Big Data Course Work Report

Mohsin Gul

mohsin.gul@city.ac.uk

Task 1D (i)

Fig. 1 Cluster metrics with 1 master with 1 vCPU + 3 workers with 1 vCPU (Maximal Cluster, default 2 partitioning)

Fig1 shows job submission with default 2 partitions. It starts by submitting into cluster, the CPU was utilized almost 15% percent. It is also the largest amount of CPU power consumed by the cluster. CPU was utilized on an average of 6% approx. For throughput, the fig1 shows that it peaks at 570bps and then rapidly decreases and then again peaks at about 500/s. Moreover, disk bytes or disk size metric follows the same trend as network packets, first it peaks at above 6 mb/s and then declines to less than 1mb/s after which it increases again to almost 6mb/s.

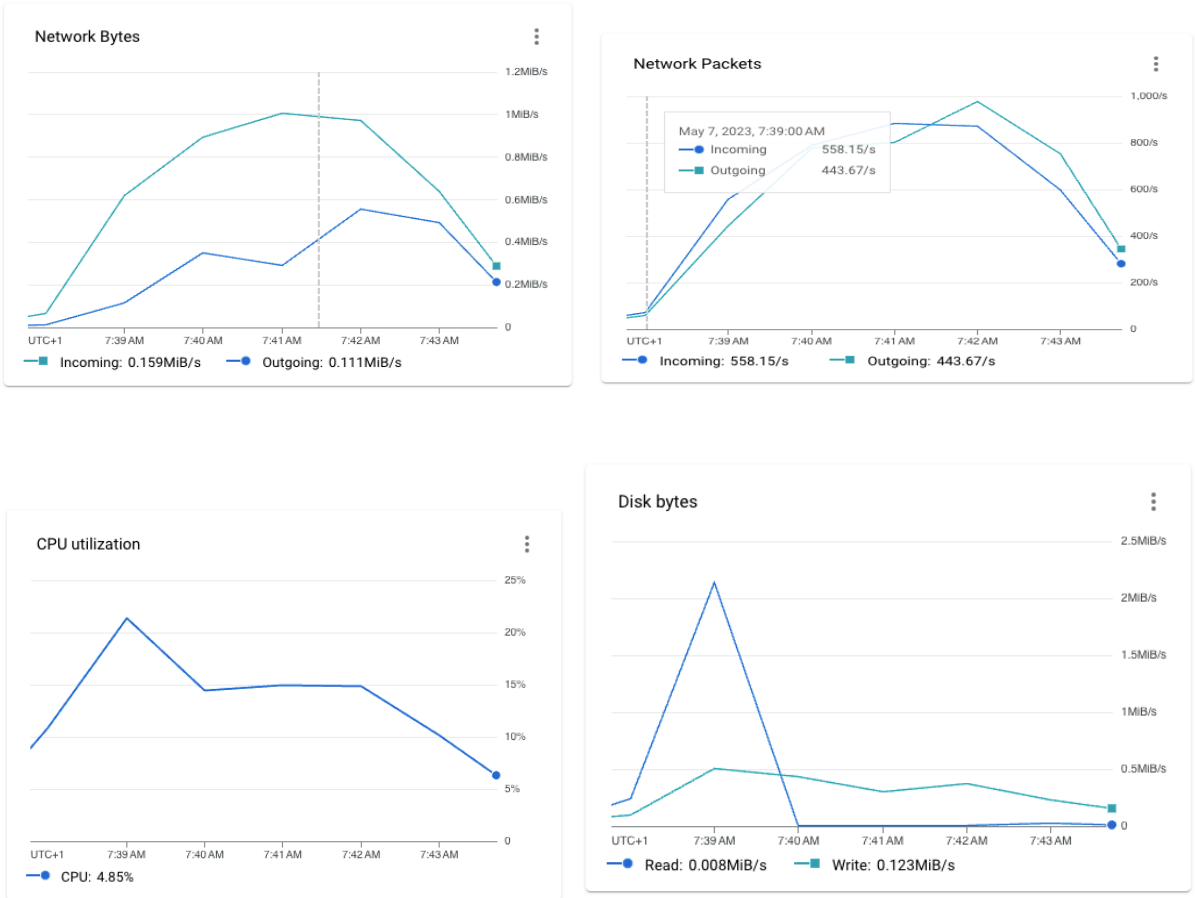
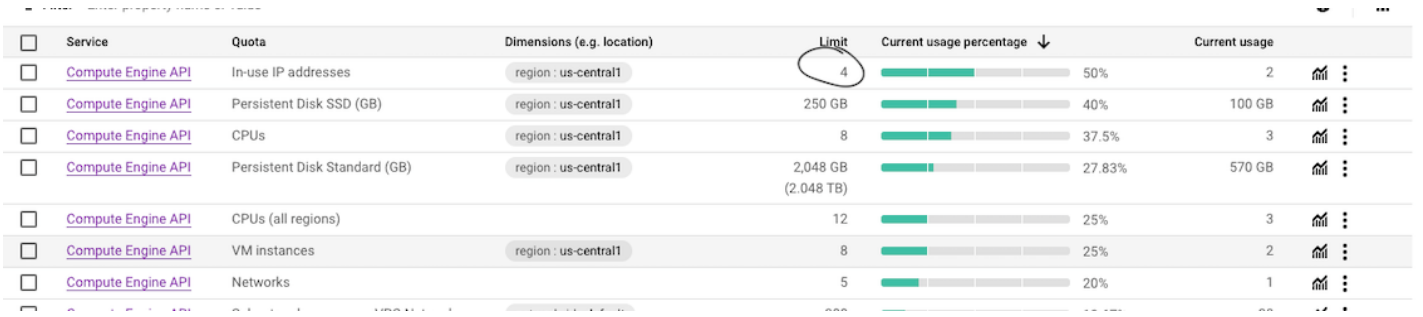


Fig 2 Cluster metrics with 1 machine with 1 vCPU + 3 workers with 1 vCPU (Maximal Cluster, 16 partitioning)

Fig 2 shows the metrics for parallelization adopted of 16 partitions. It is quite evident that increasing the partition leads to increase the packets sent or received as it leads to increase in disk bytes as it tops at almost 1000/s from 500/s in fig 1. Increasing the number of partitions generally boosts CPU usage if the workload is evenly distributed across them. If the workload is unbalanced, certain partitions may finish significantly sooner than others, leaving idle CPUs. In the fig2, it shows 22% of CPU utilized as compared to approximately 15% with 2 partitions. In general, more partitions could result in higher parallelism and quicker processing times, but more partitions could also mean more network traffic, which could slow the operation down. To determine the best partition size for a particular workload, it is crucial to test out various sizes and keep an eye on metrics.

Error:



Service	Quota	Dimensions (e.g. location)	Limit	Current usage percentage	Current usage
Compute Engine API	In-use IP addresses	region : us-central1	4	50%	2
Compute Engine API	Persistent Disk SSD (GB)	region : us-central1	250 GB	40%	100 GB
Compute Engine API	CPUs	region : us-central1	8	37.5%	3
Compute Engine API	Persistent Disk Standard (GB)	region : us-central1	2,048 GB (2.048 TB)	27.83%	570 GB
Compute Engine API	CPUs (all regions)		12	25%	3
Compute Engine API	VM instances	region : us-central1	8	25%	2
Compute Engine API	Networks		5	20%	1

Fig3: Error of Use Ip Address Limit

As illustrated in Fig. 2 the largest amount of CPU power used by the cluster is more than 85 percent. Normally the CPU utilized is 2.9 percent.

Metrics	Network Bytes	Network packets	Disk Bytes	Disk Operations
Max Incoming	20.18 MiB/s	14.53 K/s	-	-
Max outgoing	0.10 MiB/s	0.53 K/s	-	-
Max Read	-	-	32.54 MiB/s	1.54 K/s
Max Write	-	-	156.07 MiB/s	0.56 K/s

Table. 1 Maximum parameters of cluster with 1 master with 1 vCPU + 3 workers with 1 vCPU (maximal) after using second parameter.

Using 16 partitions was anticipated to result in an increase in network bytes, network packets, disc bytes, and disc operations. This is because having more partitions can result in more network and disc I/O as there are more simultaneous data streams being handled. There may be a trade-off between performance and resource utilization in this, though. It's vital to determine the ideal number of partitions for your particular use case since in some circumstances, increasing the number of partitions too much can result in excessive resource utilization and poor performance.

Clusters	Initialization time	Server connection time
1 master with 1 vCPU + 3 workers with 1 vCPU (maximal) only on two nodes	7732ms	8903ms
1 master with 1 vCPU + 3 workers with 1 vCPU (maximal) after using second parameter	11724ms	10237ms

Table. 2 Parameter Config for CPU of 1 master with 1 vCPU + 3 workers with 1 vCPU pre and post adding second parameter

Task 1D (ii)

Clusters	Initialization time	Server connection time
1 master with 2 vCPU + 3 workers with 1 vCPU	5225ms	6311ms
1 master with 8 vCPU	5373ms	5601ms

Table. 3 Parameter Config for the CPU for 1 master with 2 vCPU + 3 workers with 1 vCPU and 1 master with 8 vCPU

As shown in Table. 3, the initialization times for the 1 master with 2 vCPU + 3 workers with 1 vCPU cluster were 5225ms and 6311ms, respectively, whereas the initialization times for the 1 master with 8 vCPU cluster were 5373ms and 5601ms.

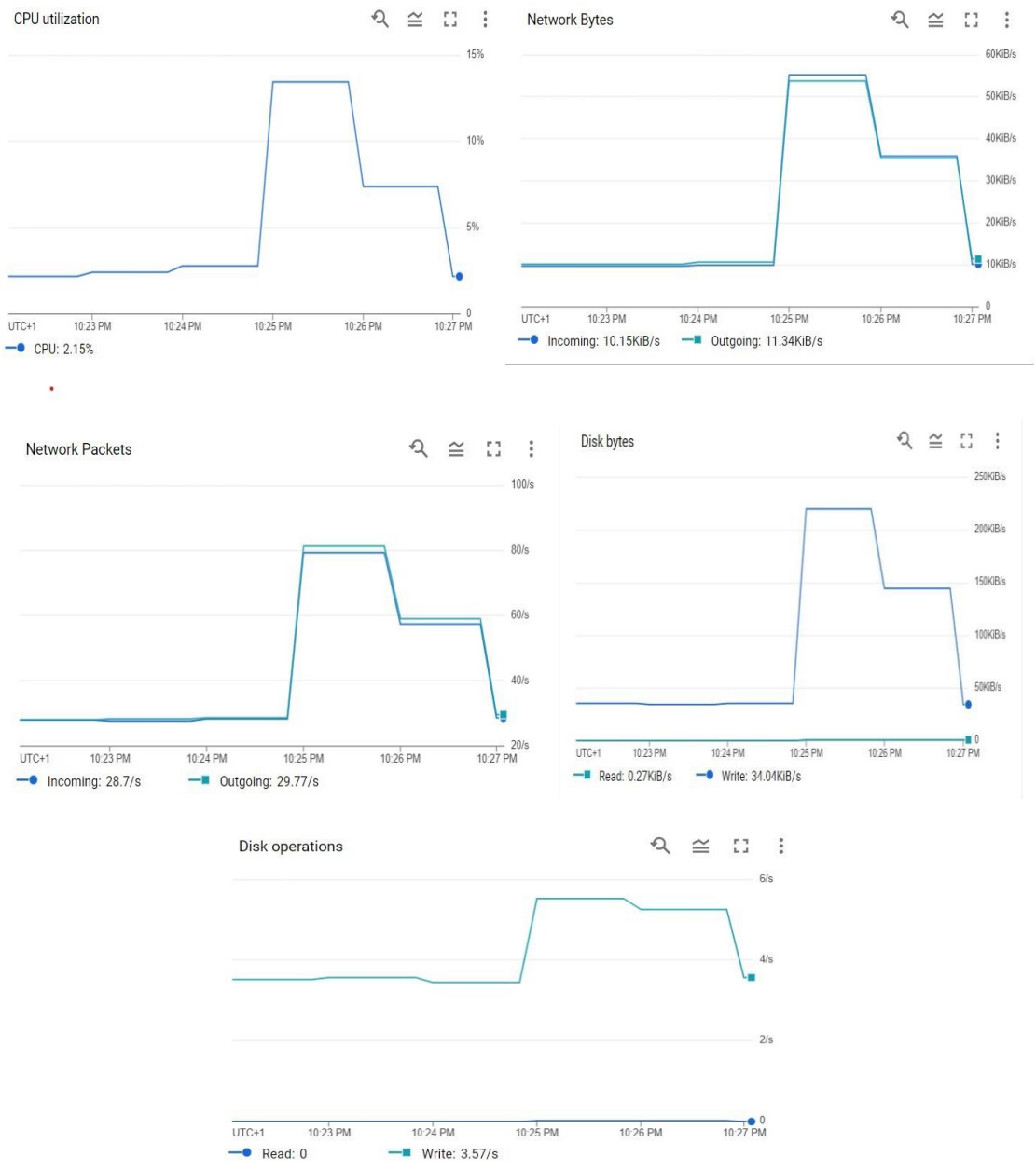


Fig. 3 Metrics of cluster with the 1 master with 2 vCPU + 3 workers with 1 vCPU

Metrics	Network Bytes	Network pack	Disk Bytes	Disk Op
Max incoming	57.1 KiB/s	81.17 /s	-	-
Max outgoing	55.66 KiB/s	79.23 /s	-	-
Max Read	-	-	0.02 KiB/s	0.03 /s
Max Write	-	-	240.19 KiB/s	6.83 /s

Table. 4 Maximum parameters of cluster with the 1 master with 2 vCPU + 3 workers with 1 vCPU

Fig. 4 Parameter Config of clusters with 1 master with 8 vCPU

Metrics	Network Bytes	Network packets	Disk Bytes	Disk Operations
Maximum incoming	3.54 MiB/s	2.393 K/s	-	-
Maximum outgoing	0.18 MiB/s	0.08 K/s	-	-
Maximum Read	-	-	8.27 MiB/s	571.18 /s
Maximum Write	-	-	18.15 MiB/s	128.72 /s

Table. 5 Maximum parameters of cluster with 1 master with 8 vCPU

Task 1D (iii)

In our labs the data was often stored in file system and processed on a single workstation. Due to that the processing speed may be poor if the data size exceeds the machine's memory capacity and, they are often not built to handle massive data sets. In contrast, Spark is made for distributed computing, where massive data volumes are processed across a cluster of devices. Each computer in the cluster processes a portion of the data that is stored in a distributed file system, such as Hadoop Distributed File System (HDFS) or Google Cloud Storage. As a result, many machines can process distinct subsets of the data concurrently in parallel. The main advantage I perceived is that parallelization approach used in spark by the method called RDD (Resilient Distributed Database) enables one to perform transformations in parallel therefore making spark to handle large data sets.

2d.

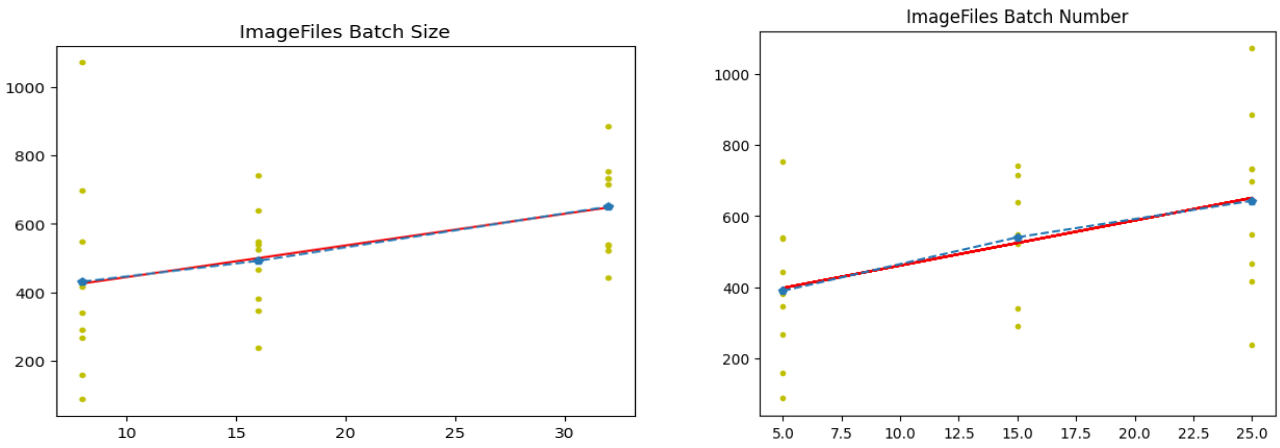


Fig. 5 Batch Size of the reading TFRecord files

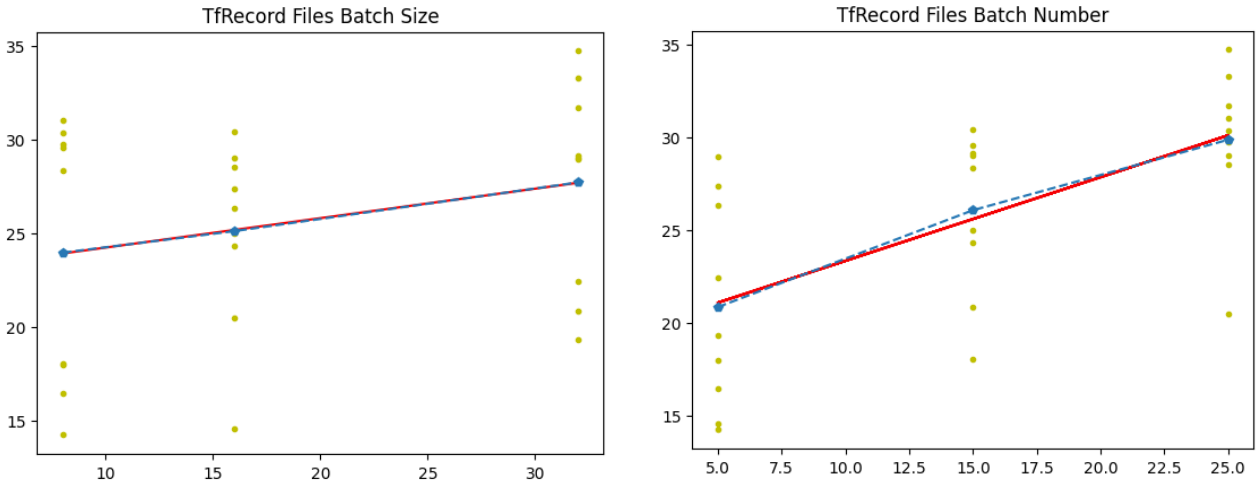


Fig. 6 Batch Number of the reading TFRecord files

Parameters	Slope	Intercept	P-value
Image files Batch Size	9.24	352.	0.03
Image files Batch Number	12.6	334	0.014
Image files Repetitions	6.35	511	0.907
TFRecord files Batch Size	0.157	22.6	0.17
TFRecord files Batch Number	0.45	18.8	0.0003
TFRecord files Repetitions	1.68	22.22	0.23

Table. 7 Slope, Intercept and P-value of all parameter combinations

From the table and fig we can see that the p- value of batch size of image files is 0.03 and batch number is 0.014 that is really low so, while testing it locally as compared to cloud revealed that the when testing locally the performance of model will be improved immensely. As shown in Figs. 6 and 7, the batch size of TfRecord files is 0.17, while the batch number of TfRecord files is 0.003, demonstrating a relationship between the two. If there is a lot of data available, selecting better combinations may allow one to increase the P-value even further. When submitting the job on cloud high latency was observed of about 5510ms when it was connected to the servers.

3a

These initial IO capacity constraints and the autoscaling mechanism, which automatically expands the IO capacity for a bucket as the request rate increases, should be considered while running speed tests concurrently on the cloud. By going beyond these thresholds, we might be able to spot any cloud infrastructure bottlenecks that might be affecting other users' performance. Moreover, when a bucket is approaching its IO capacity limit, Cloud Storage typically detects it within minutes and adjusts the load to more servers. Consequently, if the request rate on your bucket increases faster than you can handle, you may experience temporary restrictions such as higher latency and error rates. To adapt best practices, cloud storage can adapt and provide optimal performance if your request rate is increased gradually. If you suddenly start sending many queries, Cloud Storage might not be able to identify the increased pace quickly enough, and you might hit temporary restrictions, leading to higher latency and error rates. To guarantee optimum performance, it is advised to start with a lower request rate and increase it gradually.

The coursework's focus on parallelization, measuring throughput speed, and analyzing metrics such as CPU, network bytes, network packets, and disk size is related to the paper "Cherry Pick: Adaptively Unearthing the Best Cloud Configurations for Big Data Analytics" because the paper discusses the challenges of selecting the best cloud configuration for big data analytics jobs running in clouds, which involves considering various metrics such as CPU, memory, disk, and network. The paper suggests a system called Cherry Pick that uses Bayesian optimization to create performance models for different applications and cloud configurations. The models are just precise enough to tell the best or nearly the best configuration from the rest with just a few test runs. Thus, the emphasis in the coursework on examining the clusters on different configuration on metrics like CPU utilization, network bytes, network packets, and disc bytes is pertinent to comprehending the problems and suggestions made in the paper. We can maximize our rewards while assessing the chances of success in various combinations by cherry-picking. Cherry picking may have been more effective in this case, but there is a drawback to overcome, including a complex performance model, a cost model, and the heterogeneity of the applications, to mention a few.

3b.

To collect the data for this study, a batch processing speed test was conducted. To suit each user's particular requirements, the extensive range of configuration options for the CPU can be modified. It is now possible to employ cherry picking in an easy way because Bayesian optimization techniques were developed. Data extraction is essential for cherry-picking, but based on a range of factors, such as the scenario and requirements, the speed at which micro batch processing is carried out should be changed. Cherry picking is a good strategy for influencing price fluctuations since it gives the most crucial information. Network latency is encountered because of the interaction of several variables. The data had already been stored into the memory prior to starting training, which is another assumption made in this study along with training length and memory estimation. In contrast to Task 2d when applying the linear regression, the data was transferred to the cloud before analysis. Moreover, In this paper data parallelism and spatial parallax are discussed which are examples of distributed training and when considering task 1c where there were quota restrictions for me I used distributed training with various batch sizes in order to resolve this issue.

Word count:

1726

Colab link -