PERFORMANCE EVALUATION OF TERAPIXEL RENDERING IN CLOUD (SUPER) COMPUTING

SUSHANTH SHIVPURA RAMESH - C1044105
Module :CSC8634

Table of Content

1. Abstract

Cloud computing has allowed the businesses and researchers to scale up or scale down the computing resources as per the needs without requiring investing capital upfront for physical infrastructure. This has helped in leveraging the pay as you use model. By allocating and optimizing the cloud architecture resources we can further reduce the cost. The aim of the analysis is to extract information from the terapixel image generated data from the application and system metrics and pinpoint where the resource optimization should be focused. This project provides insight into key performance metrics around task assignments, GPU characteristics for terapixel rendering using numerical and exploratory data analysis.

2. Introduction

Cloud services, which offer wide a range of services such as Software as a Service (SaaS), Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) that are delivered to customers over the internet. In recent times we can clearly see a shift in using these cloud services which are offered by Google cloud Platform, Amazon Web services etc. rather than setting up their own hardware infrastructure on their premise (on prem infrastructure). Using Cloud services offers some advantages than setting up a on perm infrastructure such as, renting or leasing extensive range of hardware infrastructure which saves huge capital which is requires to setup on prem. The other major advantage of using cloud service is that resources can be scaled up or scaled down based on the work load. This would help us to leverage the pay as you use model offered by the cloud services i.e., we are only billed based on the amount of time we use the resource for.

Image rendering is a GPU resource consuming task. Using Infrastructure or Platform as a service offered by cloud services would eliminate the hardware setup and initial investment cost constraints. An example use case and implementation for rendering images in real time using IaaS cloud infrastructure can be found in “Image-Based Network Rendering of Large Meshes for Cloud Computing” [1]. Terapixel 3D city visualization of Newcastle Upon Tyne uses a scalable IaaS cloud GPU node for the rendering. The Cloud services do not completely optimize the use of the underlying resources. To improve the scalability and optimizing the GPU resource utilization we are initiating the data mining project of the Terapixel rendering cloud architecture.[2] The project will focus on exploratory data analysis and follow CRISP-DM methodology.

3. Business Understanding

The cloud architecture on which the terapixel images are rendered uses compute nodes that heavily relies on the underlying GPU hardware. GPUs are power hungry hardware, as its main performance is established based on the number of parallel tasks it can run simultaneously. So, there are lot of factors that can affect the performance of the GPU.

The aim of this data mining and analysis project is to extract the performance evaluation metrics from the cloud architecture that is implemented to support the terapixel image rendering. Analysing this metrics would help us to identify how to improve the cost efficiency in running this cloud solution and optimizing the performance of rendering the image by tweaking the GPU utilization. we are using the following questions as an aid to analyse the data extracted from the cloud solution hardware.

- How is the GPU temperature and performance interrelated?
- Is there any relation between increased power drawn by the GPU and render time?
- Is there any particular GPU cads which perform better than the others ?
- Which event types dominates the task runtimes?
- Is there any specific pattern in the tiles or pixels which utilizes the GPU more or less to render?

4. Data Set Overview

There are three CSV files extracted from the application checkpoint and system metric output while rendering of terapixel image. The three data files consists of Application Checkpoint data, GPU data and Task-X-Y data. An initial data understanding of these data frames are as below.

Application Checkpoints Data: The data frame consists of 660400 rows and 6 columns as seen summary of the data frame. The column names of the data frame are interpreted as follows:

- **Timestamp:** Date and time of the event measured in milliseconds.
- **Hostname:** Auto-assigned name by azure batch system for each of the virtual machine. There are 1024 unique hostnames.
- **EventName:** There are 5 event that takes place while rendering the image

  - **TotalRender:** Rendering the entire task
  - **Saving Config:** Specific configurations related to each task are saved
  - **Render:** Tiles of the image that is being rendere
  - **Tiling:** Post-processing of the rendered tile
  - **Uploading:** Output of the post-processed image tile are uploaded to azure storage

- **EvenType:** starting or terminating of an event , START or STOP state Jobid: There distinct states of the azure batch job which refers to the level of zoom while rendering the image (4,8,12).
- **Taskid:** Contains unique ids of the azure batch task. Each of the task id would have all the five event names associated along with the start and stop timestamp.

```
Application_checkpoint Data contains: 660400 Rows and 6 Columns
```

	timestamp	hostname	eventName	eventType	jobId	taskId
0	2018-11-08T07:41:55.921Z	0d56a730076643d585f77e00d2d8521a00000N	Tiling	STOP	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	b47f0263-ba1c-48a7-8d29-4bf021b72043
1	2018-11-08T07:42:29.842Z	0d56a730076643d585f77e00d2d8521a00000N	Saving Config	START	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	20fb9fcf-a927-4a4b-a64c-70258b66b42d
2	2018-11-08T07:42:29.845Z	0d56a730076643d585f77e00d2d8521a00000N	Saving Config	STOP	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	20fb9fcf-a927-4a4b-a64c-70258b66b42d
3	2018-11-08T07:42:29.845Z	0d56a730076643d585f77e00d2d8521a00000N	Render	START	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	20fb9fcf-a927-4a4b-a64c-70258b66b42d
4	2018-11-08T07:43:13.957Z	0d56a730076643d585f77e00d2d8521a00000N	TotalRender	STOP	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	20fb9fcf-a927-4a4b-a64c-70258b66b42d

GPU Data: file contains details of the GPUs and its associated attributes. There are 1048575 rows and 8 columns. Timestamp and Hostname columns are similar to the once in the Application Checkpoint data. The remaining column attributes are as explained below.</p>

- **gpuSerial:** Unique serial number associated with each of the 1024 GPU cards
- **gpuUUID:** Unique id generated by azure system to the GPU card
- **powerDrawWatt:** Power drawn by each GPu card in Watts.
- **gpuTemC:** Recording of temperature of the GPU card while operating in Celsius.
- **gpuUtilPerc:** Percantage of gpu processor cores utilized.
- **gpuMemUtilPerc:** Percentage of GPU RAM utilized.

```
GPU Data contains: 1543681 Rows and 8 Columns
```

	timestamp	hostname	gpuSerial	gpuUUID	powerDrawWatt	gpuTempC	gpuUtilPerc	gp
0	2018-11-08T08:27:10.314Z	8b6a0eebc87b4cb2b0539e81075191b900001C	323217055910	GPU-1d1602dc-f615-a7c7-ab53-fb4a7a479534	131.55	48	92	
1	2018-11-08T08:27:10.192Z	d8241877cd994572b46c861e5d144c85000000	323617020295	GPU-04a2dea7-f4f1-12d0-b94d-996446746e6f	117.03	40	92	
2	2018-11-08T08:27:10.842Z	db871cd77a544e13bc791a64a0c8ed50000006	323217056562	GPU-f4597939-a0b4-e78a-2436-12dbab9a350f	121.64	45	91	
3	2018-11-08T08:27:10.424Z	b9a1fa7ae2f74eb68f25f607980f97d7000010	325217085931	GPU-ad773c69-c386-a4be-b214-1ea4fc6045df	50.23	38	90	
4	2018-11-08T08:27:10.937Z	db871cd77a544e13bc791a64a0c8ed50000003	323217056464	GPU-2d4eed64-4ca8-f12c-24bc-28f036493ea2	141.82	41	90	

Task-x-y data: This file contains 6579 Rows and 5 columns associated with x y cordiantes of the tile of the image rendered. Job id is similar to the Application checkpoint data i.e the id of the azure batch task.

- **X :** x coordinates of the rendered image tile
- **Y :** y coordinates of the rendered image tile.
- **Level :** Zoom levels of image ranging from 1 to 12 while seeing only intermediate levels 4,8 and 12 derived while tiling process.

```
task-x-y data contains: 65793 Rows and 5 Columns
```

	taskId	jobId	x	y	level
0	00004e77-304c-4fbd-88a1-1346ef947567	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	116	178	12
1	0002afb5-d05e-4da9-bd53-7b6dc19ea6d4	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	142	190	12
2	0003c380-4db9-49fb-8e1c-6f8ae466ad85	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	142	86	12
3	000993b6-fc88-489d-a4ca-0a44fd800bd3	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	235	11	12
4	000b158b-0ba3-4dca-bf5b-1b3bd5c28207	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	171	53	12

5. Data Prepocessing

All the three datasets must be wrangled and merged before any analysis can be done. The following steps have been implemented as a part of data pre-processing.

First, we merge task-x-y.csv and application checkpoint.csv on job id and task id using left join. Next, we combine gpu.csv with the combined data frame of task-x-y.csv and application checkpoint.csv on time stamp. To ensure there are no duplicate data points, we remove duplicate rows from the merged data set.

The time stamps column on the data set cannot be directly used, we extract only the time part i.e. Hour:Minute:Second convert it to numeric from character and store it in a new column on the merged data set.

There are 5 events recorded namely TotalRender, Render, Tiling, saving config and uploading. Each of these events have a START and stop time. We have calculated the cumulative time duration of each of the event as follows.

10 data frames for the combination of 5 events and 2 event types has been created. Next, we merge the respective events on the task ids. With the merged data we calculate the time taken for each event using start and stop column. The duration is calculated by subtracting the start time from the stop time. Finally, we stack all the data frames and create a csv file for our analysis.

Consideration's

To analyse the tasks and event task that are run on GPU, we need to merge the GPU data with Application checkpoint data. The merge can be done based on the timestamp column and hostname. However, merging on time stamp directly would cause mismatching results. For the purpose of this analysis, we have done so, as we take the average values for most of the features this has a negligible errors on our results.

6. Data Analyses and Modelling

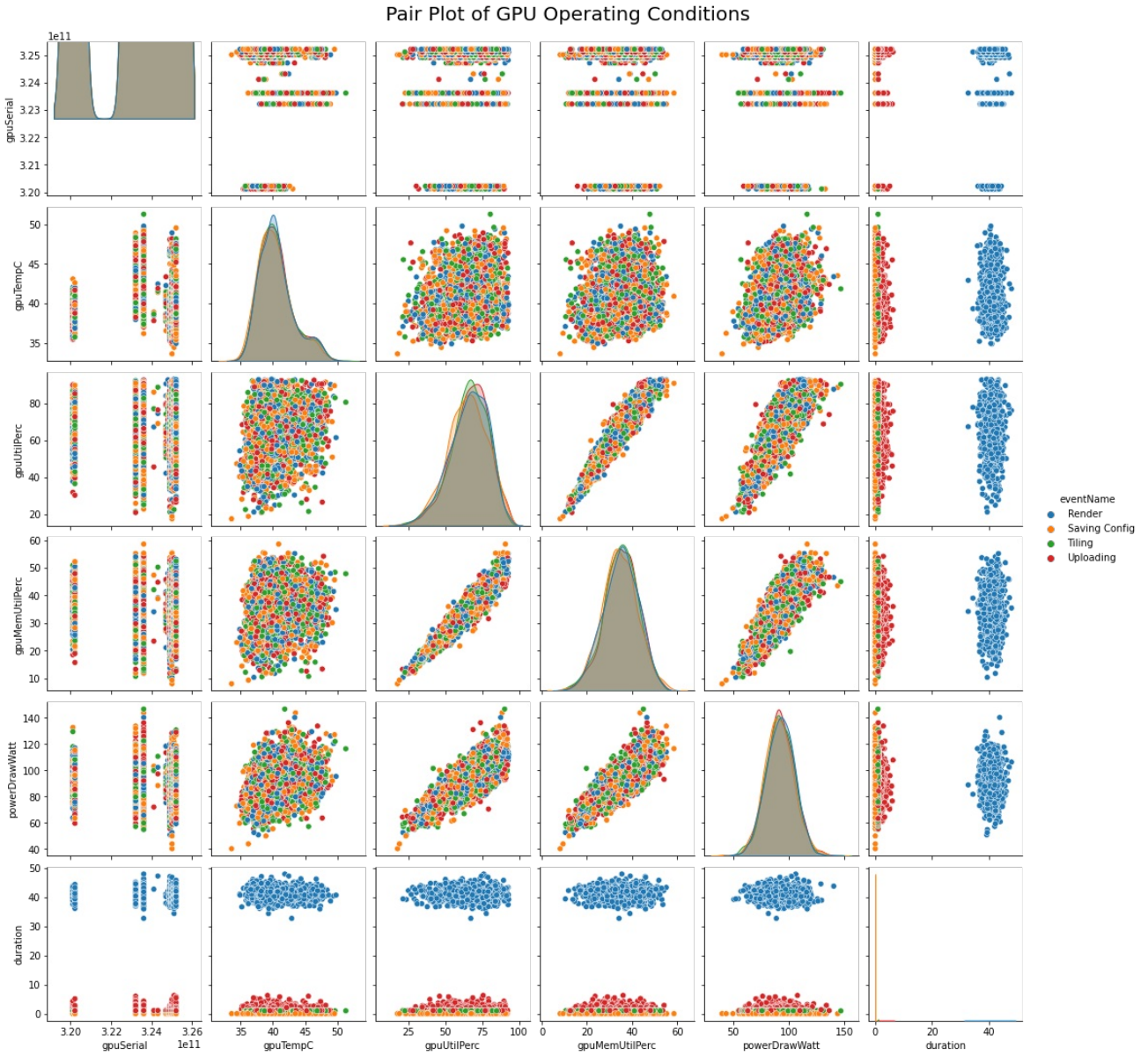
With the data now merged and wrangled, we can now extract the necessary information for the questions men-tioned in the previous sections. To get a broad overview of the data and extract some interesting features of the data we would begin our initial analysis.

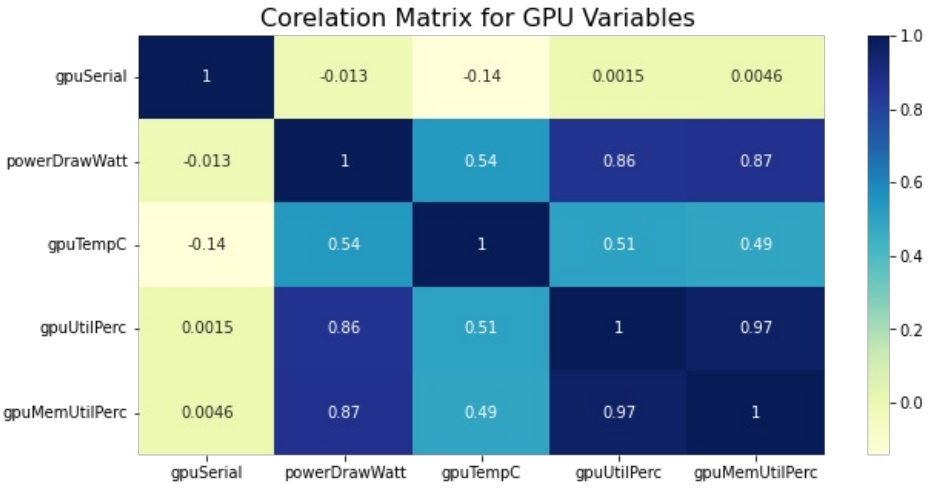
Initial Analysis

```
First few rows of the merged data frame
```

	eventName	jobId	taskId	hostname	START	x	y	level	gpuSerial
0	Render	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	e7776af5-510d-4ec4-b2d3-222b5df3307b	dcc19f48bb3445a28338db3a8f002e9c00000I	08:14:53.364000	36	180	12	323617042673
1	Render	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	339d3724-dcf6-41f1-b30d-c71107befcee	db871cd77a544e13bc791a64a0c8ed5000001B	08:21:58.845000	21	105	12	325117172104
2	Render	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	1bf9597f-77bd-4b37-aadd-a5af9baa2308	d8241877cd994572b46c861e5d144c85000006	07:53:10.809000	59	243	12	323617042670
3	Render	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	1bf9597f-77bd-4b37-aadd-a5af9baa2308	d8241877cd994572b46c861e5d144c85000006	07:53:10.809000	59	243	12	323617021127
4	Render	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	e742280f-948b-4f6c-b663-bb7f40c41af2	6139a35676de44d6b61ec247f0ed865700000Z	07:41:31.839000	173	5	12	325017018451

	gpuSerial	eventName	gpuTempC	gpuUtilPerc	gpuMemUtilPerc	powerDrawWatt	duration
0	320118118607	Render	36.916667	88.583333	48.750000	105.404167	39.366583
1	320118118607	Saving Config	36.518519	70.185185	38.333333	84.607037	0.002704
2	320118118607	Tiling	36.440000	83.040000	44.880000	103.791600	0.937080
3	320118118607	Uploading	35.428571	31.785714	18.571429	62.997857	1.050143
4	320118118641	Render	39.375000	80.500000	44.312500	100.221250	42.524313



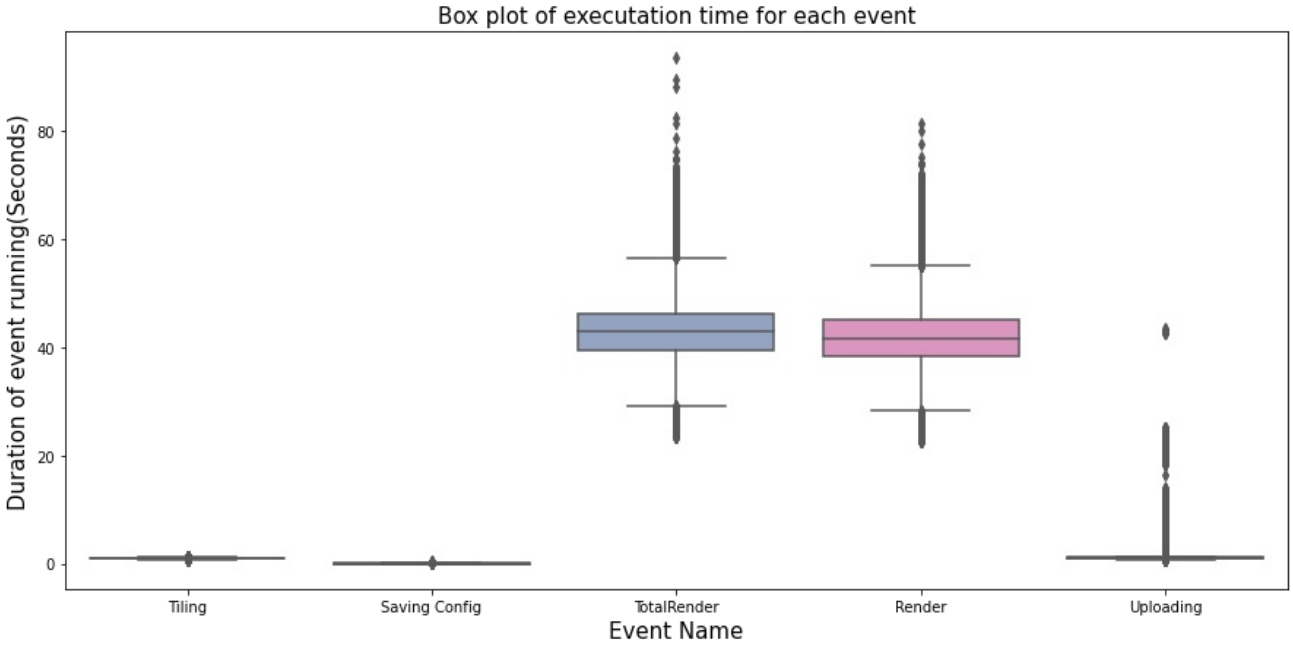


From the above pair plot and corelation matrix, we can deduce the following:

- Render event dominates most of the feature that relates to GPU performance such as duration, GPU Power Draw, Percent Utilisation of GPU Cores and Percent Utilisation of GPU Memory. This indicates that Rendering event consumes most of GPU capabilities than any other event.
- The GPU Memory utilization is very highly correlated with GPU utilization percent a value of 0.97 between them.
- power Drawn is highly correlated with GPU memory utilization and GPU utilization percent with scores of 0.87 and 0.86 respectively.
- GPU temperature feature does not show any strong correlation with any other features which have scores in the range of 0.54 to 0.49

From the above information we would be able to analyise the questions in the biusness objective.

Events Dominating the Run Time

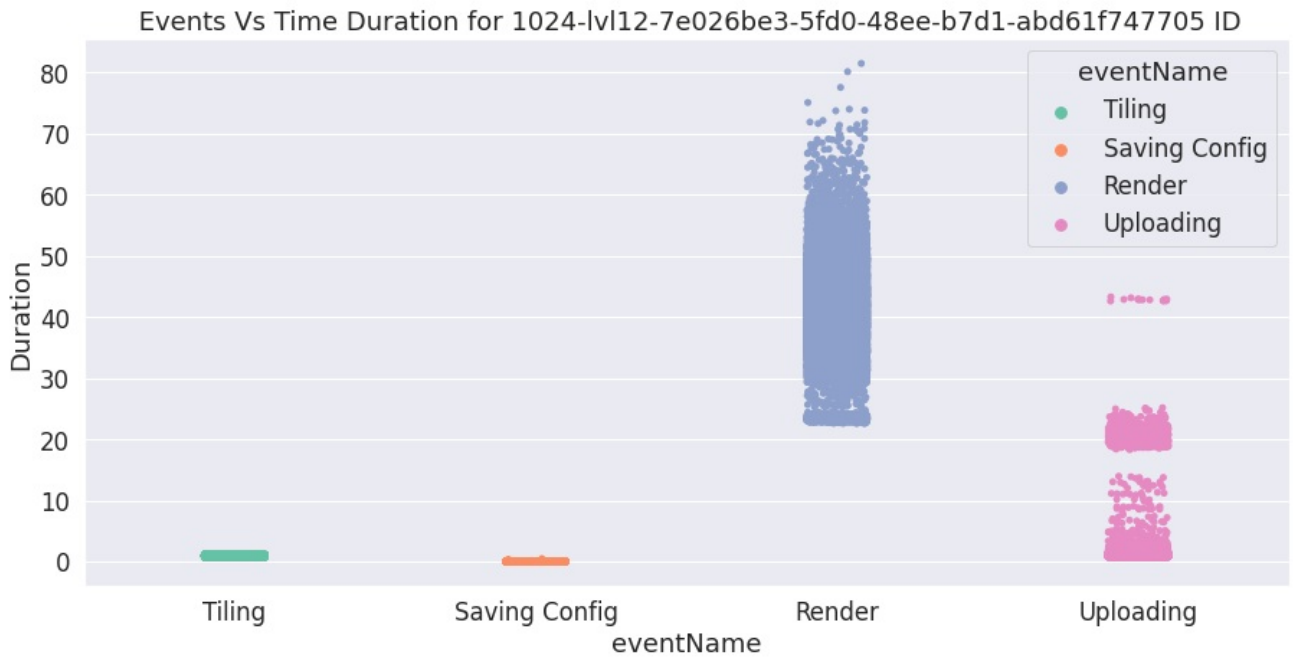


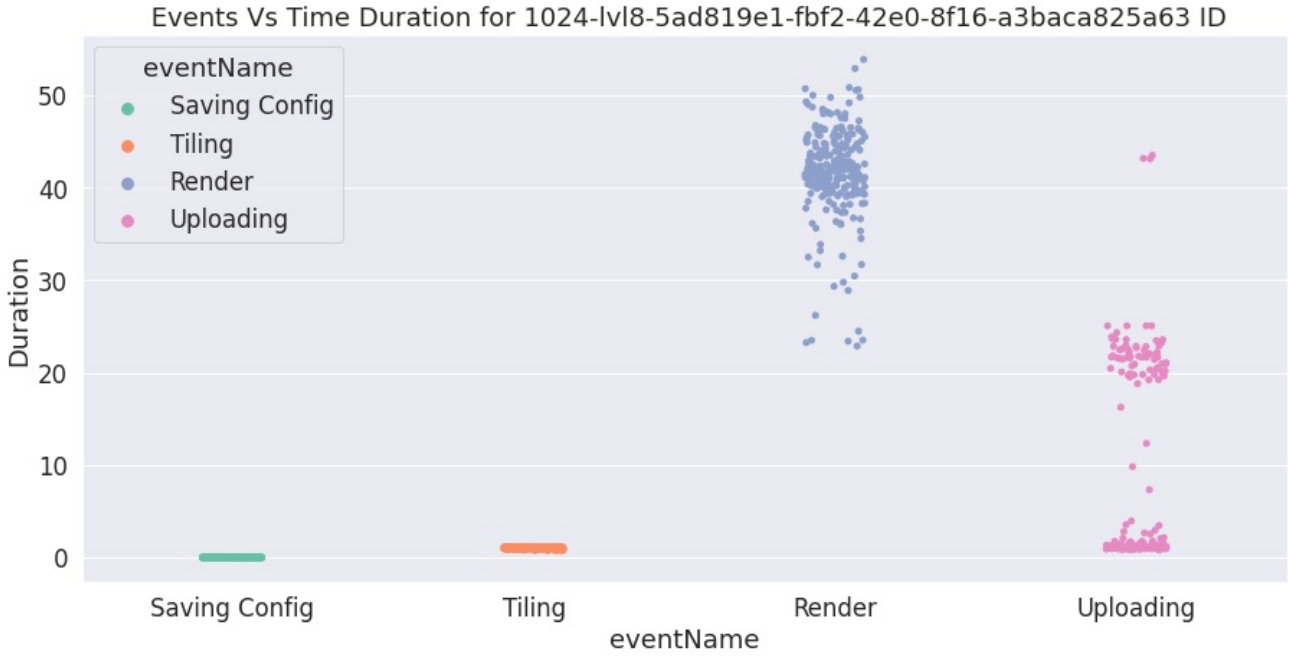
From The above box plot following conclusion can be drawn : Saving configuration, Uploading and Rendering events don't take much

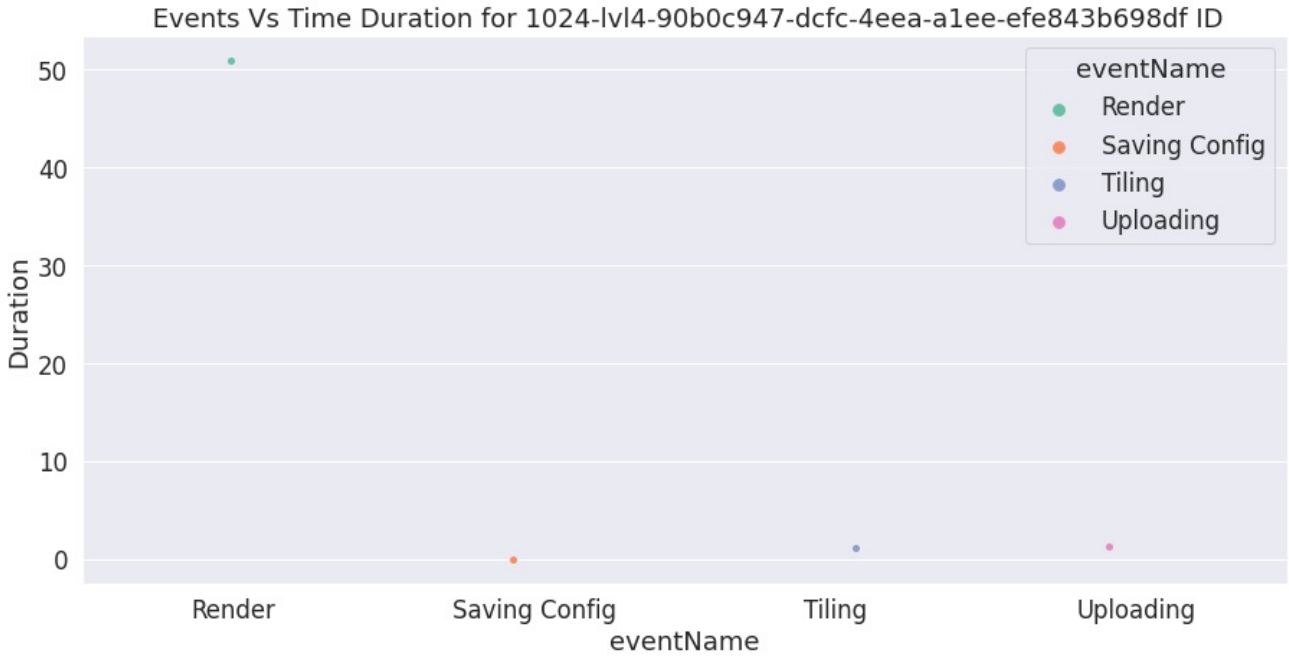
- Saving configuration, Uploading and Rendering events don't take much computation time.
- Rendering event dominates the task execution time of all the events which is nearly equal to total render event which represents the whole length of the task run time. There is also a considerable number of outliers in both events.
- On an average the rendering event takes about 43 seconds with an interquartile range 38 to 45 seconds approximately
- Uploading event has a large number of outliers.

From our intial analysis we already know that there are 3 unique Azure jobs ids present in the application-checkpoint dataset.We will try to analysise which of the event types dominate the task run time in these Azure jobs.

```
UNIQUE AZURE JOBID's
['1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705'
 '1024-lvl8-5ad819e1-fbf2-42e0-8f16-a3baca825a63'
 '1024-lvl4-90b0c947-dcfc-4eea-a1ee-efe843b698df']
```

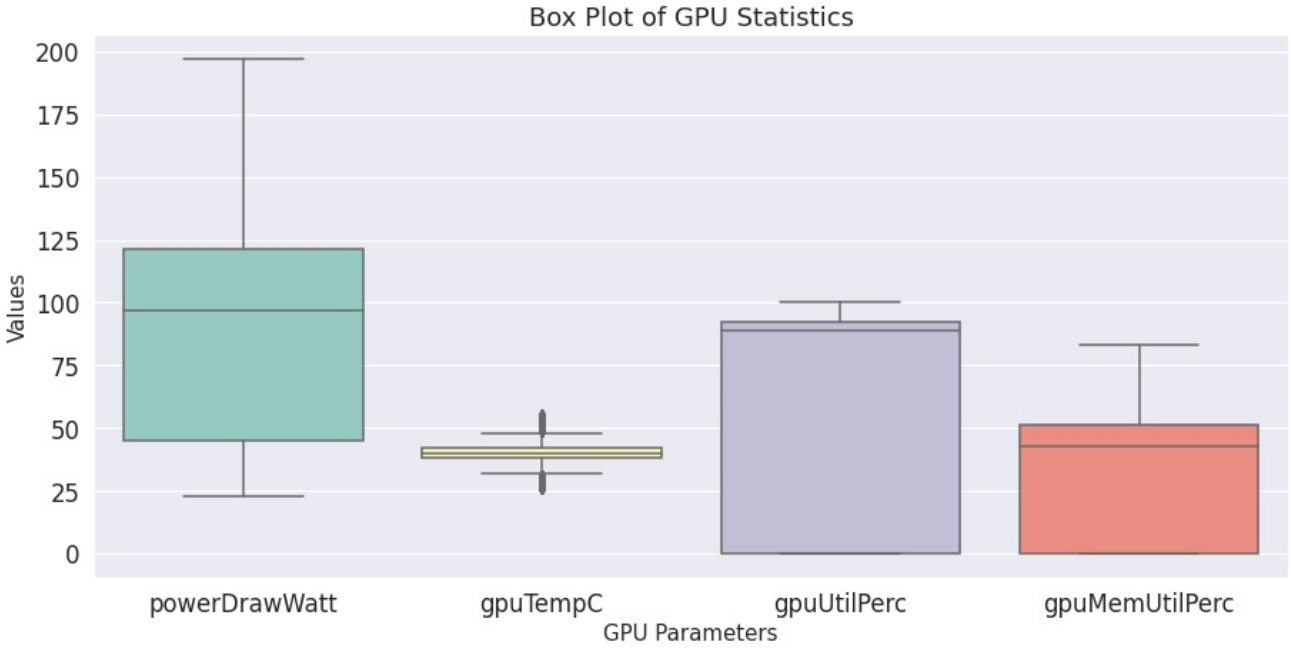






From all of the above plots for each of the unique Azure ids, we can further conclude that rendering event consumes more time than any other even followed by uploading event.

GPU Parameters Analysis



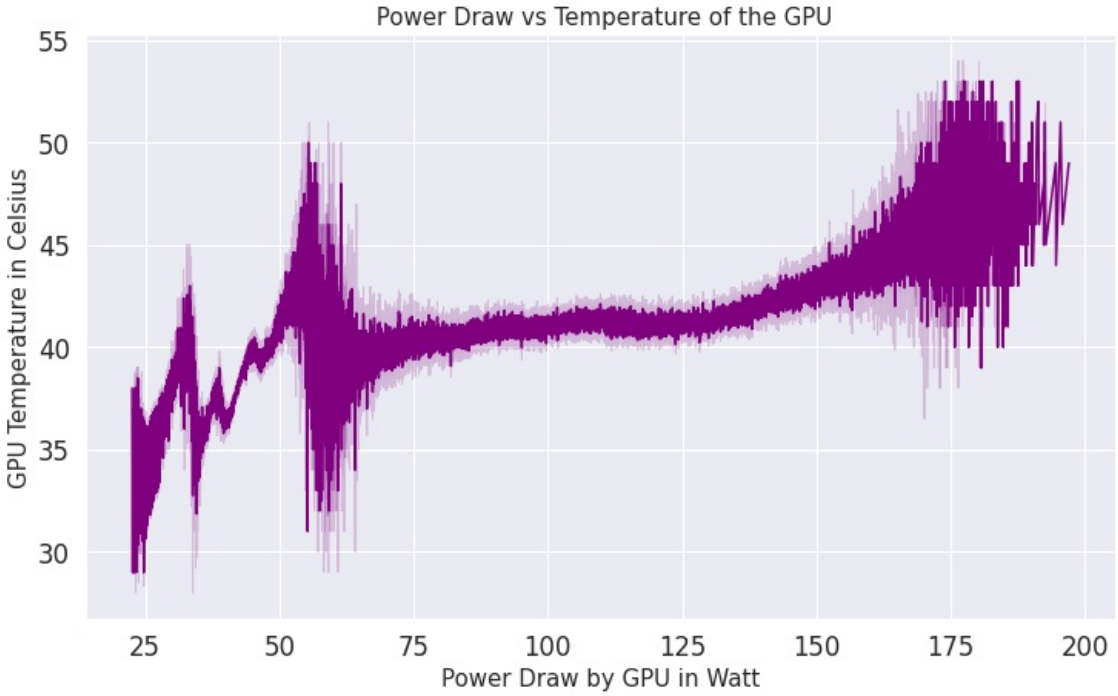
Numerical Analysis of GPU Parameters

	powerDrawWatt	gpuTempC	gpuUtilPerc	gpuMemUtilPerc
count	1.543681e+06	1.543681e+06	1.543681e+06	1.543681e+06
mean	8.919838e+01	4.007560e+01	6.305820e+01	3.341359e+01
std	3.975742e+01	3.800243e+00	4.144816e+01	2.300107e+01
min	2.255000e+01	2.600000e+01	0.000000e+00	0.000000e+00
25%	4.499000e+01	3.800000e+01	0.000000e+00	0.000000e+00
50%	9.659000e+01	4.000000e+01	8.900000e+01	4.300000e+01
75%	1.213400e+02	4.200000e+01	9.200000e+01	5.100000e+01
max	1.970100e+02	5.500000e+01	1.000000e+02	8.300000e+01

The above box plot and numerical analysis for different GPU parameters infer the following:

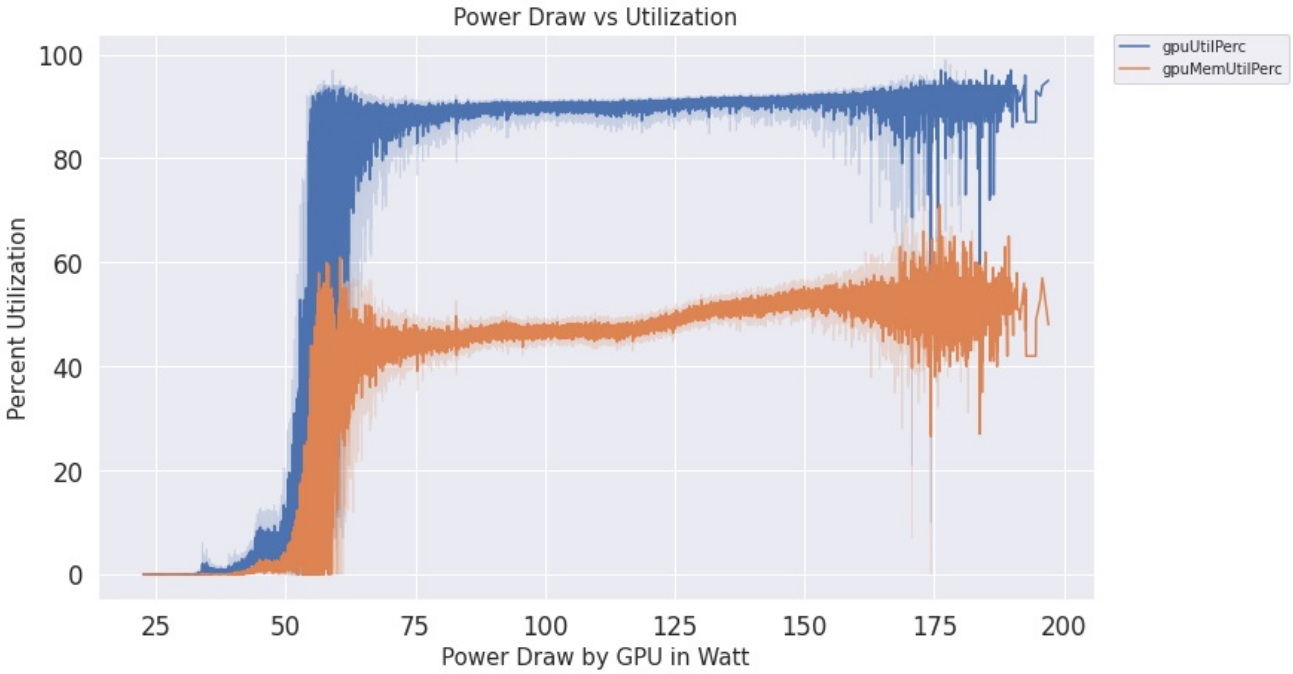
- Average GPU utility percentage is about 85-90 percent. We may assume that almost all of the GPU cores are fully utilized, however due to wide interquartile range we can explore ways to improve the efficiency.
- GPU power drawn is in the range of 45 to 121 W with a minimum and maximum of 23w and 200w respectively.
- GPU temperature is between 30 to 60 degrees Celsius for most part of the operating conditions. We can say that the systems are cooled well and there will not be any temperature throttling.
- Memory utilization Is about 50 percent with a small interquartile range , which suggests there is scope to optimize the memory usage.

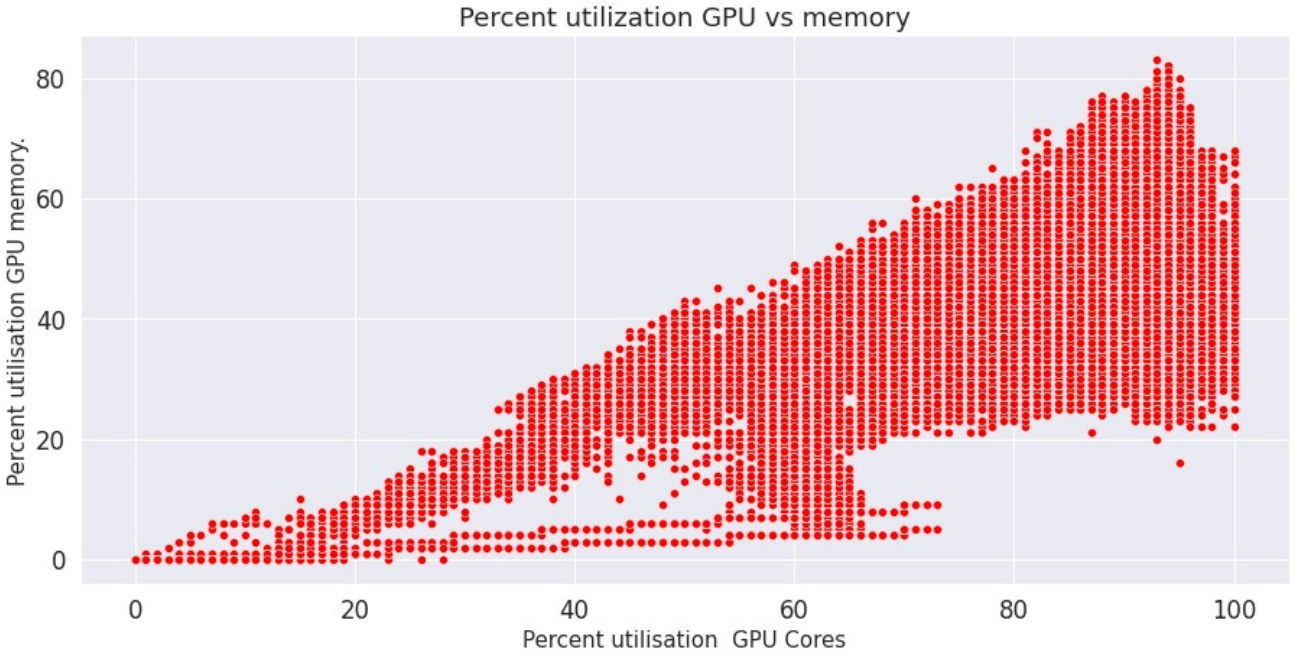
Inter Play between GPU Performance and Temperature



From the above plot we can perceive the following:

- There is an increase in GPU temperature with the increase in the GPU temperature as well.
- At around 60 Watt there is a spike in the temperature, so there may be some low performing GPUs where the temperature is increased even though the power drawn is less.
- There is no confirmation that as the temperature increases there is significant increase in performance or power drawn. Further analysis on GPU utilization and memory utilization would be required





From the above plots we find that:

- There is an increase in both GPU utilization and GPU Memory as the power drawn by the GPU increases which is as expected because the GPU components utilization increases as the performance increases in turn the power consumption also increases. So, we can deduce that increase in power drawn by the GPU increases the performance
- Performance increase of the GPU after 60w is very marginal. As the power drawn is more there is a fluctuation in GPU utilization and memory utilization which confirms that the GPU is throttled. This can be because of some of the weaker GPUs or may be because of temperature or clock speed throttling.
- we can clearly see that the utilization of the memory linearly increases with GPU core utilization percent increases .so the GPU cores and memory utilization is highly corelated which is as expected .
- All the above conclusions made can also be verified from the pair plot as well.

Corelation between Render Time and Power Drawn

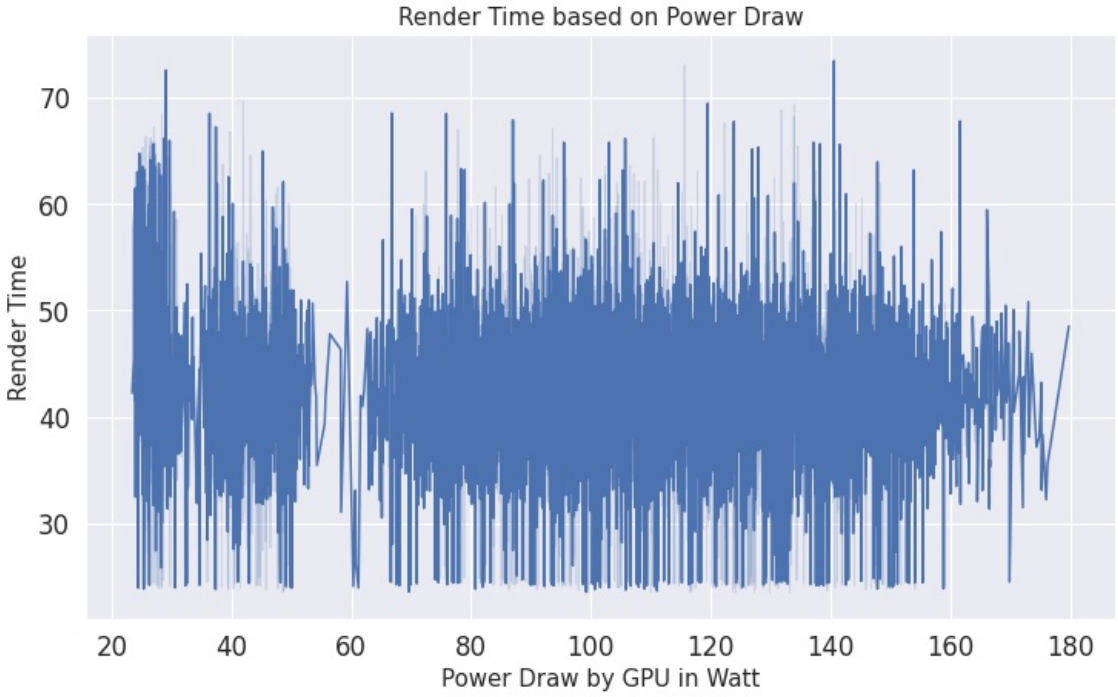


Fig 7: GPU temperature vs Render time and Power Draw

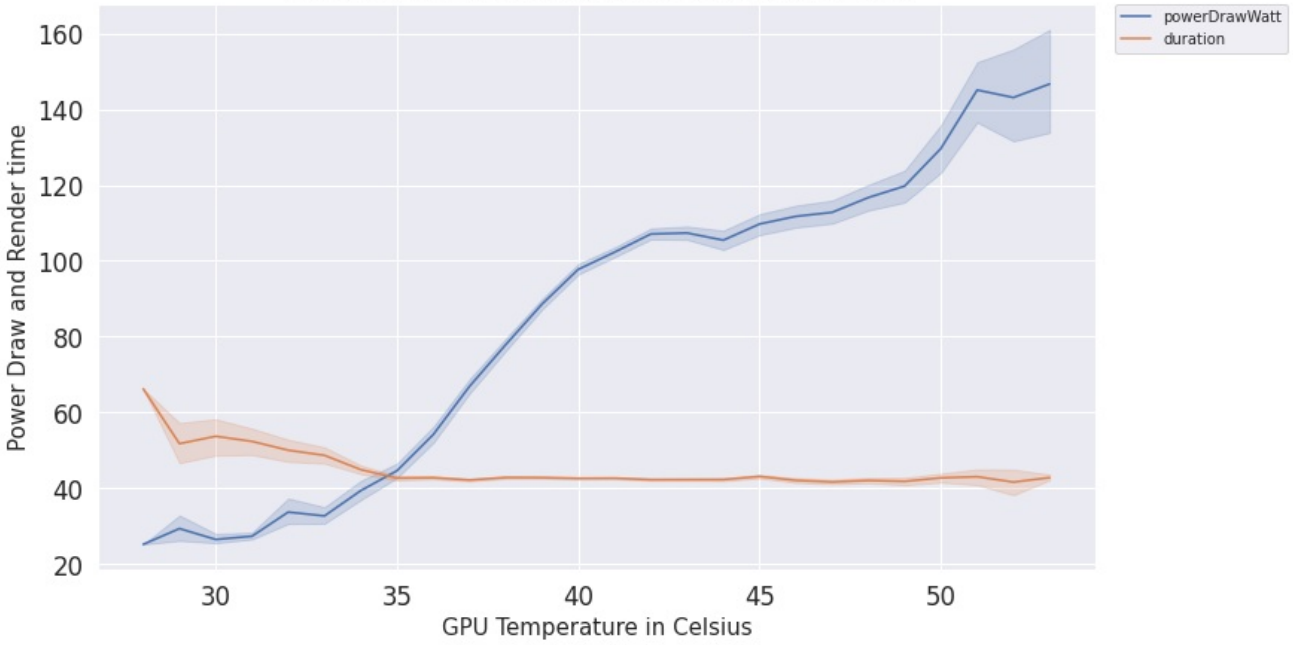
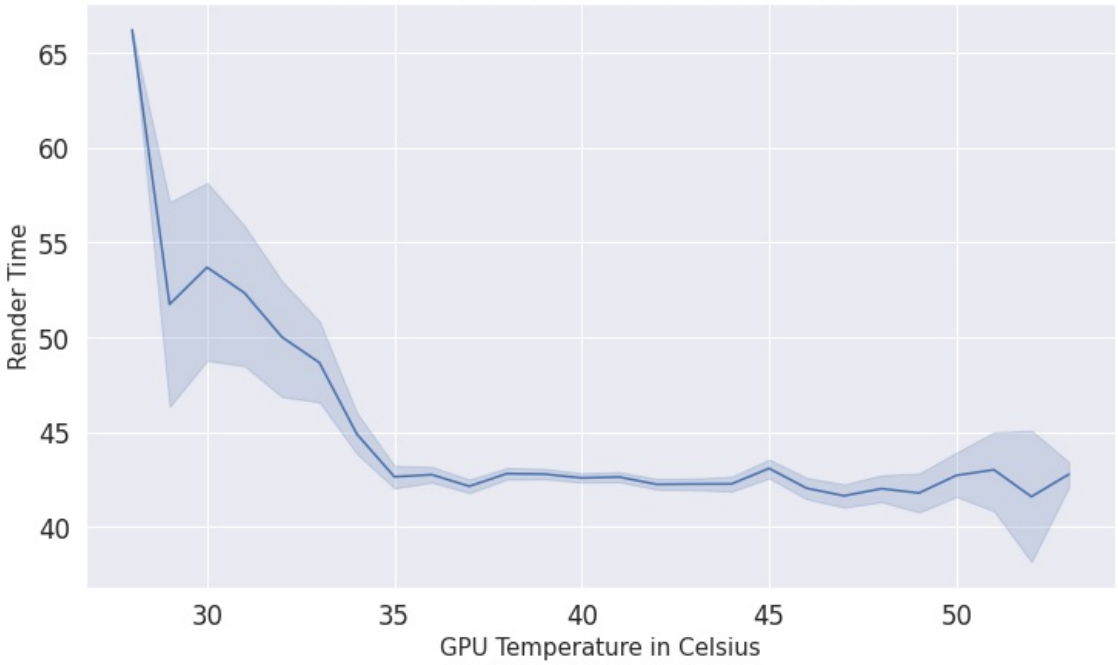


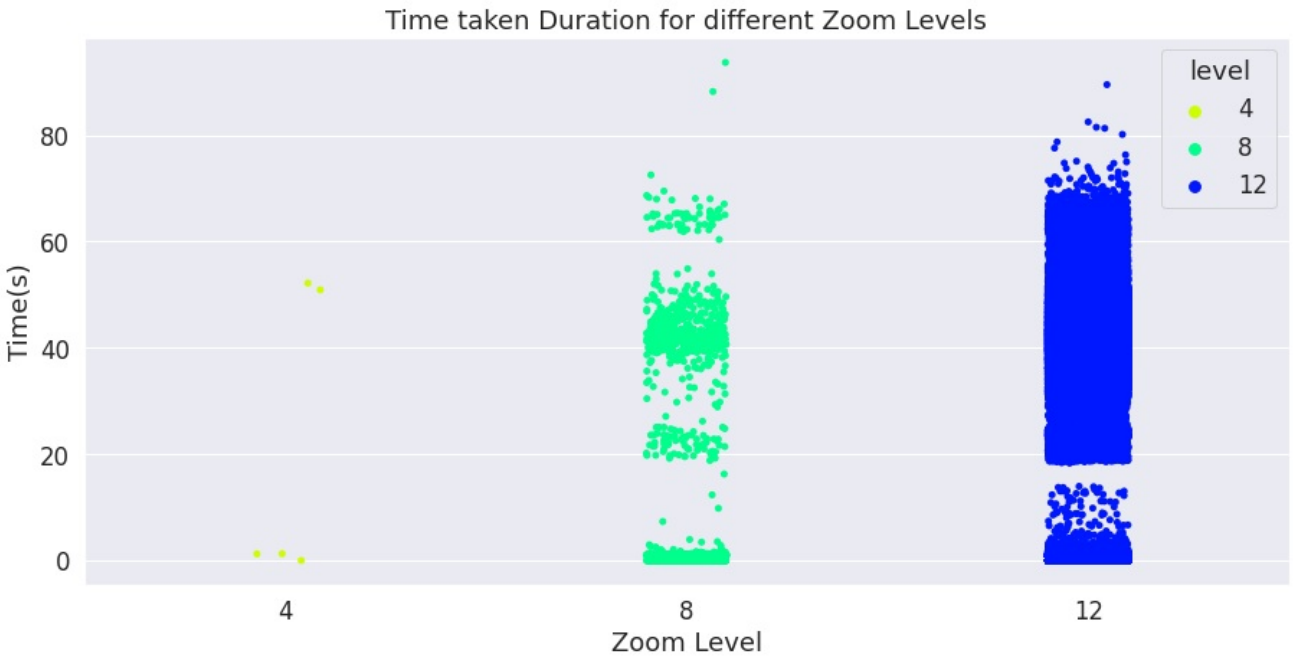
Fig 8: Render Time based on GPU Temperature



From the above plots we can conclude the following:

- There is no improvement in render time as the power drawn by the GPUs increases.
- As temperature increases, we see that power drawn is increased and render time is decreased hence, we can say increase in temperature does not affect the performance much.
- Render time is decreasing drastically when temperature is approaching 35 degrees Celsius and is consistent thereafter.
- Overall, the power draw by the GPU doesn't have any corelation with the render time.

Computation Requirements for Different Tiles



Numerical summary of Zoom Level 4

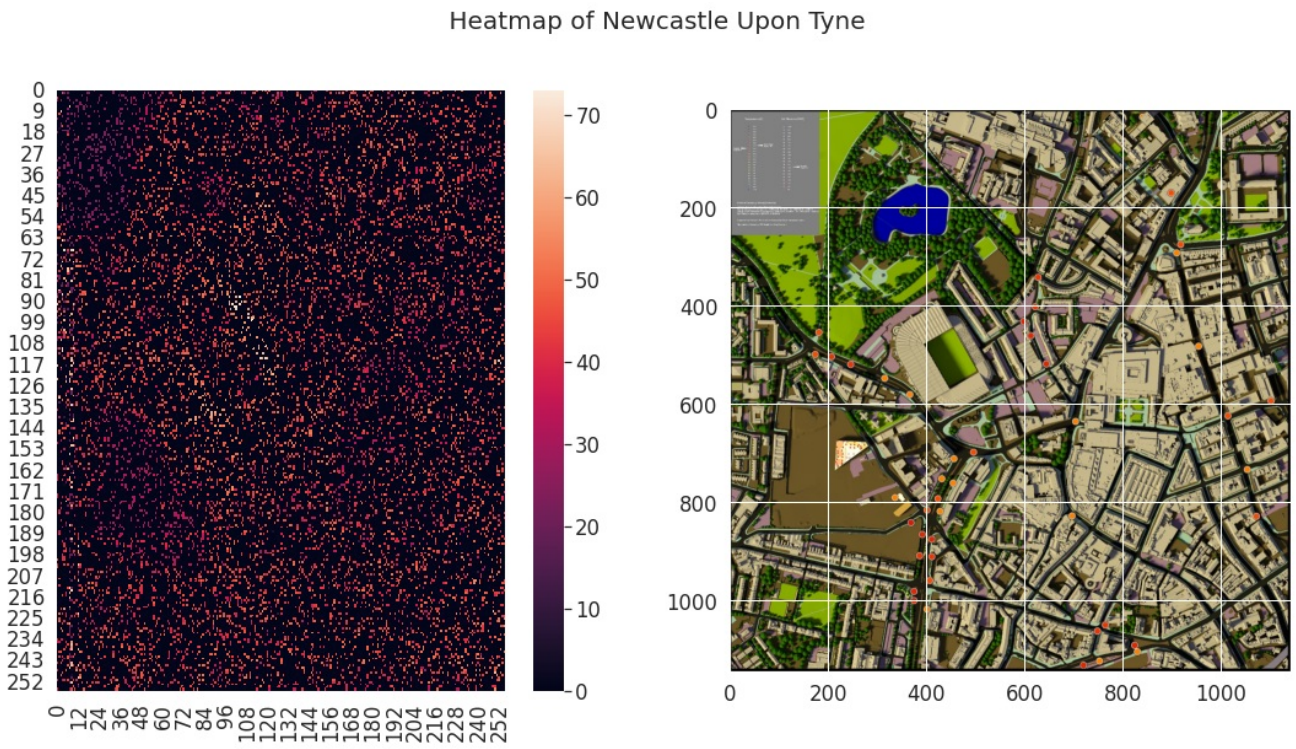
	x	y	level	Duration
count	5.0	5.0	5.0	5.000000
mean	0.0	0.0	4.0	21.122000
std	0.0	0.0	0.0	27.782006
min	0.0	0.0	4.0	0.002000
25%	0.0	0.0	4.0	1.246000
50%	0.0	0.0	4.0	1.269000
75%	0.0	0.0	4.0	50.911000
max	0.0	0.0	4.0	52.182000

Numerical summary of Zoom Level 8

	x	y	level	Duration
count	1310.000000	1310.000000	1310.0	1310.000000
mean	7.454198	7.534351	8.0	19.632202
std	4.598863	4.653457	0.0	22.159412
min	0.000000	0.000000	8.0	0.002000
25%	3.000000	3.000000	8.0	0.949000
50%	7.500000	7.500000	8.0	1.130000
75%	11.000000	12.000000	8.0	42.058250
max	15.000000	15.000000	8.0	93.697000

Numerical summary of Zoom Level 12

	x	y	level	Duration
count	331355.000000	331355.000000	331355.0	331355.000000
mean	127.471141	127.474355	12.0	17.234791
std	73.896956	73.896938	0.0	20.575444
min	0.000000	0.000000	12.0	0.002000
25%	64.000000	63.000000	12.0	0.902000
50%	127.000000	128.000000	12.0	1.066000
75%	191.000000	191.000000	12.0	40.742000
max	255.000000	255.000000	12.0	89.525000



From the above numerical and graphical analysis, we can infer the following:

- Level 12 zoom dominates the task run time. While level 4 and 8 has very less impact on the run time as the rendering is required less on these levels. Level 12 rendering is the inner most zoomed layer of the image.
- From the heat map there is not much pattern that we can identify. we see that the purple pixels where the detailing is less, and the lighter pixels are areas where rendering takes time which has more details.
- 600 to 700 is the range of the tasks assigned to majority of GPUs with negligible amount around 1200. So, there is no consistent scheduling and there could be improvements to ensure uniform spread of load.

Performance of GPUs based on their serial number.

Low Performing GPUs

	gpuSerial	powerDrawWatt
867	325117171574	80.510313
679	325017017790	80.742673
704	325017018552	81.087075
800	325017049295	81.226755
619	324917052619	81.242179

```
High Performing GPUs
```

	gpuSerial	powerDrawWatt
100	323217056123	98.698678
794	325017049041	99.057575
130	323217056368	101.549633
405	323617021202	101.974324
491	323617042596	106.247462

From the above two tables we can see the high and low performing GPUs based on their serial numbers. There is no specific pattern of their performance based on serial numbers. On an average GPU with serial number **323617042596** tends to perform better than the rest.

7. Results & Conclusion

Based on analysis from data mining and exploratory data analysis using the CRISP-DM methodology we can draw the following conclusion.

The task runtime is dominated by the rendering event when compared to other events (tiling, uploading, saving configuration). There is no implication of GPU temperature and the time taken to render while there is a slight correlation between the power drawn and the time duration taken to render. Lower zoom levels result in higher computational requirements i.e., tiles with dense area take more rendering time compared with tiles with empty spaces. Allocation of high performing GPUs appropriately with optimize the cloud infrastructure utilization and reduce the cost.

Overall, the project was moderately successful with scope for further analysis and improvements.

8. Future Scope

Possible extension of this project is a scalability planning that could use the information gain from this project as a consideration. Future implications on the work of this area are that with the rapid growth of the current cloud technology, would be able to build and implement machine learning models to allocate GPU resources based on the need of the computational power required.

9. Personal Reflection

Undertaking this project has increased my familiarity of using CRISP-DM helped me in streamlining the whole data mining process. Throughout the project, the most difficult process to be executed perfectly is the data wrangling process in which we need to redo the wrangling process until an appropriate data frame is produced. This project was built using jupyter notebook and I have used libraries such as Pandas, Matplotlib and Seaborn for exploratory data analysis and visualization which helped me learn data analysis techniques and im-prove my python coding skills. I have also explored new modules such as nbconvert in jupyter notebook to work with mark down document which helped me reducing my effort in report writing. Overall, I have enjoyed working on this project overcoming various hurdles along the way which was an experience for future projects.

10. References

[1] Okamoto, Y., Oishi, T. & Ikeuchi, K. Image-Based Network Rendering of Large Meshes for Cloud Computing. Int J Comput Vis 94, 12–22 (2011). https://doi.org/10.1007/s11263-010-0383-1 (https://doi.org/10.1007/s11263-010-0383-1)

[2] Profes-sor Nick Holliman, Manu Antony, Stephen Dowsland, Professor Philip James, Mark Turner, "Petascale Cloud Supercomputing for Terapixel Visualization of a Digital Twin" in arXiv, Online Publication 2019.

[3] Visualizing Urban IoT Using Cloud Supercomputing by Nicolas Holliman, Manu Antony, Stephen Dowsland & Mark Turner