Exploring Characteristics of Inter-cluster Machines and Cloud Applications on Google Clusters

Yuhui Lin
University of St Andrews
St Andrews, UK
Email: yl205@st-andrews.ac.uk

Adam Barker
University of St Andrews
St Andrews, UK
Email: adam.barker@st-andrews.ac.uk

Sheriffo Ceesay
University of St Andrews
St Andrews, UK
Email: sc306@st-andrews.ac.uk

***Abstract*—Modern cluster management systems have been evolving to cope with running and managing diverse cloud applications on heterogeneous computing clusters. Consequently, the system behaviours become complex and non-trivial to explain. In this paper we take the recently published Google trace data set version 3 (V3) as a case study to explore various aspects of inter-cluster differences. We analyse the distribution of underlying physical machines resource, e.g. number and types of machine, and metrics of computational job requests, e.g. job duration, utilisation and Cycles Per Instruction (CPI). We also apply an unsupervised learning algorithm on the metrics to characterise jobs. Our analysis suggests that the composition of the underlying machine resources in different cells can be substantially different, and the cells with similar machine resource structures can utilise resources differently depending on the characteristics of job requests.**

***Index Terms*—Cloud Computing, Google Cloud Traces, Cloud Application Characteristics**

I. Introduction

In the era of big data and cloud computing, data centres host an enormous number of applications with a diverse range of characteristics on heterogeneous clusters. Modern cluster management systems have been evolving to keep up with the increased usage, whilst at the same time trying to improve utilisation of the underlying machine resource. Consequently, cluster management systems are becoming highly complex entities which have to satisfy competing requirements. Furthermore, within an academic environment, it is challenging to understand the decisions that schedulers make, as well as the underlying jobs and their real-world operational demands. To shed light on the complex system behaviours, trace analysis has been playing an intensive role. Cloud infrastructure operators published their cluster trace data to encourage research to tackle the challenges, Some examples of the data sets are: Google [1], [2], Microsoft [3] and Alibaba [4].

Recently, Google published a new trace data at a large scale [5]. Comparing to the previous version [6], the data set includes 8 different Google computing clusters, called *cells*, for the month of May 2019. It enables to conduct inter-cluster analysis which was not possible in the previous studies [7]–[11].

978-1-7281-6251-5/20/$31.00 ©2020 IEEE

Inspired by the first study of the new trace data [12], which covered the analysis of overall utilisation and job characteristics, our study explores inter-cluster differences by analysing underlying machine resources and characterises of the jobs. The contributions of our study are as follows:

- Exploring the underlying machines in different cells. The composition of underlying machine resources in different cells can be substantially different, and the cells with a similar composition can utilise the resources differently.
- Analysing various metrics of jobs, including duration, task numbers, average CPU/memory utilisation, maximum utilisation, page cache, Cycles Per Instruction (CPI) and Memory Accesses Per Instruction (MAPI). Duration, CPU utilisation and CPI, are the most distinctive metrics because the rest of metrics are strongly correlated to them.
- Applying an unsupervised learning algorithm on the most distinctive metrics to categorise jobs. We identify a pattern of the resulting clustering shapes, which are concave and convex. The distribution of the job in the concave shapes lack in variance in one or more dimensions and the convex shape has relatively even variance scale in all dimensions.

The rest of this paper is organised as follows: §II gives the background of Google computing clusters as well as the new trace data set. §III presents an overview of our approach to processing and analysing the data. The details of results and analysis are presented in §IV, §V and §VI. We summarise our findings in§VII, followed by related work in §VIII. We conclude our study and discuss future work in §IX.

II. Google cloud trace data set

Borg [13] is the cluster management system in Google. Each cluster has a Borg deployment, which comprises a logically centralised cluster scheduler master and a large number of underlying machines. Such deployment is called a *cell*. There are two types of resource requests for Borg cells: *jobs* and *alloc*. Jobs specify computations or workloads that users want to run; while allocs are optional requests to allow users to reserve resources for jobs to run. Each job can have multiple *tasks*. Each task is a replicate of the job. The tasks from a job share the same machine resources that are allocated to the job. Jobs can also specify additional information such

as priority, computation latency-sensitive level and resources. Borg will then schedule tasks, according to the constraints and properties inherited from the job, to run on suitable machines with sufficient resources available in a cluster.

The Google trace 2019 [5] contains data such as jobs and machine events and usage detail. Jobs and machine events include the information for jobs and alloc scheduling events and the underlying machines life-cycle operations.

Usage data contain normalised CPU and memory utilisation information for each task, where the range of the normalised utilisation is (0, 1). It is calculated by dividing the actual usage using the maximum *Google Compute Units* (GCUs) or memory capacity of all machine in a cell, i.e.

$$util_{norm} = util_{actual}/max\_resource$$

In addition to utilisation, usage data also have low-level performance metrics, such as Cycles Per Instruction (*CPI*)[1], Memory Accesses Per Instruction (*MAPI*) and *page cache* .

The data set is available in Google *BigQuery* [14] which is a storage and analysis tools with a dialect of SQL. It is also available as a downloadable JSON file. Compared to the 2011 Google trace [5], [12], one of the new features of this data set is that, it includes traces of 8 cells, namely a - g, for the same period of time. In this paper, we focus on analysing inter-cluster utilisation between difference cells/clusters.

The trace data for each cell are stored independently in a set of four tables: $collection\_events$, $instance\_events$, $machine\_events$ and $instance\_usage$. The event tables contain scheduling information about jobs, tasks and underlying physical machine operations; while the usage table keeps records of utilisation data for tasks. Each job is uniquely identified by a $collection\_id$, and tasks are distinguished by an *index*. Each utilisation record contain statistical aggregated data for one task for the period of up to 3 minutes. That is, each record can be is uniquely identified by $collection\_id$, $instance\_index$ and $start\_time$.

III. Approach

As reported in [12], there is substantial variation in the utilisation between cells on the 15th day. Our study aims to explore possible factors that lead to a variance of inter-cluster resource utilisation. Due to the size of trace data, we choose 4 cells out of 8 at different data centre locations, where e is at Helsinki in Europe, f at Chicago in America, g at Singapore in Asia and h at Brussels in Europe [5].

We focus on collecting and comparing the inter-cluster differences from two aspects of scheduling, which are scheduling resources and scheduling targets (jobs). Figure 1 gives an overview of the data processing and analysis conducted for this paper. Details of the BigQuery and Python code used in this study are available in the paper resource *IPython* notebook [15].

Inter-cluster machine resources: We count and compare the total numbers of machine resources available in each cell, as well as their CPU and memory capacity. We also calculate the CPU and memory utilisation for each job and compare the distribution. More details are discussed in §IV.

Job metrics for application characteristics: For each job, we collect typical metrics, such as average utilisation and duration, as used in previous characteristics analysis [11]. In addition to utilisation data, we are also interested in 'low-level' usage metrics, e.g. CPI (cycles per instruction), as it indicates the type of instructions the job is running if the underlying machine is the same or similar. Table I shows a list of all the metrics for each job.

Except for *duration*, *task number* and *priority tier*, the rest of the metrics are related to time series data. It would be ideal to analyse time-series properties directly, however, because of the large data set, i.e. millions of jobs, we only take the average value of time series in this study. The average is helpful to give a general impression of the overall job characteristics, but it is not sufficient to conclude the causal correlation of utilisation. We will leave this as future work and discuss possible research direction in §IX.

TABLE I: Job Metrics

Metrics	Description
$duration$	Job duration is defined as the time difference between the maximum time stamp and the minimum one
$task\_num$	Total number of tasks for each job
cpu_usage	Average total CPU usage consumed by all parallel tasks
$memory\_usage$	Average total memory consumed by all parallel tasks
max_cpus	The CPU utilisation peak of tasks in a job
max_mem	The memory utilisation peak of tasks in a job
$page\_cache$	Average memory usage for the instance's file page cache
cpi	Cycles Per Instruction used to measure performance of process, calculated by averaging CPI of all tasks
$mapi$	Memory Accesses Per Instruction, calculated by averaging MAPI of all tasks

In addition to analysing distributions of each metrics, we choose three metrics, i.e. $duration$, cpu_usage and cpi to explore their joint distribution. The motivation, here, is that, these three metrics can be considered as different computation characteristics. Also, we apply an unsupervised learning algorithm on the metrics to explore job characteristics from correlation, as shown in Figure 1. In §V, we will present the details of metrics and the distribution of different cells.

IV. Inter-cluster resource analysis

We start our analysis with the underlying machine resources for each cell. Table II gives an overview of all the machine resources available. The numbers of machines are evenly distributed in cell e, f, g and h. with *mean* of 12678 and range of *mean* $\pm$ 11%. Similarly, the CPU and memory ratios are very similar. From this top-level information, f and h appear to have near-identical underlying machine resources structures.

[1]CPI measures one aspect of CPU performance by counting the average number of clock cycles per instruction for job.

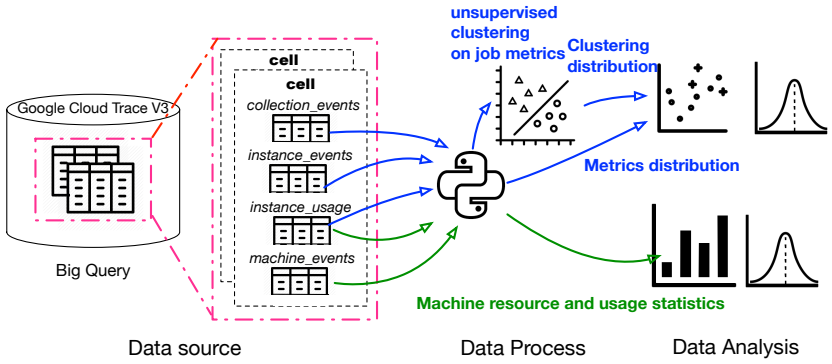

Fig. 1: An overview of the approach to data process and trace analysis in our Google trace study. The blue arrows correpond to collect and analyse metrics for jobs, and the green arrows relate to analysis of machine resources in cells.

TABLE II: Summary of machine count and resource capacity

Cell	Machine Count	Total CPU	Total Mem	CPU& Mem Ratio	Avg Resource per Machine
e	14122	7266	4314	59%	(0.51, 0.20)
f	12201	10994	5670	51%	(0.90, 0.46)
g	12796	7963	4485	56%	(0.62, 0.35)
h	11592	10317	5489	53%	(0.89, 0.47)

Note: The machine resource capacity are normalised (see in §II) to between 0 and 1. All cells have similar total machine numbers and CPU & memory ratio. In particular, f and h appear to have near-identical underlying machine structure.

Figure 2 is used to investigate individual CPU and memory capacity for each machine. The joint density plot confirms the similarity between f and h. That is, f and h have relatively similar homogeneous machine resources structure with extremely skewed distribution, i.e. high CPU with medium memory. The machine resource distribution is more diverse for e and g. The majority of machines in e are in small size for both CPU and memory. g has the most heterogeneous machine resources, where the CPU resources are evenly distributed into the categories of low, medium and high.

We then analyse the overall resource utilisation during the whole trace period for the target cells. Table III shows a summary of the statistical results aggregated for each cell. There are two main perspectives of the utilisation of data: the total resources consumed and the average utilisation for each machine. One example case to distinguish the two perspectives is when all-powerful machines experience a higher level of utilisation whiles low-spec machines are under-utilised.

TABLE III: Total resource utilisation and per-machine utilisation

		Total resource utilisation (in %)				Per-machine utilisation (in %)			
	cell	AVG	Q1	Q2	Q3	AVG	Q1	Q2	Q3
CPU	e	55%	48%	56%	62%	56%	48%	57%	64%
	f	55%	52%	55%	58%	57%	53%	56%	60%
	g	63%	60%	65%	68%	67%	64%	69%	72%
	h	58%	50%	59%	68%	59%	50%	59%	68%
Memory	e	52%	50%	52%	54%	54%	51%	53%	56%
	f	56%	54%	56%	58%	58%	58%	60%	70%
	g	51%	47%	51%	54%	56%	52%	56%	59%
	h	56%	52%	56%	59%	57%	54%	57%	60%

Overall, the utilisation rates of both perspectives are consistent for all cells, which suggests machines were evenly utilised regardless of resource specifications. We notice that g, which has the most the heterogeneous machine resource structure, has the best utilisation for both perspectives. It is also interesting to observe that Q3 (75th percentile) of h is notable higher (around 10%) than f, which indicates that h was more likely to

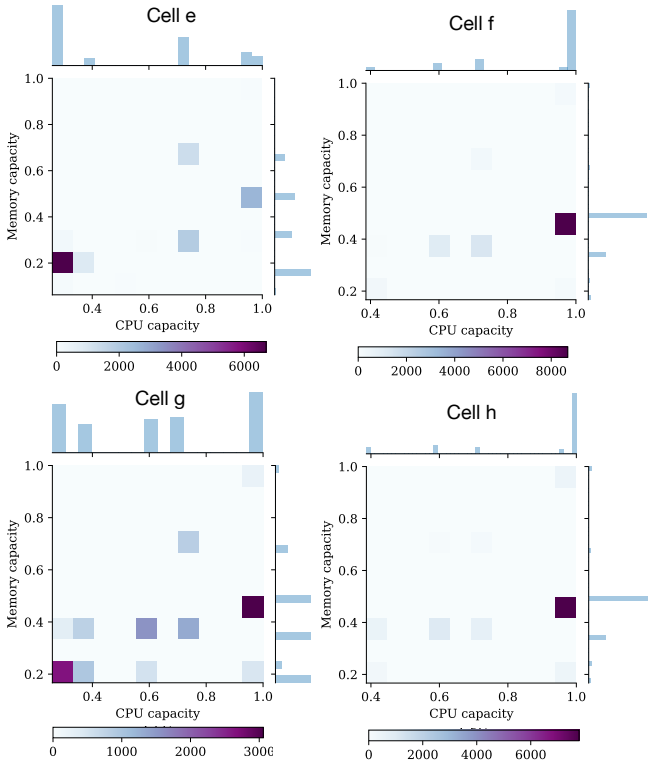

Fig. 2: Density plot showing the joint distribution of CPU and memory capacity. It suggests that f and h have relatively similar homogeneous machine resources structure with extremely skewed distribution; Most of machines in e have small size for both CPU and memory; and g has the most heterogeneous machine resource structure, where the CPU resources are evenly distributed into the categories of low, medium and high.

get higher utilisation than f although the underlying machine resources are very similar.

We have analysed the machine resources and resource utilisation in each cell. In the next section, we will analyse the differences in term of job metrics.

V. Inter-cluster job metrics

We take the process as shown in Figure 1 to collect metrics from Table I. For the full details of the metrics, we refer to the paper's Python notebook [15].

The first thing we notice is that the total number of jobs are not evenly distributed for all the cells [2]:

$$\{e : 1260650,\ f : 1263400,\ g : 967433,\ h : 3234502\}$$

The job requests in h (48% of the total job count) are 2.5 times the number in f (19%) although they have very similar machine resources structure. On the other hand, g, which is the cell with the most diverse machine resources, has the least number of jobs (14%).

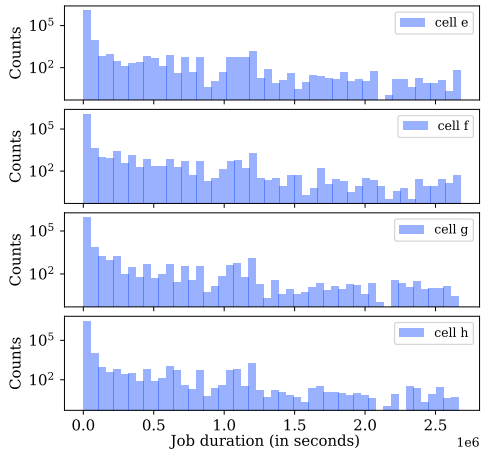

Fig. 3: Histograms showing the distribution of job duration.The duration distribution of all cells are heavily right-skewed with extreme outliers, i.e. $(\sigma/\mu)_{dur}$ = {e: 10.1, f: 9.7, g: 9.3, h: 14.0} and $(99\%/max)\%_{dur}$ = {e: 3.2%, f: 6.3%, g: 3.8%, h: 1.0%}. All cells have jobs that kept running during the period of the whole trace data. Note that y-axis is in log-scale. The x-axis is in seconds, and 2.592e+6 in seconds is 30 days.

For job duration, all cells have jobs that kept running through the whole period of trace data, and the distribution, in general, are all heavily right-skewed at different level of scale, as shown in Figure 3. Here, we use *normalised std* ($(\sigma/\mu)_{dur}$) and the ratio of 99% over the maximum duration ($(99\%/max)\%_{dur}$) to compare the scale of spreading and the outlier, respectively.

[2]Note that each cell is a cluster from different data centre. Thus, there is no global queue or meta-scheduler between them.

Similarly, CPU and memory utilisation are also heavily right-skewed with much more extreme outliers as shown in Figure 5. The level of skewness is similar for each cell, with h being most skewed with the most extreme outliers while e is the least.

To further compare the utilisation in different cells, we also plot weight *Cumulative Distribution Function* (CDF) for both CPU and memory, as shown in Figure 4. Instead of using counts for aggregation, we use the value of utilisation and the value of utilisation times duration as weights, respectively. The former is to show the utilisation rate while the latter is to show the overall consumption. We can observe that, although the majority of jobs consume little resources , e.g. μ is less than 0.6 for all cells, the substantial increase in utilisation for both CPU and memory are actually in the range of (10, 100) for CPU and (1, 100) for memory. There are mainly two increase patterns: near-linear at the log scale of x, e.g. CPU weight CDF with utilisation for cell h; and sudden jump, e.g. CPU weight CDF with utilisation for cell e.

VI. Unsupervised learning on metrics

We would like to apply unsupervised learning algorithms on the metrics to explore categories of jobs. The challenge is that the numbers of data points can be millions and each data point is multiple dimensional. Given the size of the data, each dimension increases the computation cost significantly, due to the curse of dimensionality. Moreover, the heavily skewed data with extreme outliers would bias the clustering algorithm. To tackle these problems, we first reduce the number of the metrics to 3 based on correlation analysis and then apply a non-linear power transformation, i.e. *Box-Cox*, to scale the range of outliers, followed standard scaler to centre data and remove the impact of standards deviation. For the choice of unsupervised learning algorithm, we choose Density-based spatial clustering of applications with noise (DBSCAN) [16], because of the computational efficiency, scalability of data size and the ability of handle the uneven size of clusters. In the rest of this subsection, we will present details of each clustering decision choices, as well as the clustering results.

Choice of metrics for clustering: Figure 7 shows the correlation of all the metrics. We can observe that metrics such as $task\_num$, $memory\_usage$, max_cpus, max_mem and $page\_cache$ are strongly correlated to cpu_usage. This observation follows the intuition that changes of computation requirement typically occurs at the same time with changes of required memory and number of parallelisation. Similarly, $mapi$ is strongly correlated to cpi. We also observe that not all the job have low-level measurement data such as cpi and $mapi$, i.e. only 28%. However, those jobs account for a major portion of the resource usage, as shown in Figure 7. Therefore, we choose $duration$, cpu_usage and cpi as metrics for clustering. They account for different computation aspects of a job, where $duration$ relates to the time aspect, cpu_usage shows how busy the underlying machine when running the job, and cpi can suggest the types of computation hinted from cycles of instructions. Moreover, including cpi

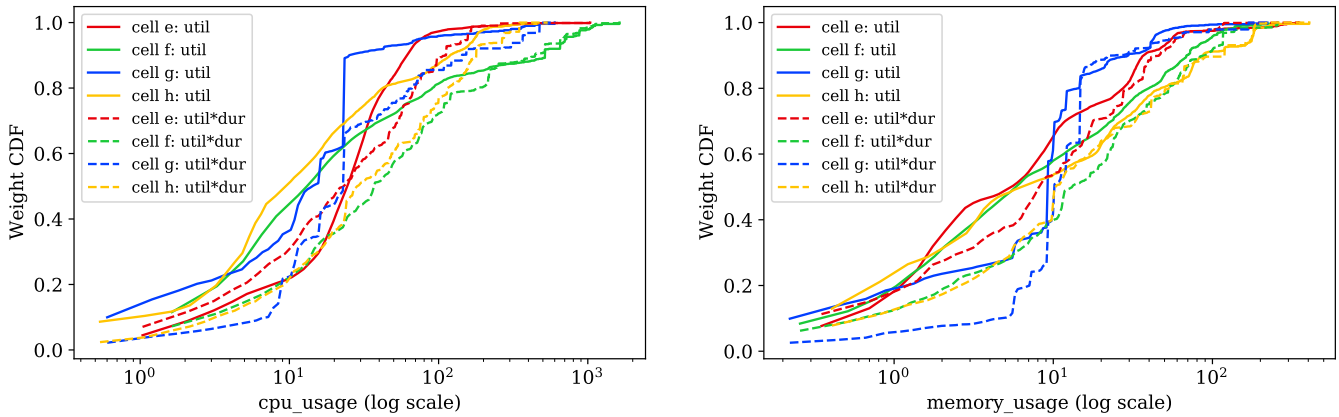

Fig. 4: Weighted CDF for CPU / Memory utilisation. Each weighted CDF is calculated by multiplying counts with the value of utilisation and the value of utilisation times duration, respectively. The former one is to show the resource consumption rate, and the latter is to show the overall consumption.

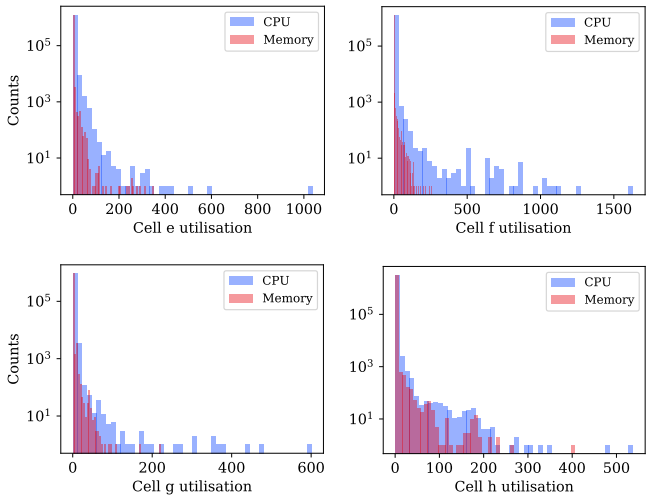

Fig. 5: CPU and memory utilisation distribution in each cell. Similar to job duration, the distrubutions are also heavily right-skewed with much more extreme outliers, i.e. $(\sigma/\mu)_{cpu}$ = {e: 7.6, f: 19.4, g: 15.3, h: 20.4}, $(99\%/max)\%_{cpu}$ = {e: 1.9%, f: 0.4%, g: 0.3%, h: 0.2%}, $(\sigma/\mu)_{mem}$ = {e: 10.7, f: 14.2, g: 11.8, h: 30.2} and $(99\%/max)\%_{mem}$ = {e: 0.7%, f: 0.7%, g: 0.4%, h: 0.1%}.

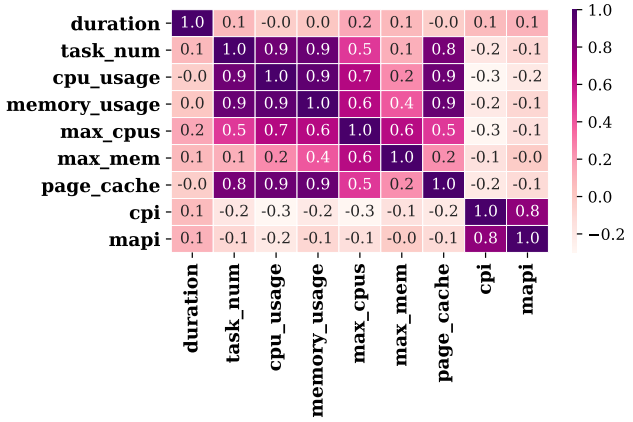

Fig. 6: Heatmap showing correlation of metrics. $task\_num$, $memory\_usage$, max_cpus, max_mem and $page\_cache$ are strongly correlated to cpu_usage. Same for cpi and $mapi$.

have the advantage of significantly reducing the size of data points.

Data scaling and normalisation: The raw metrics data are heavily right-skewed with extreme outliers (Figure 8 left and middle columns). We do not want to exclude those outliers from clustering because they can have an important impact on resource computation. Here, we first apply a power transformation, called *Box-Cox*, to scale the range in a non-linear way:

$$y_i^{(\lambda)} = \begin{cases} \frac{y_i^{\lambda}-1}{\lambda} & \text{if } \lambda \neq 0 \\ \ln y_i & \text{if } \lambda = 0 \end{cases}$$

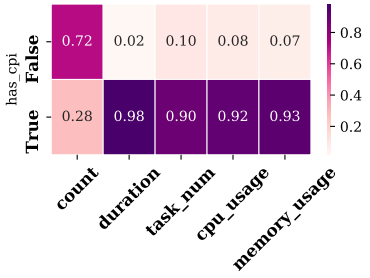

Fig. 7: Heatmap showing aggregated usage of jobs categorised of whether a job has cpi data or not. 28% of jobs with CPI data comprise majority of resource utilisation.

TABLE IV: Statistics each clustering and cell ratio

Cluster	Job Cnt	Job Cnt (in %) for Each Cell			
		e	f	g	h
0	6830	27%	28%	18%	28%
1	3626	25%	30%	19%	26%
2	3038	26%	28%	15%	32%
3	3636	18%	50%	12%	21%
4	3467	4%	65%	21%	9%
5	3619	11%	8%	73%	7%
6	17149	38%	7%	6%	48%
7	15175	85%	6%	3%	6%
8	4242	47%	47%	4%	3%
9	2864	29%	9%	7%	55%
10	35752	25%	25%	25%	25%
11	9171	67%	20%	4%	9%
12	2641	29%	12%	2%	58%
13	2840	6%	5%	0%	88%
14	2830	17%	7%	2%	75%
15	16430	0%	1%	0%	99%
16	5506	2%	3%	3%	92%
17	8299	82%	12%	3%	3%
18	2607	15%	13%	6%	65%
19	5496	6%	10%	65%	19%
20	12998	63%	2%	33%	3%
21	6146	23%	30%	13%	34%
22	7534	64%	4%	3%	28%
23	3702	8%	6%	4%	82%
24	65086	28%	2%	5%	65%
25	24941	18%	29%	16%	37%
26	2216	36%	47%	14%	2%
27	5054	35%	53%	10%	1%
28	3694	16%	7%	75%	2%
29	2238	13%	3%	81%	2%
30	24472	43%	41%	12%	3%
31	24870	6%	21%	17%	56%
32	2899	4%	11%	7%	78%
33	30181	14%	64%	18%	3%
34	3440	26%	33%	34%	7%
35	2028	37%	48%	4%	11%
36	633304	37%	17%	32%	14%
-1	862869	29%	25%	18%	28%

where $\lambda_{duration} = 0.5$, $\lambda_{cpu} = 0.25$, $\lambda_{cpi} = 0.25$. We then apply the standard scaler to centre the data as well as making them unit variance. Figure 8 (right column) shows the distributions after the pre-processing where the range has been significantly reduced with the preservation of the basic distribution shapes.

Clustering on metrics: We run DBSCAN to cluster jobs using $duration$, cpu_usage and cpi as features. By providing the minimum cluster size, i.e. $min\_cluster\_size = 2000$, DBSCAN can automatically optimise the number of clusters. $min\_cluster\_size$ is decided by checking the cluster sizes of the optimal number of clusters between (2, 50) by running *K-Means*.

DBSCAN categorises the jobs as 38 clusters. The results are plotted as 3D scatter diagrams in Figure 9 and Figure 10.

The job clusters become 3D dimensional objects in each plot. Here, the potential benefit is to suggest the likelihood of values for a dimension with given values from other dimensions, e.g. predicting values of CPU usage from given values of duration and CPI.

There are mainly two types of shape: concave and convex. The clusters in the concave shapes lack variances in one or more dimensions. In extreme cases, the shape becomes planes, e.g. clusters 0 - 4, or lines, e.g. cluster -1. Also, cluster -1 reveals an interesting pattern that a job with extreme values in one dimension tends to have extremely low values in other dimensions. We have also observed that clusters with large values of duration such as clusters 0 - 4 have convex shape. Additionally some convex shaped clusters have relatively even variance scale in all dimension, e.g. cluster 10 - 15.

Table IV summarises job counts and the ratio of jobs from different cells. When comparing the ratio of f and h in each cluster, it shows that f has a higher ratio in the clusters with longer duration with medium CPI and low CPU usage, e.g. cluster 3 and 4, while h has a higher ratio in the cluster with medium duration, medium CPI, and CPU usage > 1, e.g. 12, 13, 14. Given the observations from Table III, where h is more likely to achieve higher utilisation, these jobs could play a positive role in the resource utilisation. To draw a conclusion on the possible relation, time series analysis would be necessary. We will leave it as future work.

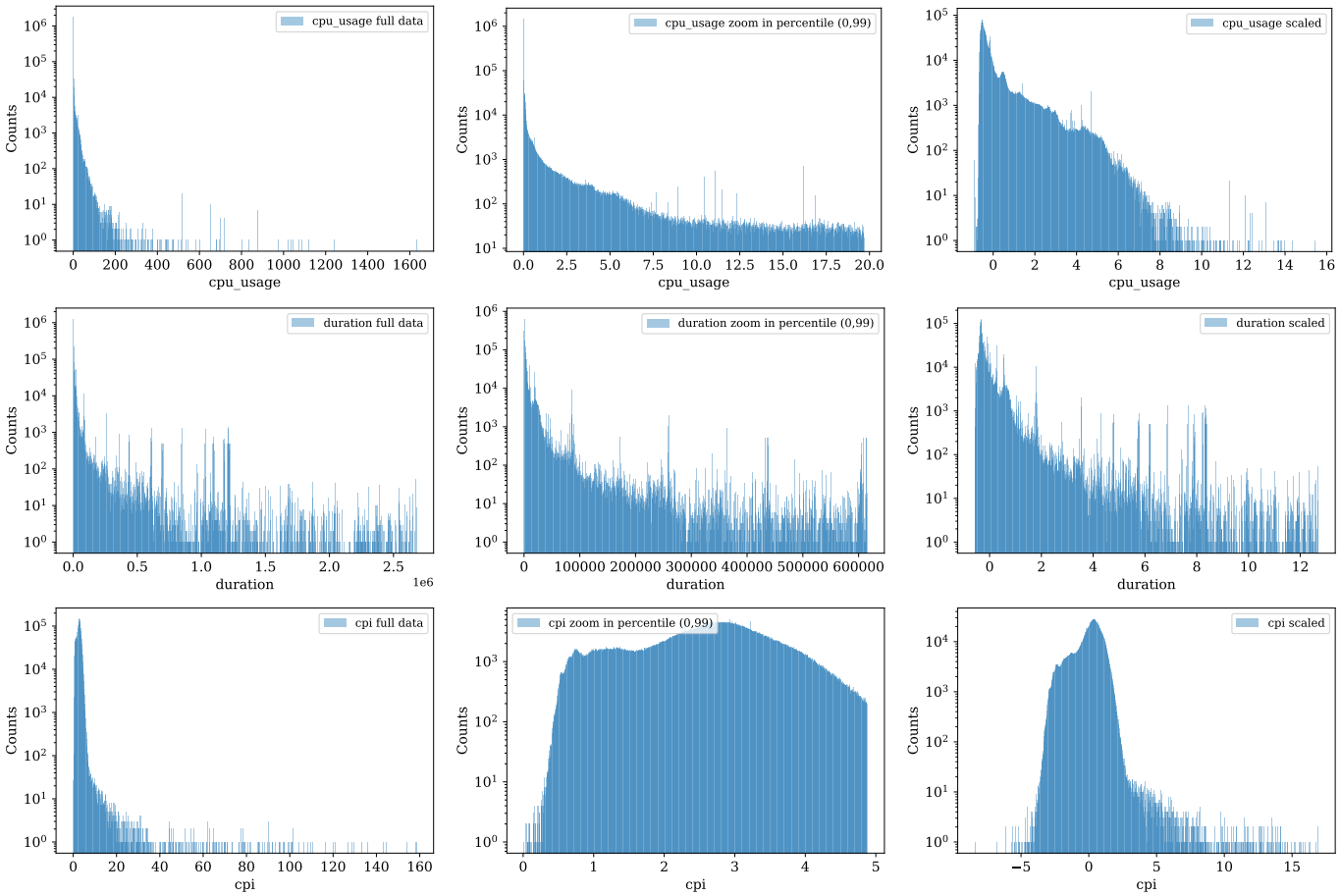

Fig. 8: Metrics distribution: the left columns show the full data, the middle column shows the zoom-in range of (0, 99) percentage of data, and the right column shows the full data after data preprocessing with scaling and normalisation. The point of data preprocessing is to reduce the range of outliers while preserve the basic shape of distribution.

VII. Lesson Learned

The features introduced in the new Google Cloud trace data V3 allow us to conduct inter-cluster analysis from the perspectives of underlying machine resources and the user job requests. As the first study of inter-cluster analysis in this data set, we summarise our findings as below:

- **Job metrics distribution**: The phenomenon of 'mice' and 'hogs' [11], [12], i.e. a minority of jobs utilise a majority of resources appears in all the cells of our study. The typical range of resource usage for both CPU and memory are between (1, 100), and the range can be further narrowed down depending on cells
- **Job metrics correlation**: Although we have collected 9 metrics, the number of metrics can be reduced down to 3, i.e. duration, CPU usage and CPI, because the remaining metrics are strongly correlated to either one of them. Such reduction cuts the search space significantly for machine learning techniques. However, it is worth noting that the conclusion of metrics correlation is drawn from the aggregated average value of the metrics. It is not strong enough to extend this conclusion to the value at time series level. And it would be interesting to further investigate the correlation and dependency relation for each metrics at a finer-grain level.
- **Inter-cluster variances in resource, jobs and utilisation**: Different cells in this data set can have different underlying machine resource structures, e.g. f and g have very similar heterogeneous compositions of machines with high in CPU and median in memory, whereas g has the most heterogeneous composition of machines. The utilisation is similar from either the perspective of total resource utilisation or per-machine utilisation. The findings could be useful for choosing a suitable subset of trace data as input to run scheduler simulation.
- **Job characteristics**: Our analysis of job metrics reveals 38 job categorises clustered using the metrics of duration, CPU usage and CPI as features. There are two patterns for the shape of jobs in the resulting clusters, i.e. the patterns of the concave shape and the convex shape. The concave shapes lack in variance in one or more dimensions and

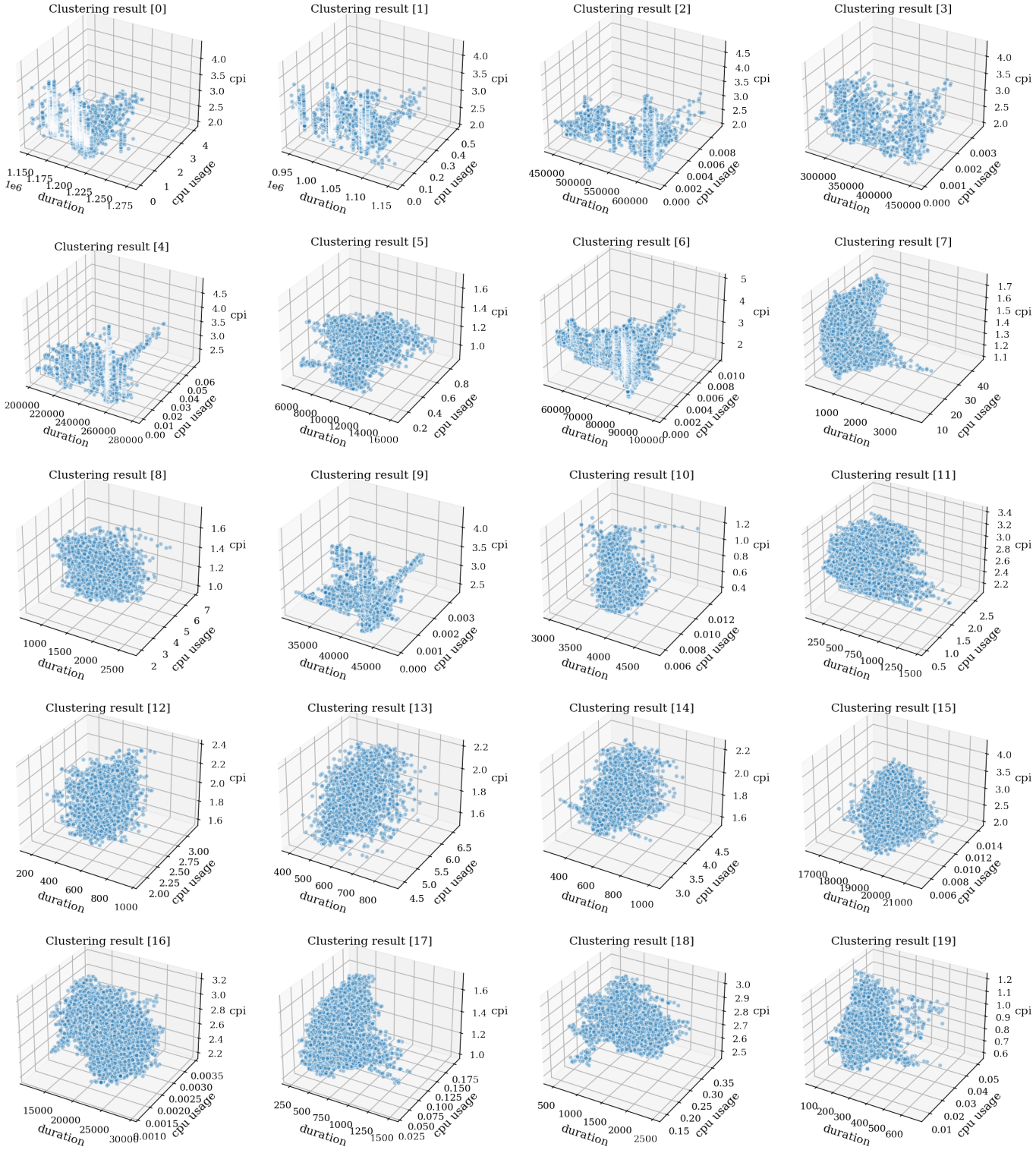

Fig. 9: 3D scatter plots showing unsupervised learning (*DBSCAN*) results part I: *index 0 - 19*. The job clusters become 3D dimensional objects in each plot. There appears to two types of shape: concave and convex. Clusters in the concave shapes lack variances in one or more dimensions. For example, cluster 0 - 4 are in the shape of planes, and cluster -1 (in part II) is lines. Clustering with high in duration are mostly in this shape, e.g. cluster 0 - 4. Clusters in convex shape have relatively even variance scale in all dimension, e.g. clusters 10 - 15.

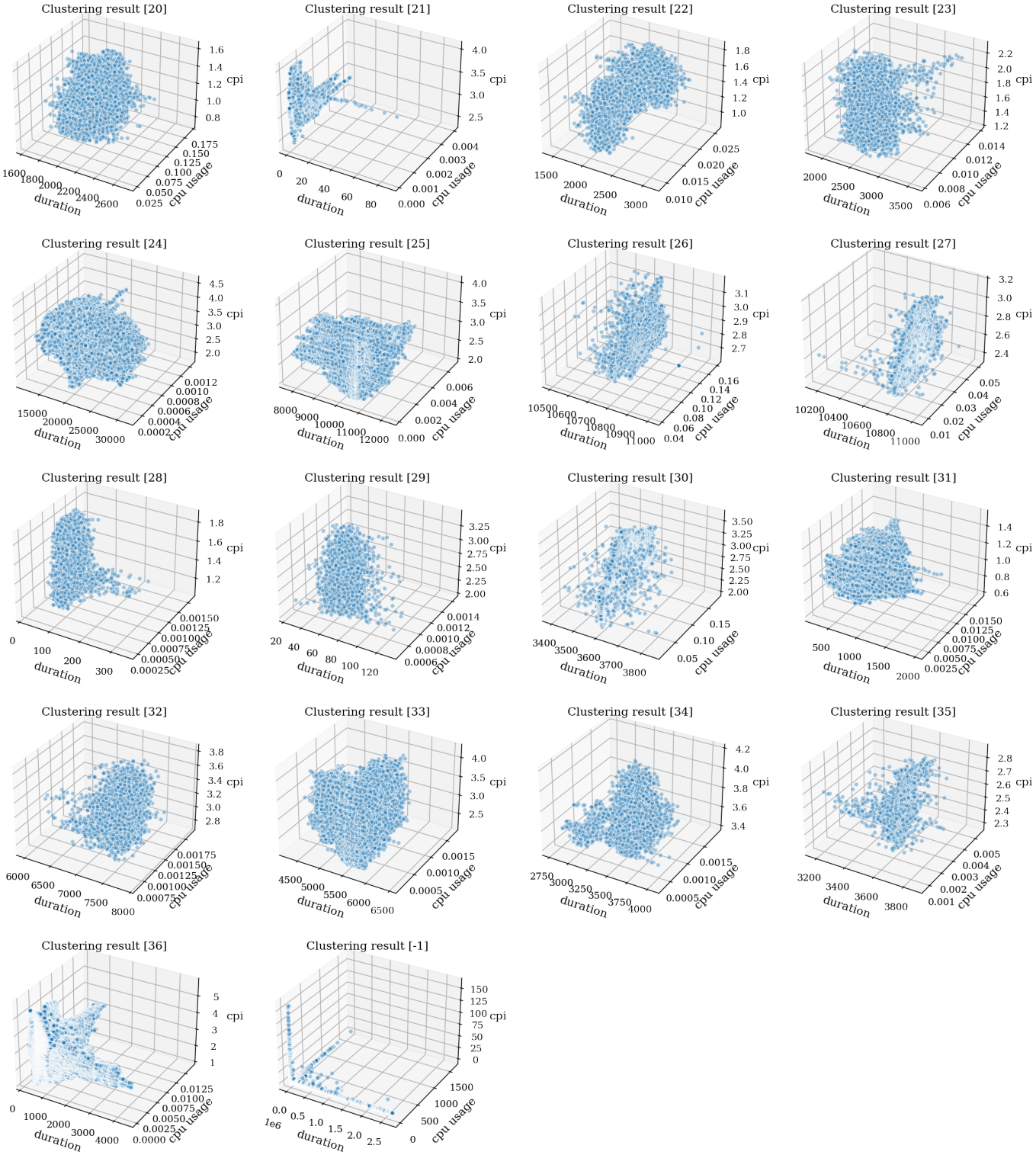

Fig. 10: 3D scatter plots showing unsupervised learning (*DBSCAN*) results part II: *index 20 - 36 & -1*.

the convex shape has a relatively even variance scale in all dimensions. Each resulting cluster with the same pattern has different value ranges in metrics. Each cell has a significantly different composition of the resulting clusters.

VIII. Related work

Trace data have been studied in analysing a wide range of phenomena of cloud data centre scheduling systems. To encourage the analysis, cloud operators publish new trace data to update the trend of changes in machine resources and user job demands, e.g. Microsoft Azure trace data [3], Alibaba cloud trace 2017 and 2019 [4] and Google Cluster data 2011 [2], [6] and 2019 [5], [12], [17].

For the previous version of Google trace data, i.e. the 2011 version, there were studies in analysing various aspects of the trace. For example, [7] studied job duration, resource utilisation and job request events to analyse differences of job/workload characteristics between Google cloud and Grid; [9] focused on machine life-cycle events and job scheduling data; [8] applied K-means on utilisation and application types to study job characteristics. A more recent study of the new Google 2019 data [12] focus is to compare 2019 new trace data with the previous version to highlight the changes in Borg scheduling as well as the trend of user job requests. Comparing to their work, our study is more focused on the new feature of the 2019 data set, i.e. multiple cells, to analyse inter-cluster utilisation differences from both machine resources and job characteristics point of view.

IX. Conclusion & Future work

Trace analysis has been playing an intensive role in understating the behaviours of modern cluster management system. In this paper, we conducted an inter-cluster analysis on the recently published Google trace 2019 data set. Our analysis focused on the distribution of underlying physical machines, e.g. machine numbers and resource, and metrics of computational job requests, e.g. job duration, utilisation and Cycles Per Instruction. We found that the composition of underlying machine resources in different cells can be substantially different, and the cells with similar composition can utilise resources differently depending on characteristics of job requests. We also identified a shape pattern from the resulting clusters which generated from applying an unsupervised learning algorithm on the metrics. The shapes are concave and convex, where the concave shapes lack in variance in one or more dimensions and the convex shape has relatively even variance scale in all dimension.

As pointed out in [12], one of the interesting uses of this trace data is to explore explainable scheduling. Our current study suggested potential factors of inter-cluster utilisation differences. However, it is not sufficient to draw strong conclusions or to answer 'why questions'. Ultimately, we would like to achieve this to understand the scheduling and utilisation status of cloud management systems. We plan to explore the fine-grained-level data, e.g. time-series, using dependency analysis in the hope of answering 'why questions' with causal relationship and confidence intervals.

Acknowledgement

This work is a part of the ABC project (Adaptive Brokerage for the Cloud) funded by EPSRC EP/R010528/1.

References

[1] J. L. Hellerstein, "Google cluster data," Google research blog, Jan. 2010, posted at http://googleresearch.blogspot.com/2010/01/google-cluster-data.html.

[2] J. Wilkes, "More Google cluster data," Google research blog, Mountain View, CA, USA, Nov. 2011, posted at http://googleresearch.blogspot.com/2011/11/more-google-cluster-data.html.

[3] "Microsoft azure trace." [Online]. Available: https://github.com/Azure/AzurePublicDataset

[4] "Alibaba cluster trace." [Online]. Available: https://github.com/alibaba/clusterdata

[5] J. Wilkes, "Google cluster-usage traces v3," Google Inc., Mountain View, CA, USA, Technical Report, Apr. 2020, posted at https://github.com/google/cluster-data/blob/master/ClusterData2019.md.

[6] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: format + schema," Google Inc., Mountain View, CA, USA, Technical Report, Nov. 2011, revised 2014-11-17 for version 2.1. Posted at https://github.com/google/cluster-data.

[7] S. Di, D. Kondo, and W. Cirne, "Characterization and comparison of cloud versus Grid workloads," in *International Conference on Cluster Computing (IEEE CLUSTER)*, Beijing, China, Sep. 2012, pp. 230–238.

[8] S. Di, D. Kondo, and C. Franck, "Characterizing cloud applications on a Google data center," in *42nd International Conference on Parallel Processing (ICPP)*, Lyon, France, Oct. 2013.

[9] Z. Liu and S. Cho, "Characterizing machines and workloads on a Google cluster," in *8th International Workshop on Scheduling and Resource Management for Parallel and Distributed Systems (SRMPDS)*, Pittsburgh, PA, USA, Sep. 2012. [Online]. Available: http://www.cs.pitt.edu/cast/abstract/liu-srmpds12.html

[10] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *ACM Symposium on Cloud Computing (SoCC)*, San Jose, CA, USA, Oct. 2012. [Online]. Available: http://www.pdl.cmu.edu/PDL-FTP/CloudComputing/googletrace-socc2012.pdf

[11] O. A. Abdul-Rahman and K. Aida, "Towards understanding the usage behavior of Google cloud users: the mice and elephants phenomenon," in *IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, Singapore, Dec. 2014, pp. 272–277.

[12] M. Tirmazi, A. Barker, N. Deng, M. E. Haque, Z. G. Qin, S. Hand, M. Harchol-Balter, and J. Wilkes, "Borg: the Next Generation," in *Proceedings of the Fifteenth European Conference on Computer Systems (EuroSys'20)*. Heraklion, Greece: ACM, 2020. [Online]. Available: https://doi.org/10.1145/3342195.3387517

[13] A. Verma, L. Pedrosa, M. R. Korupolu, D. Oppenheimer, E. Tune, and J. Wilkes, "Large-scale cluster management at Google with Borg," in *Proceedings of the European Conference on Computer Systems (EuroSys'15)*, Bordeaux, France, 2015. [Online]. Available: https://dl.acm.org/doi/10.1145/2741948.2741964

[14] "Google BigQuery." [Online]. Available: https://cloud.google.com/bigquery

[15] "Paper resource IPython Notebook." [Online]. Available: https://colab.research.google.com/drive/1uQY6xfeMgL4QfzbPDo14Z8958dfANXxa

[16] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[17] J. Wilkes, "Yet more Google compute cluster trace data," Google research blog, Mountain View, CA, USA, Apr. 2020, posted at https://ai.googleblog.com/2020/04/yet-more-google-compute-cluster-trace.html.