Cross Platform Analysis of Cloud Utilization Patterns for Optimized Resource Allocation

Akshat Karwa
akarwa7@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia, USA

Mehul Rastogi
mehulrastogi@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia, USA

1 Project Objective

In this project, we compare some of the largest cloud service providers: *Google Cloud, Microsoft Azure, Alibaba Cloud*. Utilizing their publicly available datasets (released about twice a decade), we delve deep into their unique resource utilization patterns and analyze the trends in their cluster trace data.

Moreover, we quantify ***the utilization gap*** - the discrepancy between the resources allocated and the actual utilization. Through this, we understand the magnitude and patterns of inefficiencies in the workings of cloud providers. Furthermore, we contrast the strategies of resource allocation that the different providers utilize.

We then evaluate how effectively the clusters handle workloads under diverse conditions and observe the unpredictable changes in workloads such as spikes or seasonal variations or changing usage patterns with time. We identify the best resource management techniques that can be generalized as the best practices in the industry and also look at provider-specific approaches that have the best outcomes.

Overall, we believe that our insights can help reduce over allocation of resources, reduce idle capacity at most times and consequently lower operational costs. Through this, we hope to provide the community with a better understanding of resource patterns and management strategies.

2 Motivation

With the continuous growth of Cloud computing, efficient resource management has become a critical challenge. Our project is motivated primarily by the utilization gap that leads to over-provisioning, idle capacity, and suboptimal responses to variations in workloads leading to unnecessary costs and reduced efficiency.

Individual studies in the past have analyzed traces from specific providers. Our comparative analysis examines different providers and how they handle challenges. There is a lack of valuable insights into common patterns and unique approaches of different cloud providers, and there is limited research in comparative analysis across platforms.

The economic stakes are substantial, Microsoft's research [2] demonstrates that companies with large cloud deployments can gain valuable insights through deep trace analysis. The Azure dataset we work with alone comprising of 2.7 Million VMs and 469.4 Million core hours and thus, even small improvements in resource utilization can significantly save costs and benefit the environment by reducing data center energy consumption. Through this project, we can potentially help companies reduce operational costs and make better design decisions. Generalizable insights would lead to more accurate resource allocation and reduce the costs of operating data centers.

,
2025.

3 Related Work

Previous research has primarily focused on analyzing cloud traces from individual providers instead of conducting comparative analyses across different cloud platforms.

For Google Cloud, Reiss et al. [1] completed a comprehensive analysis of Google traces to reveal a lot of heterogeneity in the cluster workloads also identifying inefficiencies in infrastructure. Similarly, Lu et al. [3] analyzed Alibaba's trace data and highlighted imbalances during resource allocation. Their findings shows how workload characteristics keep impacting resource utilization. For Microsoft Azure, Cortez et al. [2] developed Resource Central, a system which predicts workloads for improved resource management in large cloud platforms. Their research leveraged Azure's large dataset and demonstrated the value of trace data analysis in optimizing cloud infrastructure.

The gap in comparative research across cloud providers presents a significant opportunity. There are many individual trace analyses which provide valuable insights into specific platforms but they don't address whether observed patterns are provider-specific or represent industry-wide phenomena. These studies do not show any common issues or common effective techniques. This limits our ability to develop generalizable principles for resource management. Our work aims to address this gap.

4 Challenges

While working on this project, we faced many challenges. The main challenges, that led to several bottlenecks and efficiency constraints in the project are:

The trace data which we have used from all the three sources: Alibaba, Google, and Azure is extremely heterogeneous. Data from different cloud providers has different schema and relationship models. The metrics calculated and reported by each of them are also completely different. Due to this, it became exceedingly difficult to perform analysis which compares and contrasts the trace data across the three providers in an efficient manner. For example, Azure doesn't have data on memory usage. It only contains data on CPU/compute usage.

A big challenge, also was to analyze the huge datasets with the limited amount of compute resources we have available. It was challenging to efficiently process the data from all the three sources because of their sheer magnitude. Therefore, compute bottlenecks was an extremely large challenge for this project. For example, we have Azure's data, for approximately ≈ 2.7 million virtual machines. For Alibaba, we have the data for 4000 machines for a period of 8 days. For Google, the dataset is ≈ 2.14 TiB compressed.

Another challenge was the lack of available documentation to efficiently understand the schema for the Alibaba traces. Different approaches needed to be tried and tested in-order to find the correct set of tables to merge in-order to get accurate values that represent the metrics we are looking for. On top of that, the three cloud providers have different mechanisms for job/task scheduling and processing and how they tackle workloads. This made it extremely difficult to find generalizable insights across the three datasets. The timestamp durations for which the data is given for each is different as well. We used different technologies for different datasets, BigQuery, Pandas, etc. in-order to utilize the most efficient way to process and analyze data.

Since the variables present in the each of trace datasets are different, doing predictive modeling across the three cloud providers was hard. Even though, we ended up using the same XGBoost model for all predictions, since the parameters / variables are very different across the cloud providers, the prediction accuracies and RMSE end up being completely different. Due to this, giving generalizable trends, became even harder. Therefore, finding the right variables to perform predictive analysis and generalize trends for compute/memory usage across three providers was another major challenge.

5 Approach

The resource utilization patterns across public trace data from Google Cloud, Alibaba Cloud, and Microsoft Azure have been compared.

5.1 Data

The following datasets have been analyzed:

- **Google Cluster Data (2019)** [5]
- **Alibaba Cluster Trace Program Data (2018)** [7]
- **Microsoft Azure Public Dataset (2019)** [9]

All cloud providers store data differently and we will address the heterogeneous nature of the three datasets and normalize them to a common format.

5.2 High Level Approach

The data provided by each of the cloud providers is extremely heterogeneous. There are a lot of variations in the manner in which these cloud providers store data.

We pre-processed the data in an efficient manner in-order to generate a homogeneous dataset to work with, which eventually helped us in doing a comparative analysis between the three cloud providers.

We performed exploratory data analysis in isolation for each cloud provider which helped us in understanding the nuances and schema associated with each cloud provider.

We performed standardized robust and informative plotting in-order to draw comparisons between the trace data from each of the three cloud providers.

Moreover, for each of the three trace datasets we conducted predictive training, modeling, and tuning using the XGBoost Regression Machine Learning model to accurately predict CPU and memory utilization rates. Different variables / parameters were used depending on the trace data, for this prediction process.

Finally, we consolidated all the data and plots, and performed robust comparative analysis which lead to insightful conclusions.

5.3 Schema

5.3.1 Google

Google's data for each cluster is divided into multiple cells from a to h - **(a, b, c, d, e, f, g, h)**. The data for each cell is in a folder with name - clusterdata_2019_{x}. Inside each cell, data is split into shards. For each shard, we have 5 tables. The 2 tables that we are interested in for our use case are:

- `Instance Events`
- `Instance Usage`

Schema of the `Instance Events` :

time, **type**, **collection_id**, **scheduling_class**, **missing_type**, **collection_type**, **priority**, **alloc_collection_id**, **instance_index**, **machine_id**, **alloc_instance_index**, **resource_request**, **constraint**

Schema of the `Instance Usage` :

start_time, **end_time**, **alloc_collection_id**, **collection_type**, **average_usage**, **maximum_usage**, **sample_rate**, **cpu_usage_distribution**, **tail_cpu_usage_distribution**

Detailed in-depth schema for each table and relationship diagrams between entities can be found at [4].

GitHub for Google Trace Data - [5]

5.3.2 Azure

The Azure dataset comprises **2.7M** VMs spanning **6.7K** subscriptions and **33.2K** deployments, with **162.5M** VM hours and **469.4M** core hours of utilization data.

VM table is the main data table, which we are utilizing for our analysis. The schema for the VM Table data is as follows:

vmid, **subscriptionid**, **deploymentid**, **vmcreated**, **vmdeleted**, **maxcpu**, **avgcpu**, **p95maxcpu**, **vmcategory**, **vmcorecountbucket**, **vmmemorybucket**, **lifetimecorehour**

The schema is in the **schema.csv** file at [8].

The cumulative data is in the files:
vmtable.csv and **deployments.csv**.

Azure Cluster Data: [9] Links to Download: [8]

5.3.3 Alibaba

Alibaba's cluster trace data includes data for 4000 machines for a periods of 8 days and consists of 6 tables: `Machine Meta`, `Machine Usage`, `Container Meta`, `Container Usage`, `Batch Task`, and `Batch Instance`.
The schema for all the tables is available in the schema details link below. The main table for our use case would be `Batch Instance` and `Machine Usage`.

The schema for `Batch Instance` is as follows:

instance_name, **task_name**, **job_name**, **task_type**, **status**, **start_time**, **end_time**, **machine_id**, **seq_no**, **total_seq_no**, **cpu_avg**, **cpu_max**, **mem_avg**, **mem_max**

The schema for `Machine Usage` is as follows:

machine_id, **time_stamp**, **cpu_util_percent**, **mem_util_percent**, **mem_gps**, **mkpi**, **net_in**, **net_out**, **disk_io_percent**

Alibaba Trace Data Schema: [6], Alibaba Cluster Data: [7]

5.4 Data Processing

5.4.1 Google

The data used is from one cluster (a) out of the 8 clusters (a through h) in the dataset. We processed our implementation for $\approx$ 10 million rows of data.

We merged the *instance_events_data* and *instance_usage_data* tables using *instance_index*, *machine_id*, and *collection_id* as join keys. The join performed was an inner-join.

Through this merge, we were able to correlate resource requests with actual utilization patterns across all the instances. After getting the resource allocation and utilization data, we calculated the normalized resource utilization (percentage). The final homogeneous data table that we got for Google Trace data is:

start_time	end_time	collection_id	machine_id	type	scheduling_class
2100000000	2330000000	226455519451	198555643071	2	2
300000000	600000000	215354841713	22338307	2	1
600000000	900000000	244773171840	1128087483	7	3
2987000000	2988000000	104894292360	20935939	3	3
600000000	900000000	220585838132	71880674234	0	2

priority	cpus_util_perc	mem_util_perc	cpus_max_util_perc	mem_max_util_perc
200	5.708092	60.644531	24.566474	60.742188
200	6.961634	1.644036	31.064356	1.757188
200	39.500942	50.132802	86.346516	50.265604
200	0.000000	0.000000	0.000000	0.000000
205	0.713554	4.707792	23.126464	4.829545

Figure 1: Google Schema

5.4.2 Azure

After initial pre-processing and cleaning, we were able to work with data from the entire *vmtable* data table for our analysis and comparisons.

The data table has $\approx$ **2.695 Million VM IDs**.

We utilized all of them for our analysis. The `max_cpu` and `avg_cpu` values for Azure, in the table are already normalized / in percentage format. The final homogeneous table, that we worked with for Azure Trace data is:

vm_id	vm_creation_timestamp	vm_deletion_timestamp
rKggHO/04j31UFy65mDTwtjdMQL/G03xWfl3xGeiilB4/W...	424500	425400
YrR8gPtBmfNaOdnNEW5lf1SdTqQgGQHEnLHGPjySt53bKW...	1133100	1133700
xzQ++JF1UAkh70CDhmzkiOo+DQn+E2TLErCFKEmSswv1pl...	0	2591400
vZEivnhabRmlmDr+JqKqZnpIM3WxtypwoxjfjnklR/idyR...	228300	229800
MqvcZ6Au5oul6if56MJHmoSqHtX8oRv0dPkaxCld3aUcr1...	1395600	1397700

max_cpu	avg_cpu	p95_max_cpu	vm_category	vm_core_count	vm_memory	start_hour	end_hour
37.879261	3.325358	37.879261	Unknown	4	32	117	118
0.304368	0.220553	0.304368	Unknown	4	32	314	314
98.573424	30.340054	98.212503	Interactive	2	4	0	719
82.581449	13.876299	82.581449	Unknown	2	4	63	63
0.097875	0.035215	0.097875	Unknown	4	32	387	388

Figure 2: Azure Schema

5.4.3 Alibaba

For Alibaba, the 2 main tables that we have worked with are *batch_instance* and *machine_usage*. The *machine_usage* table already included the normalized, percentage values for CPU and Memory utilization. However, for predictions, we utilized the *batch_instance* data table to make predictions about the number of CPU cores and Memory cores that were utilized by each instance.

The final two homogeneous tables, that we worked with, for Alibaba Trace data are:

instance_name	task_name	task_type	start_time_instance
ins_74901673	task_LTg0MTUwNTA5Mjg4MDkwNjIzMA==	10	673795
ins_815802872	M1	1	158478
ins_564677701	M1	1	372602
ins_257566161	M1	1	372602
ins_688679908	M1	1	372602

end_time_instance	machine_id	cpu_avg	cpu_max	mem_avg	mem_max
673797	m_2637	0.13	0.16	0.02	0.02
158520	m_3430	0.03	0.19	0.13	0.18
372616	m_1910	0.87	1.16	0.04	0.05
372615	m_2485	0.91	1.23	0.05	0.05
372615	m_993	0.93	1.41	0.05	0.05

Figure 3: Alibaba Schema - Batch Instance

machine_id	time_stamp	cpu_util_percent	mem_util_percent	mem_gps	mkpi	net_in	net_out	disk_io_percent
m_425	0	47	89	NaN	NaN	34.90	28.60	3
m_626	0	20	90	NaN	NaN	37.23	32.58	5
m_3089	0	7	88	NaN	NaN	29.93	20.88	1
m_111	0	18	92	NaN	NaN	39.17	32.09	3
m_796	0	24	75	NaN	NaN	41.86	37.79	5

Figure 4: Alibaba Schema - Machine Usage

6 Results, Analysis, & Evaluation

6.1 Instance Level Analysis

6.1.1 Google - Average CPU Utilization - Time Series

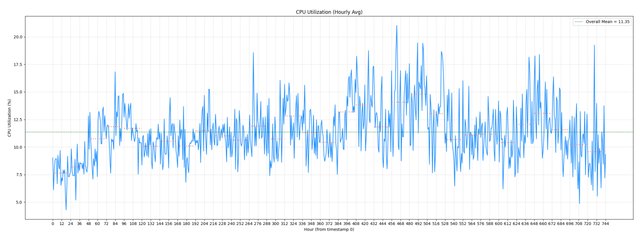

Figure 5: Average CPU Utilization over Time (Google)

The time series plot in Figure 5 shows the hourly average mean CPU utilization across Google's machines. The utilization fluctuates significantly across hours, mostly staying within the 30–50% range. There are some sharp spikes and dips and mean is at 39.63%. Overall, the workload patterns are dynamic but we see moderate utilization.

6.1.2 Azure - Average CPU Utilization - Time Series

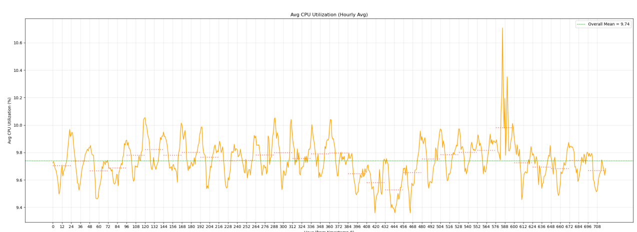

Figure 6: Average CPU Utilization over Time (Azure)

The time series plot in Figure 6 shows the hourly average mean CPU utilization across Azure's machines with an overall mean of 9.74%. The utilization is consistently low and fluctuations are around the mean with only occasional spikes. The data shows under-utilization where CPU usage rarely crosses 10%. Thus, either most workloads are lightweight or resources are largely over-provisioned. The variation in demand is also very low over time.

6.1.3 Alibaba - Average CPU Utilization - Time Series

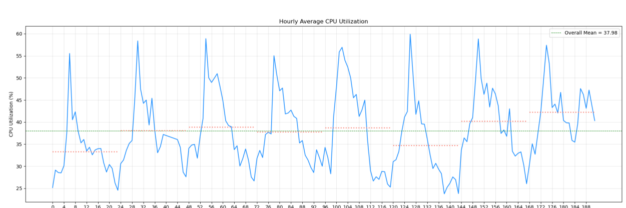

Figure 7: Average CPU Utilization over Time (Alibaba)

The time series plot in Figure 7 shows the hourly average mean CPU utilization for Alibaba's machines with an overall mean of 37.98%. We see regular sharp peaks and some drops but overall the workload behavior is very periodic. Thus, the demand patterns are predictable and usage is mostly around the mean.

6.1.4 Google - Average Memory Utilization - Time Series

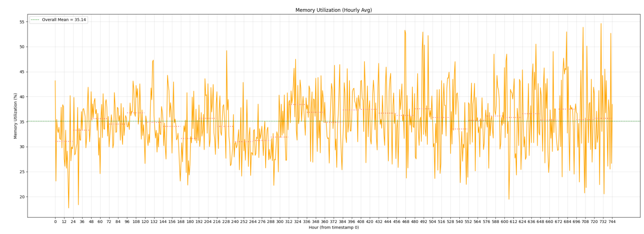

Figure 8: Average Memory Utilization over Time (Google)

The time series plot in Figure 8 shows the hourly average mean memory utilization across Google's machines with an overall mean of 36.10%. We see high variability and fluctuations between 20% and 55%. The demands are therefore inconsistent and change across different hours. It is hard to find any long-term trends and there are frequent spikes and drops which indicate that the workload is dynamic.

6.1.5 Azure - Average Memory Utilization - Time Series

Average memory utilization data was not available for the Azure Cluster data.

6.1.6 Alibaba - Average Memory Utilization - Time Series

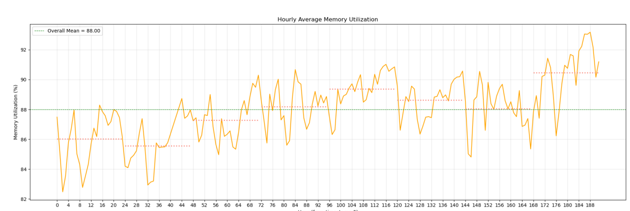

Figure 9: Average Memory Utilization over Time (Alibaba)

The time series plot in Figure 9 shows the hourly average mean memory utilization across Alibaba's machines. The plot shows that there is consistently high usage with an overall mean of 88%. There are low fluctuations but mostly between 83% to 93%. Overall, memory is a highly utilized and might be constrained in Alibaba (very optimal allocation). There is low variation which implies a stable workload.

6.1.7 Google - Max CPU Utilization - Time Series

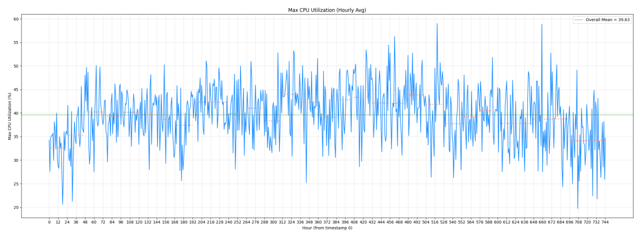

Figure 10: Max CPU Utilization over Time (Google)

The time series plot in Figure 10 shows the hourly average mean maximum CPU utilization across Google's machines with an overall mean of 39.63%. There are high fluctuations mostly ranging between 20% and 60% which shows that even during high CPU demand, the allocation was really high while the usage was moderate. Overall, the trends are stable around the mean and the workload patterns are dynamic with varied CPU usage over time.

6.1.8 Azure - Max CPU Utilization - Time Series

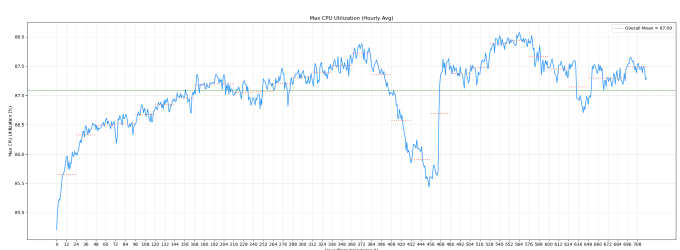

Figure 11: Max CPU Utilization over Time (Azure)

The time series plot in Figure 11 shows the hourly average mean maximum CPU utilization in Azure's cluster with an overall mean of 87.09%. We see that the utilization is initially very low and then we see high maximum utilization. There is a strong dip between hours 400 and 480 after which utilization again sharply increases. Overall, there is constantly high CPU utilization and fluctuations are very less. There is a high resource demand across most machines over time.

6.1.9 Alibaba - Max CPU Utilization - Time Series

Maximum cpu utilization data was not available for the Alibaba Cluster data.

6.1.10 Google - Max Memory Utilization - Time Series

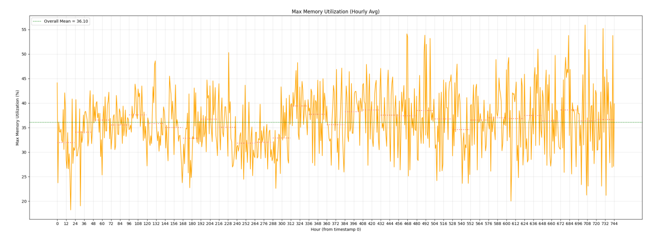

Figure 12: Max Memory Utilization over Time (Google)

The time series plot in Figure 12 shows the hourly average mean maximum memory utilization for Google's cluster with a mean of 36.10%. The memory usage has frequent random spikes and dips with utilization staying mostly between 20% to 50%. There is a lack of a strong upward or downward trend. This implies that the system stays under consistent pressure but faces variable demands with some times having much higher demand than others.

6.1.11 Azure - Max Memory Utilization - Time Series

Assuming that the allocation was optimal and all memory allocated was utilized, the memory utilization patterns for Azure are:

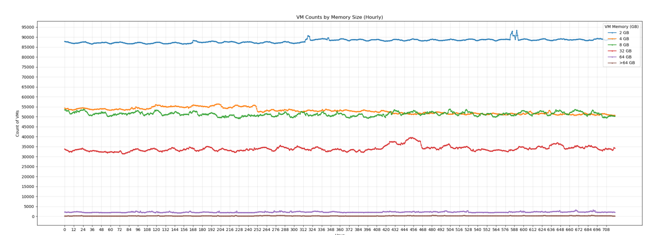

Figure 13: Max Memory Utilization over Time (Azure)

The time series plot in Figure 13 shows that Azure's memory demand is concentrated in smaller VM configurations. 2GB VMs make up the majority (around 87,000 machines) and we also see high number of VMs utilizing 4GB, 8GB, and 32GB VMs. Even with lower counts, larger VMs (64GB and >64GB) contribute heavily to the total memory usage. If we assume all allocated resources were utilized (full utilization), then the memory demands would be a lot. This might imply cost-optimized microservices or a smaller number of memory-intensive applications.

6.1.12 Alibaba - Max Memory Utilization - Time Series

Maximum memory utilization data was not available for the Alibaba Cluster data.

6.2 Machine Level Analysis

6.2.1 Google - Average CPU Utilization

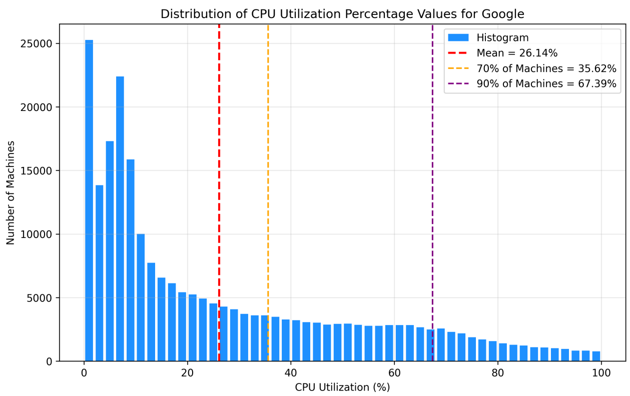

Figure 14: Average CPU Utilization (Google)

The histogram in Figure 14 shows that CPU utilization in the Google cluster is left-skewed. Most of the machines operate at very low utilization levels with majority under 20%, and a sharp peak at 0–5%. The average utilization is only 26.14%, and 70% of the machines use less than 35.62% of their CPU allocation. Even the 90th percentile utilization remains below 67.39%. This shows that there is significant underutilization across the cluster.

6.2.2 Azure - Average CPU Utilization

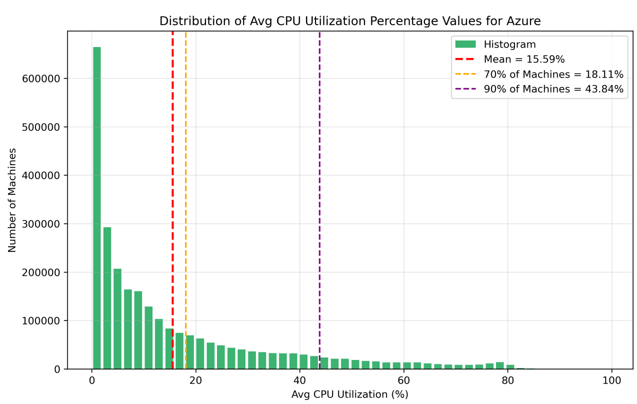

Figure 15: Average CPU Utilization (Azure)

The histogram in Figure 15 shows that CPU utilization in Azure's cluster is extremely low for most machines. The mean utilization is only at 15.59% and 70% of machines stay below 18.11%. Moreover for 90% machines usage is less than 43.84%. The distribution is highly skewed towards the left and majority machines run between 0–10% utilization. This suggests that there is significant under-utilization and a large amount of idle capacity.

6.2.3 Alibaba - Average CPU Utilization

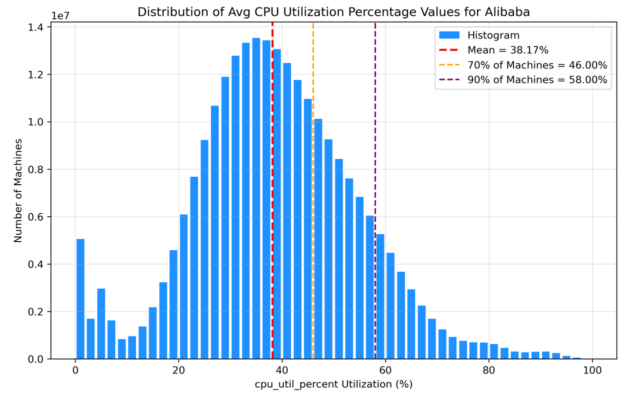

Figure 16: Average CPU Utilization (Alibaba)

The histogram in Figure 16 shows that CPU utilization in Alibaba's machines is more balanced and efficient as compared to Google and Azure. The distribution is bell-shaped and is centered around a mean of 38.17%. Most machines utilize only a moderate amount with 70% operating below 46%. Moreover, 90% of machines stay under 58%. In contrast to the heavily skewed distributions we saw in Google and Azure, Alibaba's usage is spread much more evenly with lesser idle machines. There is a peak in the mid-utilization range.

6.2.4 Google - Average Memory Utilization

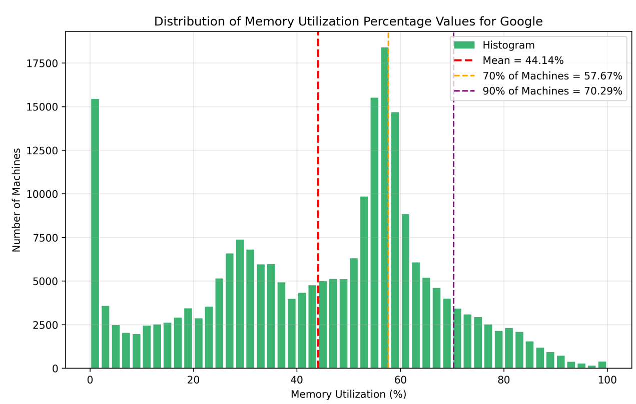

Figure 17: Average Memory Utilization (Google)

The histogram in Figure 17 shows that Memory utilization across Google's machines is consistent. Like CPU, there's a spike at very low utilization (near 0-5%) but for most machines memory usage is moderate. The average usage is 44% and about 30% of machines have upto 57.67% usage of their allocated memory. About 10% of machines even go beyond 70.29% of memory usage of the allocated. Most machines stay in the 55–60% range. Therefore, memory utilization is moderate.

6.2.5 Azure - Average Memory Utilization

Average memory utilization data was not available for the Azure Cluster data.

6.2.6 Alibaba - Average Memory Utilization

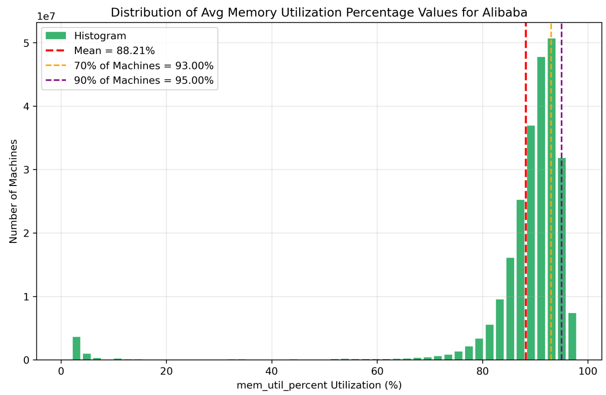

Figure 18: Average Memory Utilization (Alibaba)

The histogram in Figure 18 shows that the Memory utilization in Alibaba's clusters is extremely high and concentrated. The average memory usage is 88.21%. About 30% of machines use upto 93% of memory. Moreover, 10% go upto 95% utilization. The distribution is right-skewed and majority of machines operate between 85% and 100% utilization. This shows that Alibaba's memory resources are used efficiently and consistently.

6.3 Task Type Analysis

6.3.1 Google - Average CPU & Memory Utilization by Task Type

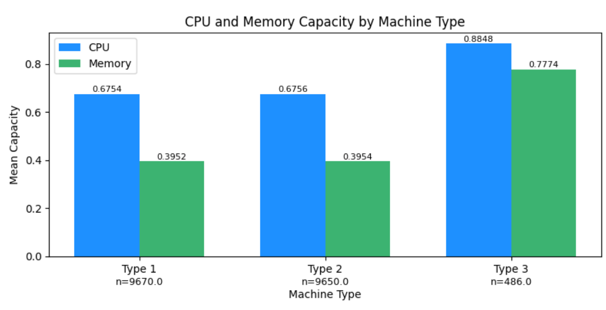

Figure 19: Average CPU & Memory Utilization by Task Type (Google)

6.3.2 Azure - Average CPU & Memory Utilization by Task Type

Average CPU & Memory utilization data by task type was not available for the Azure Cluster data.

6.3.3 Alibaba - Average CPU & Memory Utilization by Task Type

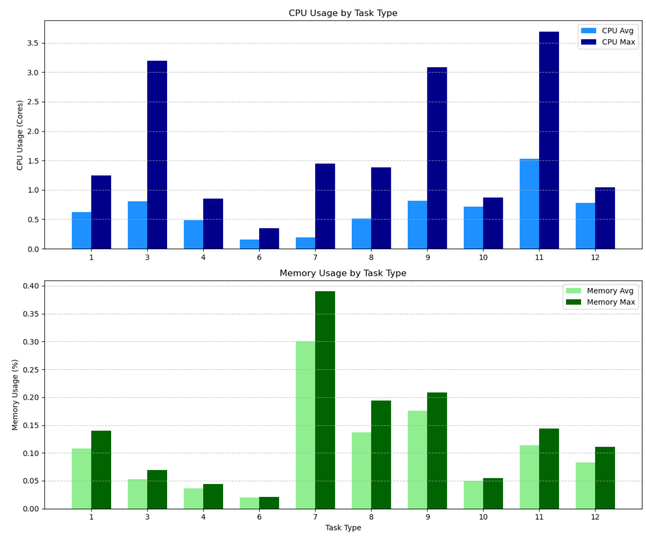

Figure 20: Average CPU & Memory Utilization by Task Type (Alibaba)

As we can see, CPU and Memory utilization rates change a lot based on the Task Type. We can see from these graphs that there can be several generalizations that can be drawn for CPU and Memory utilization based on just the Task Type. With the help of this, we inferred that `Task Type` will be an extremely helpful and valuable parameter during the training of our predictive Machine Learning Model. Therefore, we have used it as a parameter in our XGBoost model, for predicting both CPU and Memory utilization for Alibaba and Google.

6.4 Predictive Modeling / Analysis

The main goal of performing predictive modeling, on the three cloud traces was to find a way to efficiently predict the CPU and Memory utilization based on the different parameters present in each dataset.

The Machine Learning model which we used for training / prediction purposes was the `XGBoost Regression` Model, having `1500 Decision Trees`. Also, from each of the datasets, we used 95% of the data as the Training data, and the remaining 5% of the data as the Test data.

Then, we have computed the Root Mean Squared Error (RMSE) of the XGBoost model, which have trained. Also, we have constructed a custom accuracy metric, such that if the prediction for a case, is in the ± 10 range, then the prediction is considered to be accurate. Based on that, we have computed the percentage of cases, for which the prediction is accurate for both CPU and Memory utilization across the three cloud providers.

6.4.1 Google

Training Variables Used

- Start Hour [0-23]
- Total Running Time
- Task Type
- Priority
- Scheduling Class

CPU Utilization Pattern Prediction

- RMSE ≈ 9.2035
- Accuracy ≈ 81.8068%

Memory Utilization Pattern Prediction

- RMSE ≈ 15.8080
- Accuracy ≈ 67.2593%

Thus, for Google, we can easily decipher that the prediction accuracy is really high, and RMSE is low, for CPU utilization. However, for Memory Utilization, the prediction accuracy is relatively low, and RMSE is relatively higher, which implies that the predictions are much worse.

6.4.2 Azure

Training Variables Used

- Start Hour [0-23]
- Total Running Time
- VM Core Count
- VM Memory
- VM Category

CPU Utilization Pattern Prediction

- RMSE ≈ 14.8718
- Accuracy ≈ 65.3044%

For Azure, the prediction accuracy for CPU Utilization, from the XGBoost model, is slightly sub par, and the RMSE is also quite average, which indicates that the variables which we are using for training don't help that much, relatively, in helping make a deterministic choice on the CPU utilization for a particular task.

6.4.3 Alibaba

Training Variables Used

- Start Hour [0-23]
- Total Running Time
- Task Type

CPU Utilization Pattern Prediction

- RMSE ≈ 23.3734
- Accuracy ≈ 52.5361%

Memory Utilization Pattern Prediction

- RMSE ≈ 18.4607
- Accuracy ≈ 87.7181%

For Alibaba, the XGBoost model is able to extremely accurately predict the memory utilization patterns, even thought the RMSE is on the higher end. However, the accuracy for memory utilization is quite high, which is extremely useful. However, the bottleneck of the prediction algorithm is the CPU Utilization prediction, as it leads to the worst accuracy overall. This is probably because there aren't enough parameters / variables available for efficiently training the model in Alibaba's dataset, as compared to the other cloud providers, which leads to the worst CPU utilization accuracy.

6.5 Exploratory Data Analysis

We conducted exploratory data analysis on the different cloud provider datasets. Here are some plots that are the most insightful and relevant. These highlight some key trends and behaviors observed. The variables in these plots play a significant role in datacenter operations and therefore we utilized these variables in our predictive modeling as well. Furthermore, our analysis was also guided by the insights we gained from them.

6.5.1 Alibaba - Tasks Running over Time

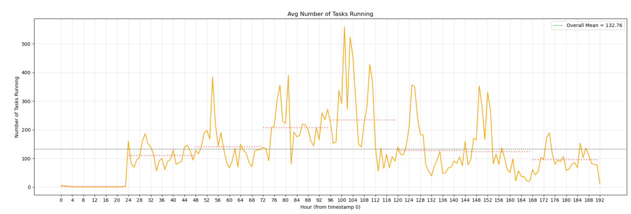

Figure 21: Number of Tasks Running over Time (Alibaba)

6.5.2 Azure - VM Count Distribution over Time

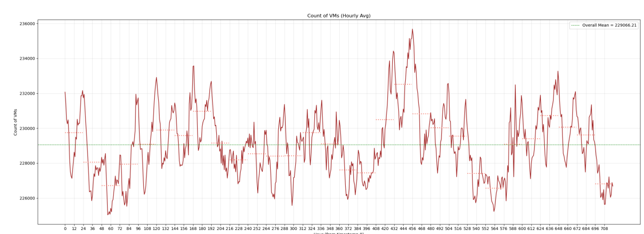

Figure 22: VM Count Distribution over Time (Azure)

6.5.3 Azure - VM Core Count Distribution over Time

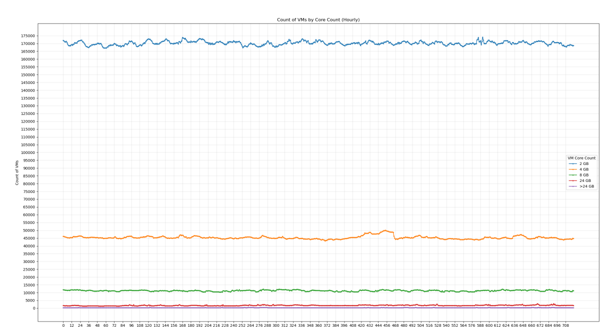

Figure 23: VM Core Count Distribution over Time (Azure)

6.5.4 Azure - VM Category Count Distribution over Time

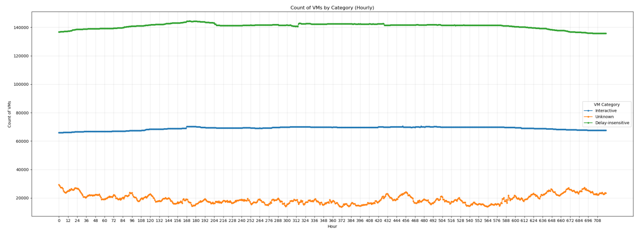

Figure 24: VM Category Count Distribution over Time (Azure)

7 Conclusions

7.1 Summary of Cloud Utilization Behaviors

Every cloud provider has distinct strategies and workload behaviors. Based on our analysis of the resource utilization patterns, we conclude that:

Google has moderate utilization with high fluctuations and varied demands over time. The average CPU utilization is 26.14% and average memory utilization is 44%. This implies that there is under-utilization and room for improvement in optimizing resource allocation strategies.

Azure, on the contrary has a really low average CPU utilization at 15.59% and a really high max CPU utilization at 87.09%. This suggests that most VMs have light demands but the system still has to sometimes accommodate high demands (occasional spikes). This might be the reason why we see high over-provisioning and low utilization.

Alibaba shows the most balanced and efficient patterns. The average CPU utilization is 38.17%. The trends are periodic and the memory utilization remains high at 88.21%. This implies that minimum resources are wasted and the resource packing is very efficient.

7.2 Challenging Common Assumptions About Cloud Datacenters

Through our analysis, we challenge the different assumptions that exist about cloud datacenter behavior. Most of these arise from examining a single provider in isolation.

It's often assumed that CPU utilization is low across providers - we find that this is true for Azure (15.59%) and Google (26.14%) but we see how Alibaba (38.17%) shows significantly higher usage. Also, the variation is a lot between all the three.

Memory is not always underutilized - Google averages 44% but we see that Alibaba maintains a high utilization 88.21% which highlights the difference in optimal allocation across providers.

There is an incorrect belief that workloads are mostly idle with rare spikes. We see that Google's workloads are random and bursty but Azure on the other hand shows flat and idle patterns with Alibaba's following a periodic pattern.

Furthermore, not all providers use centralized scheduling. Google uses Borg while Azure relies on user-managed deployments. Alibaba adopts a dynamic, adaptive approach.

Even cloud workloads do not show uniform behavior across the different providers. We see unique patterns for each depending on their infrastructures and diverse user demands.

Overall, these findings emphasize how important it is to conduct a multi-provider analysis to avoid drawing misleading conclusions about datacenter behavior. We hope that this project encourages future work on datacenter optimization.

7.3 Predictions

7.3.1 CPU Utilization

For CPU utilization, the best predictions and least RMSE was achieved for Google's Trace Data, 81.8068%. The worst predictions and highest RMSE was achieved for Alibaba's Trace Data 52.5361%. This is mostly because of the robust nature of the parameters and variables available for Google, like Task Type, Priority, Scheduling Class etc. Variables such as Scheduling Class and Priority aren't available for aren't available in Alibaba's trace data, which makes the prediction algorithm worse.

Therefore, based on our predictive analysis, we can say that the CPU utilization patterns can be judged and predicted in a reasonable manner, in the following order:

$$\text{Alibaba} < \text{Azure} < \text{Google}$$

7.3.2 Memory Utilization

For memory utilization, we don't have data available for Azure. However, for Alibaba, accuracy is 87.7181% and RMSE is much better as compared to Google's prediction accuracy of 67.2593% for memory utilization. Even though Alibaba, has a lesser number of variables and parameters that are used in model training, Alibaba's Memory utilization is more predictable. This implies that it's more generalizable and predictable.

Therefore, based on our predictive analysis, the Memory utilization patterns for the Alibaba and Google Cloud can be generalized / predicted, in the order:

$$\text{Google} < \text{Alibaba}$$

8 Responsibilities

Akshat Karwa and Mehul Rastogi jointly contributed to all the aspects of the project including - understanding the datasets, writing code, performing analysis, generating visualizations, and compiling the final report. Both were equally involved in the experimentation, the interpretation of results, and the documentation.

9 Bibliography

References

[1] Reiss, C., Tumanov, A., Ganger, G. R., Katz, R. H., Kozuch, M. A., Intel Science and Technology Center for Cloud Computing, & Carnegie Mellon University. (2012). Towards understanding heterogeneous clouds at scale: Google trace analysis (Report ISTC-CC-TR-12-101). Intel Science and Technology Center for Cloud Computing. https://www.pdl.cmu.edu/ftp/CloudComputing/ISTC-CC-TR-12-101.pdf

[2] Cortez, E., Bonde, A., Microsoft, Muzio, A., Russinovich, M., Fontoura, M., & Bianchini, R. (2017). Resource Central: Understanding and predicting workloads for improved resource management in large cloud platforms. In *Proceedings of SOSP'17*. ACM. https://www.microsoft.com/en-us/research/wp-content/uploads/2017/10/Resource-Central-SOSP17.pdf

[3] Lu, C., Ye, K., Xu, G., Xu, C.-Z., & Bai, T. (2017). Imbalance in the cloud: An analysis on Alibaba cluster trace. In *Proceedings of the IEEE Conference*. IEEE. https://ieeexplore.ieee.org/document/8258257

[4] Google. (n.d.). Google cluster-usage traces v3. Google Docs. https://drive.google.com/file/d/10r6cnJ5cJ89fPWCgj7j4LtLBqYN9RiI9/view

[5] Google. (n.d.). GitHub - google/cluster-data: Borg cluster traces from Google. GitHub. https://github.com/google/cluster-data

[6] Alibaba. (n.d.). clusterdata/cluster-trace-v2018/schema.txt at master · alibaba/clusterdata. GitHub. https://github.com/alibaba/clusterdata/blob/master/cluster-trace-v2018/schema.txt

[7] Alibaba. (n.d.-b). clusterdata/cluster-trace-v2018/trace_2018.md at master · alibaba/clusterdata. GitHub. https://github.com/alibaba/clusterdata/blob/master/cluster-trace-v2018/trace_2018.md

[8] Azure. (n.d.-a). AzurePublicDataset/AzurePublicDatasetLinksV2.txt at master · Azure/AzurePublicDataset. GitHub. https://github.com/Azure/AzurePublicDataset/blob/master/AzurePublicDatasetLinksV2.txt

[9] Azure. (n.d.). GitHub - Azure/AzurePublicDataSet: Microsoft Azure Traces. GitHub. https://github.com/Azure/AzurePublicDataset