharsh Kannan, and R Preston McAfee. 2017. Usage patterns and the economics of the public cloud. In *WWW*.

[31] Alok Gautam Kumbhare, Reza Azimi, Ioannis Manousakis, Anand Bonde, Felipe Frujeri, Nithish Mahalingam, Pulkit A Misra, Seyyed Ahmad Javadi, Bianca Schroeder, Marcus Fontoura, et al. 2021. Prediction-Based Power Oversubscription in Cloud Platforms. In *USENIX ATC*.

[32] Nuno Laranjeiro and Marco Vieira. 2007. Towards fault tolerance in web services compositions. In *Proceedings of the 2007 Workshop on Engineering Fault Tolerant Systems*. 2–8.

[33] Bingwei Liu, Yinan Lin, and Yu Chen. 2016. Quantitative workload analysis and prediction using Google cluster traces. In *INFOCOM Workshops*.

[34] Zitao Liu and Sangyeun Cho. 2012. Characterizing machines and workloads on a Google cluster. In *ICPP Workshops*.

[35] Chengzhi Lu, Kejiang Ye, Guoyao Xu, Cheng-Zhong Xu, and Tongxin Bai. 2017. Imbalance in the cloud: An analysis on alibaba cluster trace. In *Big Data*.

[36] Shutian Luo, Huanle Xu, Chengzhi Lu, Kejiang Ye, Guoyao Xu, Liping Zhang, Yu Ding, Jian He, and Chengzhong Xu. 2021. Characterizing microservice dependency and performance: Alibaba trace analysis. In *SoCC*.

[37] Ashraf Mahgoub, Edgardo Barsallo Yi, Karthick Shankar, Eshaan Minocha, Sameh Elnikety, Saurabh Bagchi, and Somali Chaterji. 2022. WISEFUSE: Workload Characterization and DAG Transformation for Serverless Workflows. *POMACS* (2022).

[38] Ming Mao and Marty Humphrey. 2011. Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In *SC*.

[39] Microsoft. 2023. Azure. https://azure.microsoft.com/.

[40] Microsoft Azure. 2022. Overview of autoscale in Microsoft Azure. https://docs.microsoft.com/en-us/azure/azure-monitor/autoscale/autoscale-overview.

[41] Asit K Mishra, Joseph L Hellerstein, Walfredo Cirne, and Chita R Das. 2010. Towards characterizing cloud backend workloads: insights from Google compute clusters. *SIGMETRICS* 37, 4 (2010), 34–41.

[42] Md Rasheduzzaman, Md Amirul Islam, Tasvirul Islam, Tahmid Hossain, and Rashedur M Rahman. 2014. Task shape classification and workload characterization of google cluster trace. In *IACC*.

[43] Vijay Janapa Reddi, Christine Cheng, David Kanter, Peter Mattson, Guenther Schmuelling, Carole-Jean Wu, Brian Anderson, Maximilien Breughe, Mark Charlebois, William Chou, et al. 2020. MLPerf inference benchmark. In *ISCA*.

[44] Benjamin Reidys, Jinghan Sun, Anirudh Badam, Shadi Noghabi, and Jian Huang. 2022. BlockFlex: Enabling Storage Harvesting with Software-Defined Flash in Modern Cloud Platforms. In *OSDI*.

[45] Charles Reiss, Alexey Tumanov, Gregory R Ganger, Randy H Katz, and Michael A Kozuch. 2012. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In *SoCC*.

[46] Charles Reiss, Alexey Tumanov, Gregory R Ganger, Randy H Katz, and Michael A Kozuch. 2012. Towards understanding heterogeneous clouds at scale: Google trace analysis. *Intel Science and Technology Center for Cloud Computing, Tech. Rep* 84 (2012), 1–21.

[47] Mohammad Shahrad, Rodrigo Fonseca, Íñigo Goiri, Gohar Chaudhry, Paul Batum, Jason Cooke, Eduardo Laureano, Colby Tresness, Mark Russinovich, and Ricardo Bianchini. 2020. Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider. In *USENIX ATC*.

[48] Mohammad Shahrad and David Wentzlaff. 2016. Availability Knob: Flexible User-Defined Availability in the Cloud. In *SoCC*.

[49] Jiuchen Shi, Kaihua Fu, Quan Chen, Changpeng Yang, Pengfei Huang, Mosong Zhou, Jieru Zhao, Chen Chen, and Minyi Guo. 2022. Characterizing and orchestrating VM reservation in geo-distributed clouds to improve the resource efficiency. In *SoCC*.

[50] Huangshi Tian, Yunchuan Zheng, and Wei Wang. 2019. Characterizing and synthesizing task dependencies of data-parallel jobs in alibaba cloud. In *SoCC*.

[51] Muhammad Tirmazi, Adam Barker, Nan Deng, Md E Haque, Zhijing Gene Qin, Steven Hand, Mor Harchol-Balter, and John Wilkes. 2020. Borg: The Next Generation. In *EuroSys*.

[52] Astrid Undheim, Ameen Chilwan, and Heegaard. 2021. Differentiated Availability in Cloud Computing SLAs. In *Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing*. 129–136.

[53] Cheng Wang, Bhuvan Urgaonkar, Neda Nasiriani, and George Kesidis. 2017. Using burstable instances in the public cloud: Why, when and how? *POMACS*.

[54] Yawen Wang, Kapil Arya, Marios Kogias, Manohar Vanga, Aditya Bhandari, Neeraja J Yadwadkar, Siddhartha Sen, Sameh Elnikety, Christos Kozyrakis, and Ricardo Bianchini. 2021. SmartHarvest: Harvesting idle CPUs safely and efficiently in the cloud. In *EuroSys*.

[55] Philipp Wiesner, Ilja Behnke, Dominik Scheinert, Kordian Gontarska, and Lauritz Thamsen. 2021. Let's wait awhile: how temporal workload shifting can reduce carbon emissions in the cloud. In *Middleware*.

[56] Dan Williams, Hani Jamjoom, Yew-Huey Liu, and Hakim Weatherspoon. 2011. Overdriver: Handling Memory Overload in an Oversubscribed Cloud. *VEE* (2011).

[57] Carole-Jean Wu, David Brooks, Kevin Chen, Douglas Chen, Sy Choudhury, Marat Dukhan, Kim Hazelwood, Eldad Isaac, Yangqing Jia, Bill Jia, et al. 2019. Machine learning at Facebook: Understanding inference at the edge. In *HPCA*.

[58] Randolph Yao, Chuan Luo, Bo Qiao, Qingwei Lin, Tri Tran, Gil Lapid Shafriri, Yingnong Dang, Raphael Ghelman, Pulak Goyal, Eli Cortez, et al. 2021. Infusing ML into VM Provisioning in Cloud. In *CloudIntelligence*.

[59] Randolph Yao, Chuan Luo, Bo Qiao, Qingwei Lin, Tri Tran, Gil Lapid Shafriri, Yingnong Dang, Raphael Ghelman, Pulak Goyal, Eli Cortez, Daud Howlader, Sushant Rewaskar, Murali Chintalapati, and Dongmei Zhang. 2021. Infusing ML into VM Provisioning in Cloud. In *CloudIntelligence*.

[60] Chaojie Zhang, Alok Gautam Kumbhare, Ioannis Manousakis, Deli Zhang, Pulkit A Misra, Rod Assis, Kyle Woolcock, Nithish Mahalingam, Brijesh Warrier, David Gauthier, et al. 2021. Flex: High-availability datacenters with zero reserved power. In *ISCA*.

[61] Chengliang Zhang, Minchen Yu, Wei Wang, and Feng Yan. 2019. MArk: Exploiting Cloud Services for Cost-Effective, SLO-Aware Machine Learning Inference Serving. In *USENIX ATC*.

[62] Qi Zhang, Joseph Hellerstein, and Raouf Boutaba. 2011. Characterizing Task Usage Shapes in Google Compute Clusters. (2011).

[63] Yanqi Zhang, Weizhe Hua, Zhuangzhuang Zhou, G Edward Suh, and Christina Delimitrou. 2021. Sinan: ML-based and QoS-aware resource management for cloud microservices. In *ASPLOS*.

A Generality of the Study

To evaluate the generality of the results, we compare the general characteristics of the sample workloads in our survey with the rest of the division and across divisions. Table 11 shows the corresponding values. This data indicate that the characteristics of the workloads in the sample are very similar to the rest of the division and the company and that our survey samples are representative. Moreover, Table 10 supports our conclusion that the selected division is a good representation of Microsoft.

The workload classes evolved as of February 2025 due to the significant shift in the industry. Table 12 shows the latest distribution (Class and Frequency).

	Survey	Selected division	Overall
Number of workloads	188	1034	4275
Number of regions	49	49	55
P50 & P99 Avg CPU	[5.5%, 91.0%]	[5.2%, 95.7%]	[5.6%, 74.9%]
P50 & P99 Max CPU	[35.2%, 99.2%]	[28.2%, 99.9%]	[18.7%, 99.2%]
P50 & P99 Avg Memory	[30.1%, 100.0%]	[29.9%, 100.0%]	[31.2%, 100.0%]
P50 & P99 Max Memory	[39.1%, 100.0%]	[37.9%, 100.0%]	[40.4%, 100.0%]
P50 & P99 #VMs	[353, 34K]	[43, 344K]	[12K, 29M]
P50 & P99 VMs lifetime (h)	[646, 23,320.42]	[4, 18,751.72]	[0, 21,411]

Table 11: Comparison of characteristics of the workloads in the survey, selection division, and overall at Microsoft.

Workload Class	Frequency
Web Apps/Services	36.7%
DevOps	24.8%
Big Data	7.7%
Generative AI workloads	5.3%
RTC	4.9%
Web Proxy	2.5%
ML Inference	1.3%
Others	2%

Table 12: Workload class and frequencies.

"Generative AI workloads" refer to the computational tasks and processes involved in creating or generating new content, data, or models that mimic or replicate human-like outputs. These workloads are powered by Generative Artificial Intelligence technologies, which leverage machine learning models, particularly generative models, to produce outputs that can include text, images, videos, music, code, and more.

Though, Web Apps/Services still dominates, the frequency of workloads from each class shows significant changes. Optimizing the performance of newly emerged workloads is a future work.