Student ID: 220586610 / c2058661@newcastle.ac.uk

MSc Data Science, Newcastle University

Cloud Computing – CSC8634

Performance Evaluation of Terapixel Rendering in Cloud Computing

Table of Contents

1. Introduction: -

Cloud computing has been utilized for diverse computational requirements, encompassing the creation of a three-dimensional urban model. The present study involved the generation of a 3-D representation of Newcastle Upon Tyne utilizing supercomputers, with a focus on Terapixel rendering. The aforementioned data set will be utilized for our work. The analysis of such data facilitates the production of images with greater precision, efficiency, and cost-effectiveness. The utilization of the Crisp-DM model has been employed in this study to achieve a similar objective. The present study entails the regular examination of data, integration of data sets, data refinement, and identification of patterns in GPU performance through diverse techniques.

2. Business Understanding: -

During the business understanding phase, the main objective is to gain a comprehensive understanding of the project's requirements from a business perspective. The previous information holds paramount importance for the project, as it empowers organizations to comprehend the potential and constraints of their cloud-based high-performance computing infrastructure in terms of managing image rendering tasks on a massive scale.

Through the acquisition of this comprehension, entities can arrive at judicious determinations about the employment of cloud computing assets for extensive image rendering purposes. The decision to invest in additional resources or optimize existing ones can be determined by them. Moreover, this comprehension aids organizations in recognizing hindrances in their operational processes and implementing requisite modifications to augment efficacy and cost-effectiveness.

Furthermore, the assessment of performance aims to contrast the efficiency of different cloud providers or configurations. This comparative analysis facilitates the process of identifying the optimal solution that meets the unique needs of organizations. This process essentially streamlines the optimization of cloud-based supercomputing infrastructure to effectively manage large-scale image rendering tasks, facilitates informed decision-making regarding cloud resource utilization, and enhances overall performance and cost-efficiency.

Through this analysis, we will gain insights into the following:

- Which event type dominates task runtimes?

- What is the interplay between GPU temperature and performance?

- What is the interplay between increased power draw and render time?

- Can we identify GPU cards (based on their serial numbers) whose performance differs from other cards? (i.e., perpetually slow cards).

3. Objective: -

This report's objective is to look into trends and bottlenecks in the use of GPUs for terapixel rendering operations in Newcastle Upon Tyne via a cloud computing platform. The anticipated criteria that will be looked at are listed below.

- Relationship between GPU statistics.

- Parallel workload distribution by the cloud architecture.
- Memory Usage.
- Event type runtime pattern and domination.

4. Tools and Methodologies: -

Tools – Python programming language was utilized to ensure reproducibility. A Python notebook was employed as per the instructions provided in the readme file, enabling centralized access and easy sharing of code and analysis.

To mitigate the risk of unintended or erroneous changes, version control was implemented using GitHub. This facilitated effective management of different development paths, providing the ability to revert changes if necessary.

5. Data Understanding: -

Following the phase of Business Understanding, the subsequent stage entails the discernment, accumulation, and scrutiny of pertinent datasets that facilitate the attainment of project goals. In this phase, the requisite data is collected and imported into our analytical tool to derive a comprehensive understanding of the dataset, encompassing its structure, number of records, and other relevant attributes.

The information provided in this report was obtained through an experimental procedure that employed a comprehensive setup consisting of 1024 GPUs. The process of rendering was partitioned into 3 discrete tiers, namely levels 4, 8, and 12, which correspond to the final output. The dataset provides comprehensive insights into diverse facets, including the temporal aspects of rendering performance, graphics processing unit (GPU) performance, and the particular image regions processed by each task.

Our data consist of three files:

- *application-checkpoint.csv*
- *gpu.csv*
- *task-x-y.csv*

- **application-checkpoint.csv:** This file includes information on various events which have occurred during the rendering process which include the type of event, the timestamp of when it occurred, and the hostname, taskId, and jobId associated with it.
  - **timestamp:** This column records the specific moment the event began or ended.
  - **hostname:** This column lists the 1024 unique GPUs used in the operation.
  - **eventName:** this column includes 5 different types of events: Tiling, Render, Saving Config, Uploading, and TotalRender.
    - **TotalRender:** this represents the overall task.
    - **Render:** This is when specific tiles of the image are rendered.
    - **Tiling:** This is the post-processing of the rendered tiles.
    - **Saving Config:** is when the specific configurations for each task are saved
    - **Uploading:** This is when the post-processed tiles are uploaded to Azure Blob Storage.
  - **eventType:** This column indicates whether the event is starting or ending.

- **jobId:** This column indicates the specific level of the visualization output (level 12, level 8, and level 4)
- **taskId:** this column provides a unique identification for each set of events, including their start and stops timestamps. There will be n/10 unique tasks in the file, where n is the number of rows in the application-checkpoint.csv.

- **gpu.csv:** This file contains information about the status of the graphics processing units (GPUs) at a specific point in time.

  - **gpuSerial:** This file includes details such as the unique serial number for each of the 1024 GPUs.
  - **gpuUUID:** This is the system-assigned unique identifier for each GPU.
  - **powerDrawWatt:** The amount of power being used by the GPU in watts.
  - **gpuTempC:** The temperature of the GPU in degrees Celsius
  - **gpuUtilPerc:** The percentage of the GPU's cores that are being used.
  - **gpuMemUtilPerc:** The percentage of the GPU's memory that is being utilized.
- **task-x-y.csv:** This file contains information about the specific sections of an image that are being rendered.

  - **x and y:** It include's the x and y coordinates of the tiles being processed.
  - **level:** this is the level of zoom being used. The visualization is designed to be similar to a "google maps style" map, with 12 levels of zoom available. The data only includes levels 4, 8, and 12, as the intermediate levels are created during the post-processing, or tiling, phase.

---application- checkpoints.csv---	
timestamp	object
hostname	object
eventName	object
eventType	object
jobId	object
taskId	object

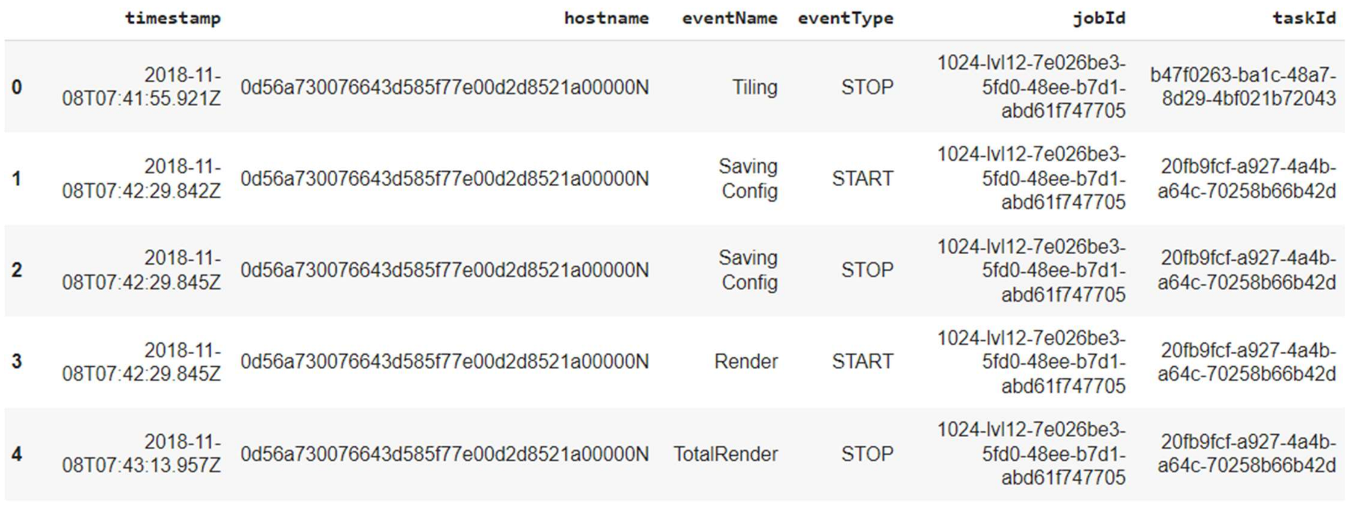
---gpu.csv---	
timestamp	object
hostname	object
gpuSerial	int64
gpuUUID	object
powerDrawWatt	float64
gpuTempC	int64
gpuUtilPerc	int64
gpuMemUtilPerc	int64

task-x-y.csv--	
taskId	object
jobId	object
x	int64
y	int64
level	int64

6. Data Preprocessing: -

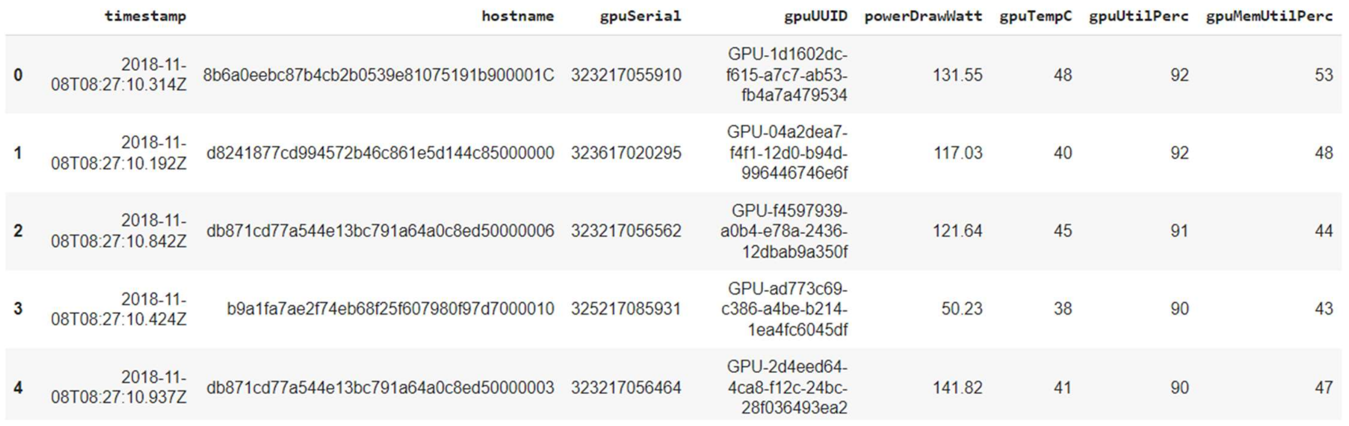
This phase is also called data munging. Here we will now merge the 3 data files into one to make a meaningful dataset.

- The first step is to import the data frames with the name's application_file, gpu_file, and task_xy, respectively, fromthe CSV files application-checkpoints.csv, gpu.csv, and task-x-y.csv into the Python notebook.
- Head of application_file: -



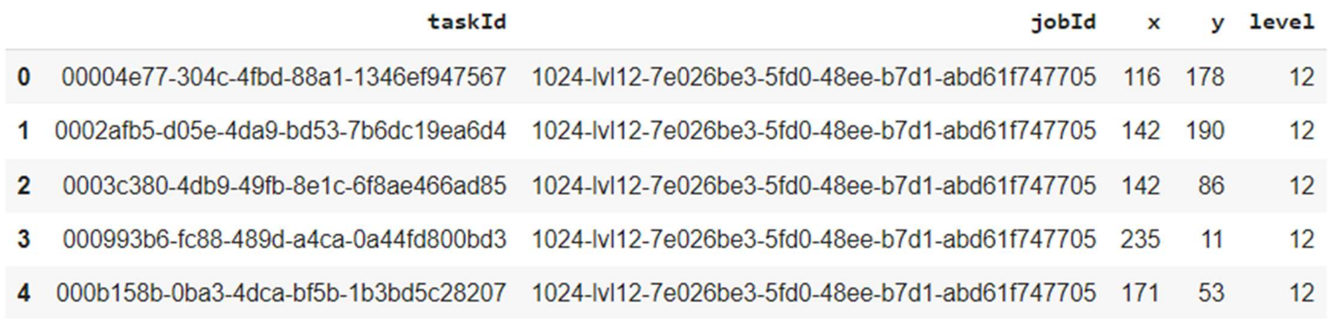
	timestamp	hostname	eventName	eventType	jobId	taskId
0	2018-11-08T07:41:55.921Z	0d56a730076643d585f77e00d2d8521a00000N	Tiling	STOP	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	b47f0263-ba1c-48a7-8d29-4bf021b72043
1	2018-11-08T07:42:29.842Z	0d56a730076643d585f77e00d2d8521a00000N	Saving Config	START	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

- Head of gpu_file:-



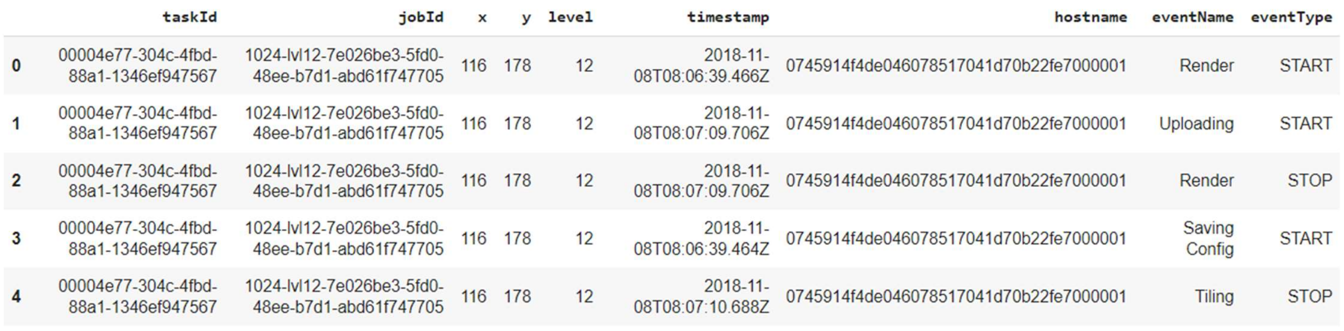
	timestamp	hostname	gpuSerial	gpuUUID	powerDrawWatt	gpuTempC	gpuUtilPerc	gpuMemUtilPerc
0	2018-11-08T08:27:10.314Z	8b6a0eebc87b4cb2b0539e81075191b900001C	323217055910	GPU-1d1602dc-f615-a7c7-ab53-fb4a7a479534	131.55	48	92	53
1	2018-11-08T08:27:10.192Z	d8241877cd994572b46c861e5d144c85000000	323617020295	GPU-04a2dea7-f4f1-12d0-b94d-996446746e6f	117.03	40	92	48
2	2018-11-08T08:27:10.842Z	db871cd77a544e13bc791a64a0c8ed50000006	323217056562	GPU-f4597939-a0b4-e78a-2436-12dbab9a350f	121.64	45	91	44
3	2018-11-08T08:27:10.424Z	b9a1fa7ae2f74eb68f25f607980f97d7000010	325217085931	GPU-ad773c69-c386-a4be-b214-1ea4fc6045df	50.23	38	90	43
4	2018-11-08T08:27:10.937Z	db871cd77a544e13bc791a64a0c8ed50000003	323217056464	GPU-2d4eed64-4ca8-f12c-24bc-28f036493ea2	141.82	41	90	47

- Head of task_xy: -



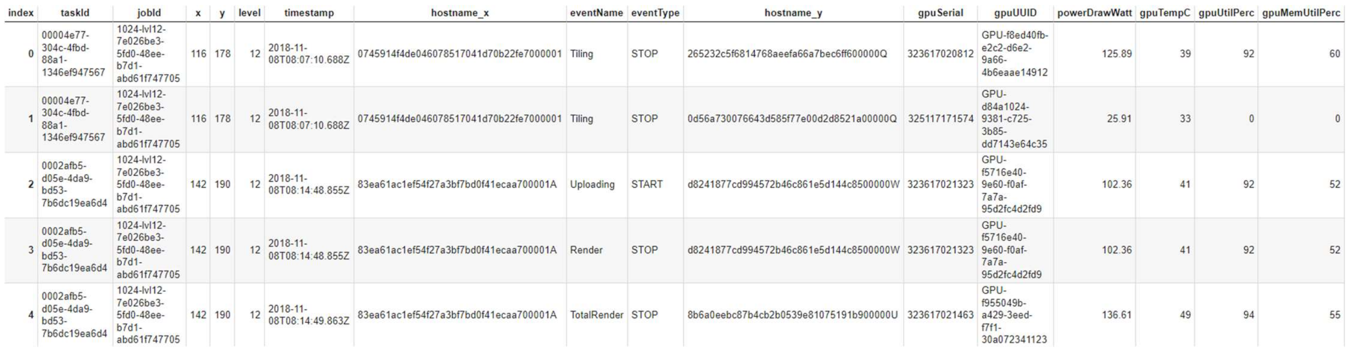
	taskId	jobId	x	y	level
0	00004e77-304c-4fbd-88a1-1346ef947567	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	116	178	12
1	0002afb5-d05e-4da9-bd53-7b6dc19ea6d4	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	142	190	12
2	0003c380-4db9-49fb-8e1c-6f8ae466ad85	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	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

- Firstly, left join the 2 data files *task-x-y.csv* and *application-checkpoint.csv* based on two columns *jobId* and *taskId*.



	taskId	jobId	x	y	level	timestamp	hostname	eventName	eventType
0	00004e77-304c-4fbd-88a1-1346ef947567	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	116	178	12	2018-11-08T08:06:39.466Z	0745914f4de046078517041d70b22fe7000001	Render	START
1	00004e77-304c-4fbd-88a1-1346ef947567	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	116	178	12	2018-11-08T08:07:09.706Z	0745914f4de046078517041d70b22fe7000001	Uploading	START
2	00004e77-304c-4fbd-88a1-1346ef947567	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	116	178	12	2018-11-08T08:07:09.706Z	0745914f4de046078517041d70b22fe7000001	Render	STOP
3	00004e77-304c-4fbd-88a1-1346ef947567	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	116	178	12	2018-11-08T08:06:39.464Z	0745914f4de046078517041d70b22fe7000001	Saving Config	START
4	00004e77-304c-4fbd-88a1-1346ef947567	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	116	178	12	2018-11-08T08:07:10.688Z	0745914f4de046078517041d70b22fe7000001	Tiling	STOP

- Secondly inner join the *gpu.csv* file with the data from the previous step based on column *timestamp*.

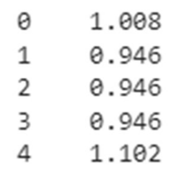


index	taskId	jobId	x	y	level	timestamp	hostname_x	eventName	eventType	hostname_y	gpuSerial	gpuUUID	powerDrawWatt	gpuTempC	gpuUtilPerc	gpuMemUtilPerc
0	00004e77-304c-4fbd-88a1-1346ef947567	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	116	178	12	2018-11-08T08:07:10.688Z	0745914f4de046078517041d70b22fe7000001	Tiling	STOP	265232c5f6814768aeefa66a7bec6ff600000Q	323617020812	GPU-f8ed40fb-e2c2-d6e2-9a66-4b6eaae14912	125.89	39	92	60
1	00004e77-304c-4fbd-88a1-1346ef947567	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	116	178	12	2018-11-08T08:07:10.688Z	0745914f4de046078517041d70b22fe7000001	Tiling	STOP	0d56a730076643d585f77e00d2d8521a00000Q	325117171574	GPU-d84a1024-9381-c725-3b85-dd7143e64c35	25.91	33	0	0
2	0002afb5-d05e-4da9-bd53-7b6dc19ea6d4	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	142	190	12	2018-11-08T08:14:48.855Z	83ea61ac1ef54f27a3bf7bd0f41ecaa700001A	Uploading	START	d8241877cd994572b46c861e5d144c8500000W	323617021323	GPU-f5716e40-9e60-f0af-7a7a-95d2fc4d2fd9	102.36	41	92	52
3	0002afb5-d05e-4da9-bd53-7b6dc19ea6d4	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	142	190	12	2018-11-08T08:14:48.855Z	83ea61ac1ef54f27a3bf7bd0f41ecaa700001A	Render	STOP	d8241877cd994572b46c861e5d144c8500000W	323617021323	GPU-f5716e40-9e60-f0af-7a7a-95d2fc4d2fd9	102.36	41	92	52
4	0002afb5-d05e-4da9-bd53-7b6dc19ea6d4	1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705	142	190	12	2018-11-08T08:14:49.863Z	83ea61ac1ef54f27a3bf7bd0f41ecaa700001A	TotalRender	STOP	8b6a0eebc87b4cb2b0539e81075191b900000U	323617021463	GPU-f955049b-a429-3eed-f7f1-30a072341123	136.61	49	94	55

- **Calculating Duration Time between START and STOP**

The final data file necessitates data cleaning to address the presence of duplicate values and the improper formatting of the timestamp column. Therefore, the elimination of duplicate entries is performed on the ultimate data file. The *datetime* library is utilised to format the timestamp column into a standardised time format, which is subsequently employed for additional calculations.

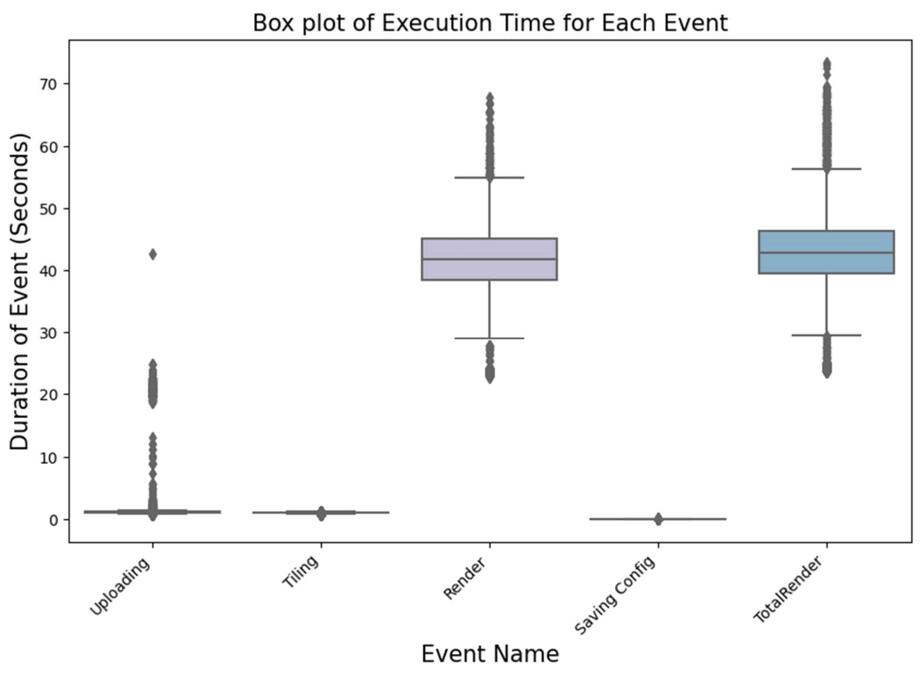
As established during the Business Understanding phase, it is necessary to compute the duration of the events, thus requiring the execution of data wrangling on the timestamp column. Two data frames, namely **df_start,** and **df_stop**, are instantiated. To obtain the *df_start*, the eventType parameter is filtered to retrieve only those instances where its value is 'START'. The timestamp is then extracted and assigned to the *start_time* variable. Similarly, for eventType = 'STOP', the timestamp is extracted and assigned to the *stop_time* variable. The start and stop data frames of the events were merged based on the *taskId*. Subsequently, the duration was computed by subtracting the *start_time* from the *stop_time*, and the resulting value was stored in a column named 'duration'. Now the final merged dataset has the duration time as well for further computations.



```
0    1.008
1    0.946
2    0.946
3    0.946
4    1.102
```

7. Exploratory Data Analysis: -

7.1. Execution Time by Event Name: -



From Fig. 1 provided box plot shows the duration of each event mentioned and it's abundantly evident that GPUs spend themajority of their time rendering, with little time spent saving configuration.

Fig. 1

7.2 Power Drawn v/s GPU Temperature in Watt: -

From Fig 2, we can observe that,

- We can see that the temperatures fluctuate dramatically around 60 watts, which indicates that some GPUs with lower performance are achieving higher temperatures at lower power draw.

- We can see that the temperatures climb as the Power Draw increases.
- We can observe that there is a correlation between the increase in power draw and the temperature rise, although this correlation is not a very large one. As a result of this, we have led to the realization that the GPU's cooling infrastructure is of very high quality.

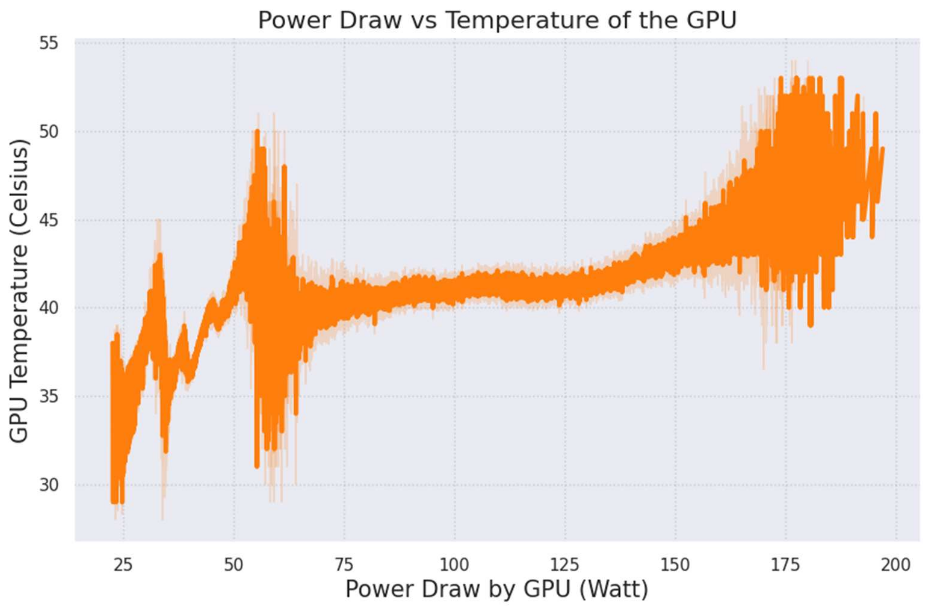


Fig. 2

As this does not depict the complete picture we will dig more into utilization and memory.

7.3 Power Draw v/s GPU Memory Utility: -

The evidence presented in Fig. 3 demonstrates that,

- Even though this graph likewise illustrates the same story, we can see that an increase in power draw results in a greater amount of GPU memory being employed.

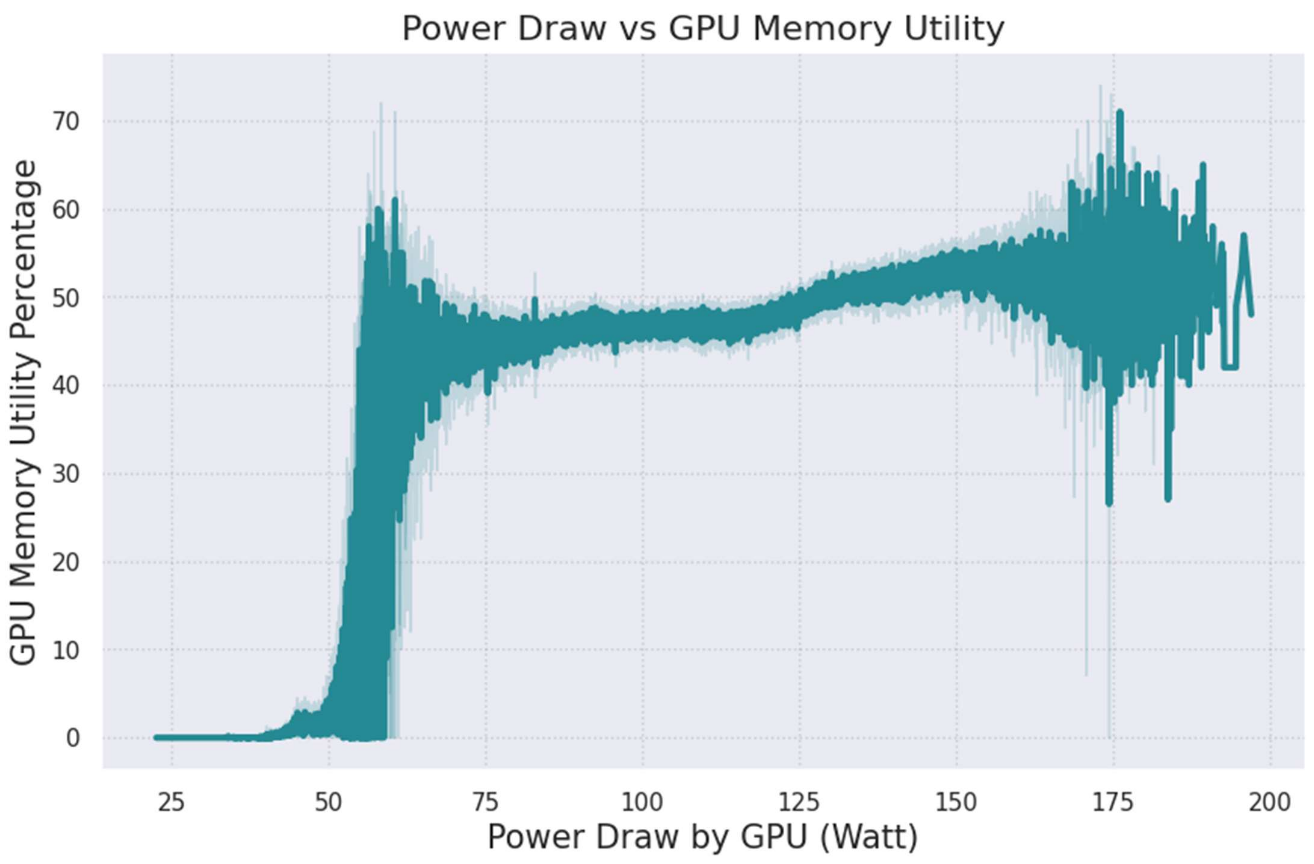
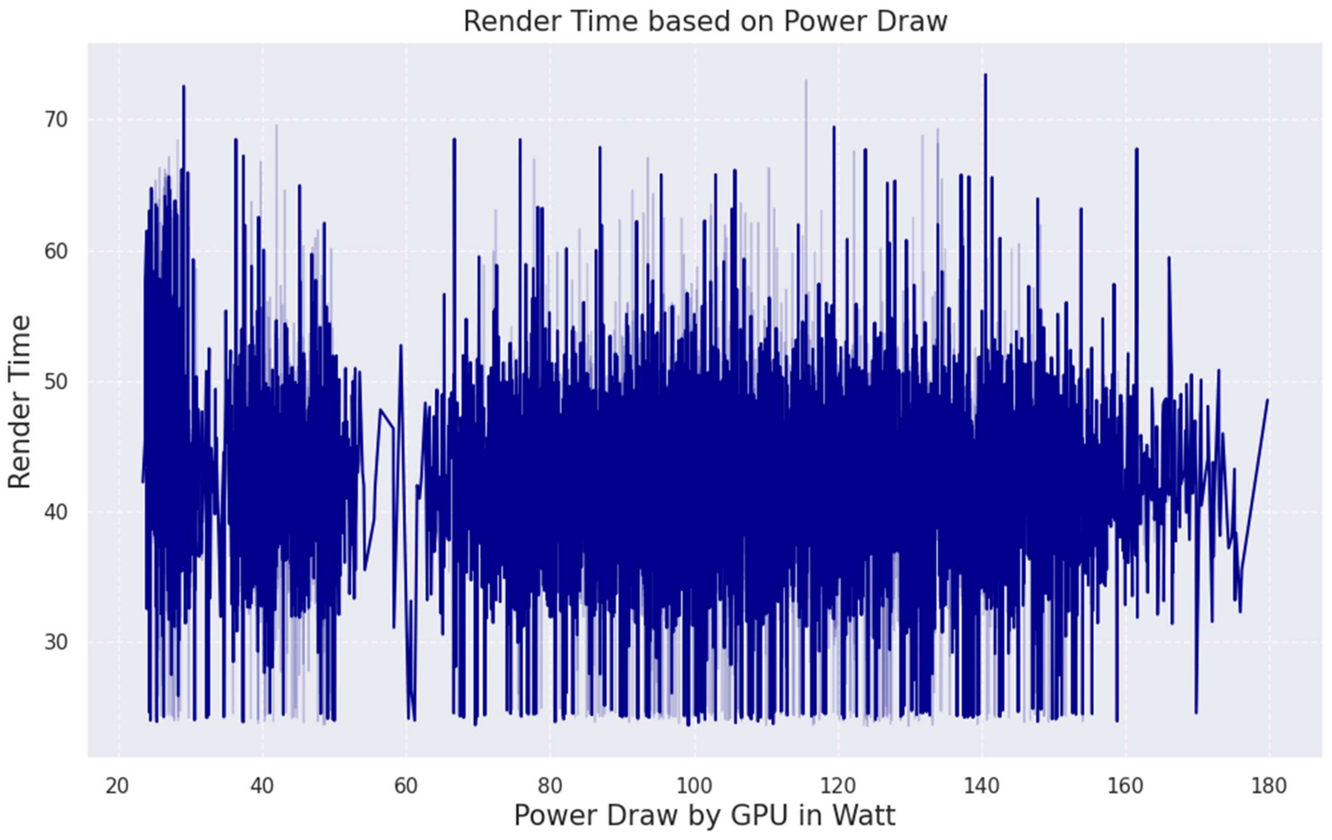


Fig. 3

- Similar to the graph before it, we can see that only a small percentage of GPUs have hit their maximum capacity at 60 watts. Therefore, we can deduce that those GPUs have a poor level of performance.
- Also, as the temperature reaches 160W, we start to witness GPU throttling, which might lead to a loss in performance.

7.4 Tasks assigned per GPU: -

Fig. 4



Given the evidence that is shown in Fig. 4, it is not difficult to deduce that,

- Even though the GPUs are now consuming more power, we have not observed any increase in the length of time required to create the image.
- Even if there has been a major shift in the general demand for power, the length of time it takes to render has not lowered by a significant amount. This is the case even though the overall power demand may have changed.
- On the other hand, although the temperature has increased, there has not been a significant reduction in the amount of time required for rendering.

7.5 Plotting pair subplots to obtain the relationship between GPU variables: -

The below given scatter plot i.e., Fig 5 shows the relationship between GPU statistics. From this plot it is also clear that among all 4 GPU statistics, GPU memory util percentage and GPU utility percentage are highly related.

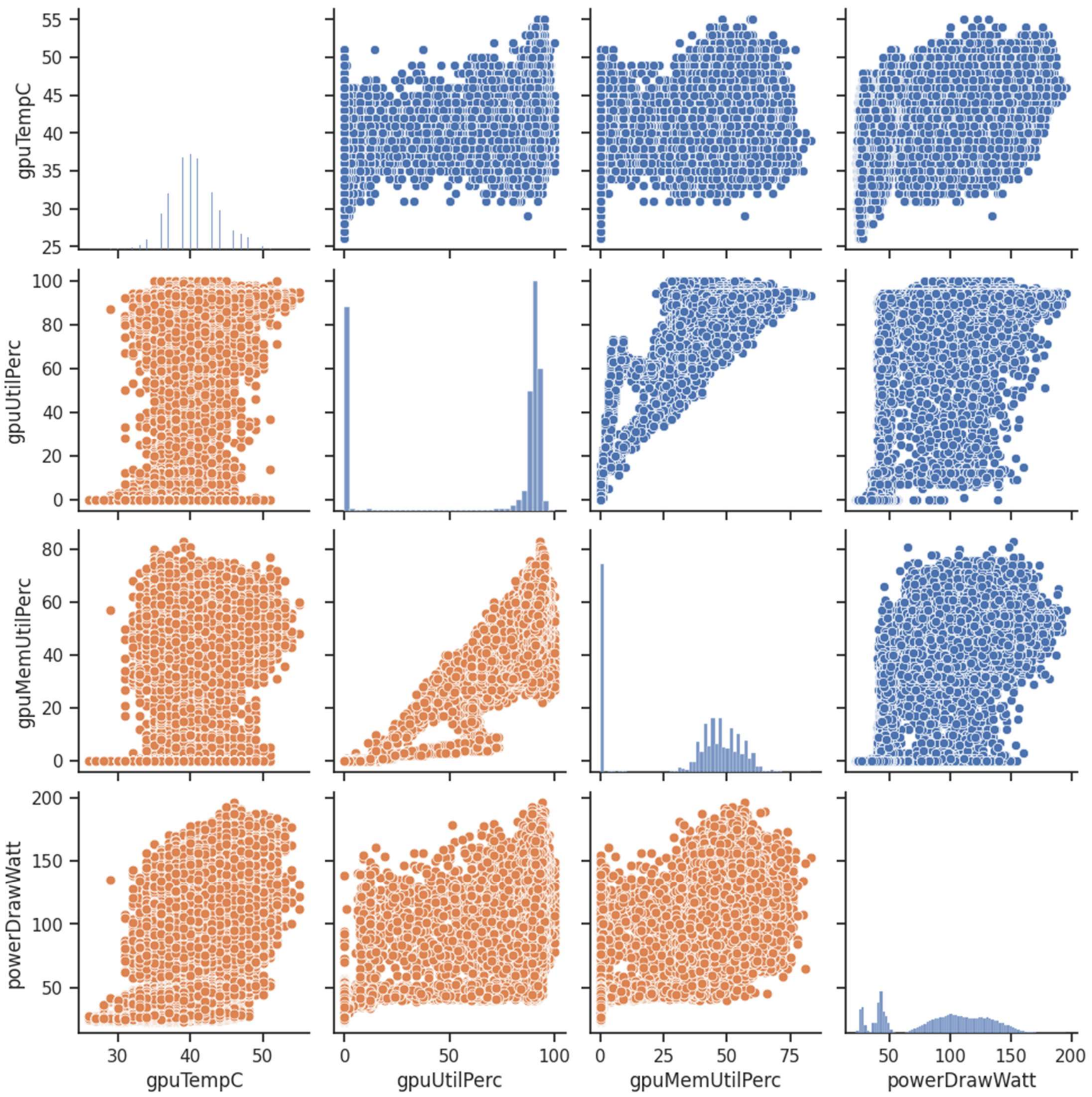


Fig. 5

7.6 Correlation heat map for GPU statistics: -

The correlation coefficient matrix has been plotted below as a heatmap thereby making it easier to visualize the relationship concerning color intensity.

From below Fig 6, it’s clear that power drawn has a strong relationship between GPU Util percentage and GPU memory Util percentage which implies that as power is taken more, the graphics processing unit and memory are used. Also, from this heat map, we can validate Fig 5 as GPU Util percentage and GPU memory Util percentage are highly related. Further, GPU temperature is having a weak relationship with GPU memory until percentage which shows that changing the percentage of memory utility has no discernible impact on the GPU's temperature.

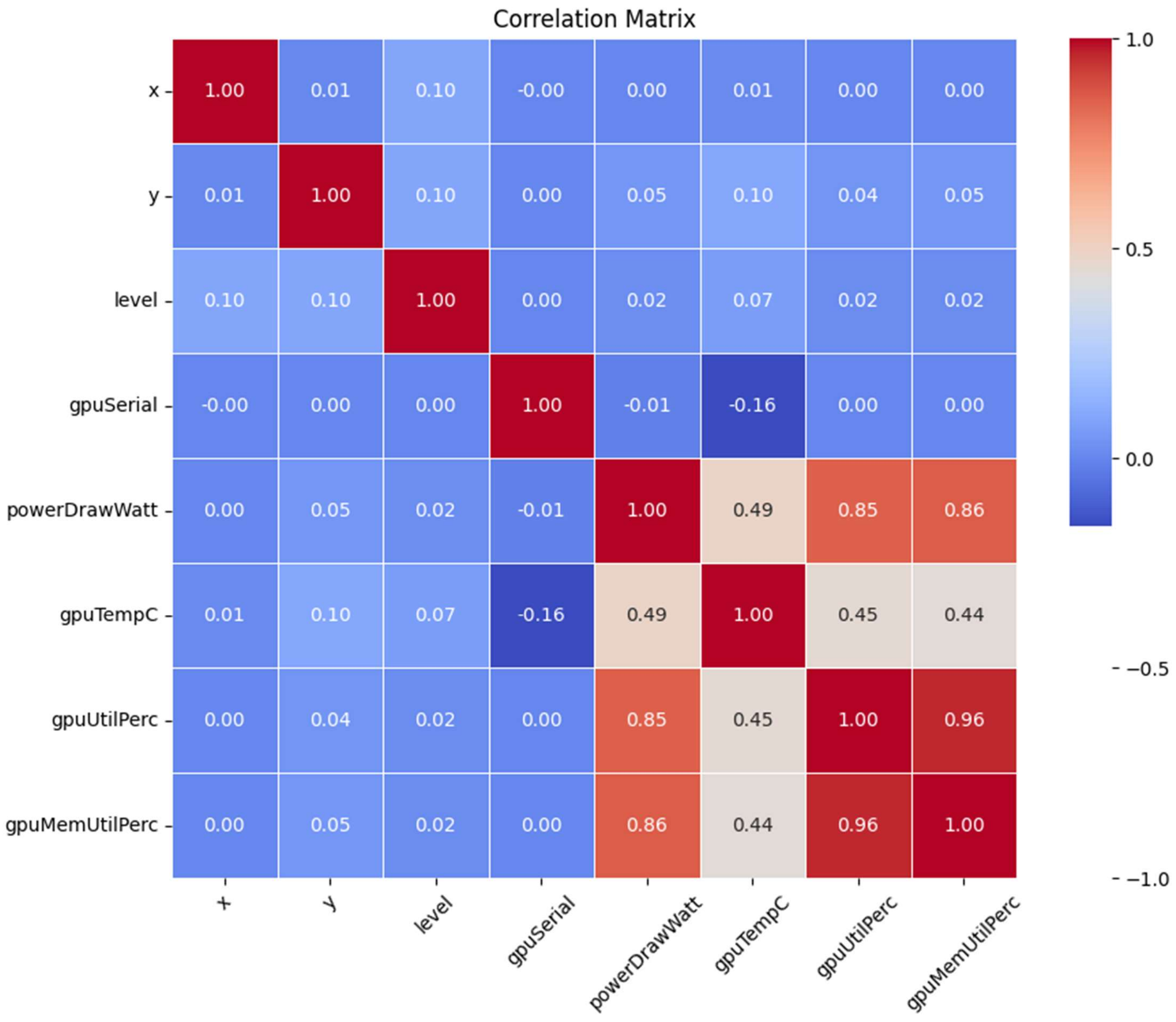


Fig. 6

7.7 Plotting pair subplots between GPU variables and power drawn as hue: -

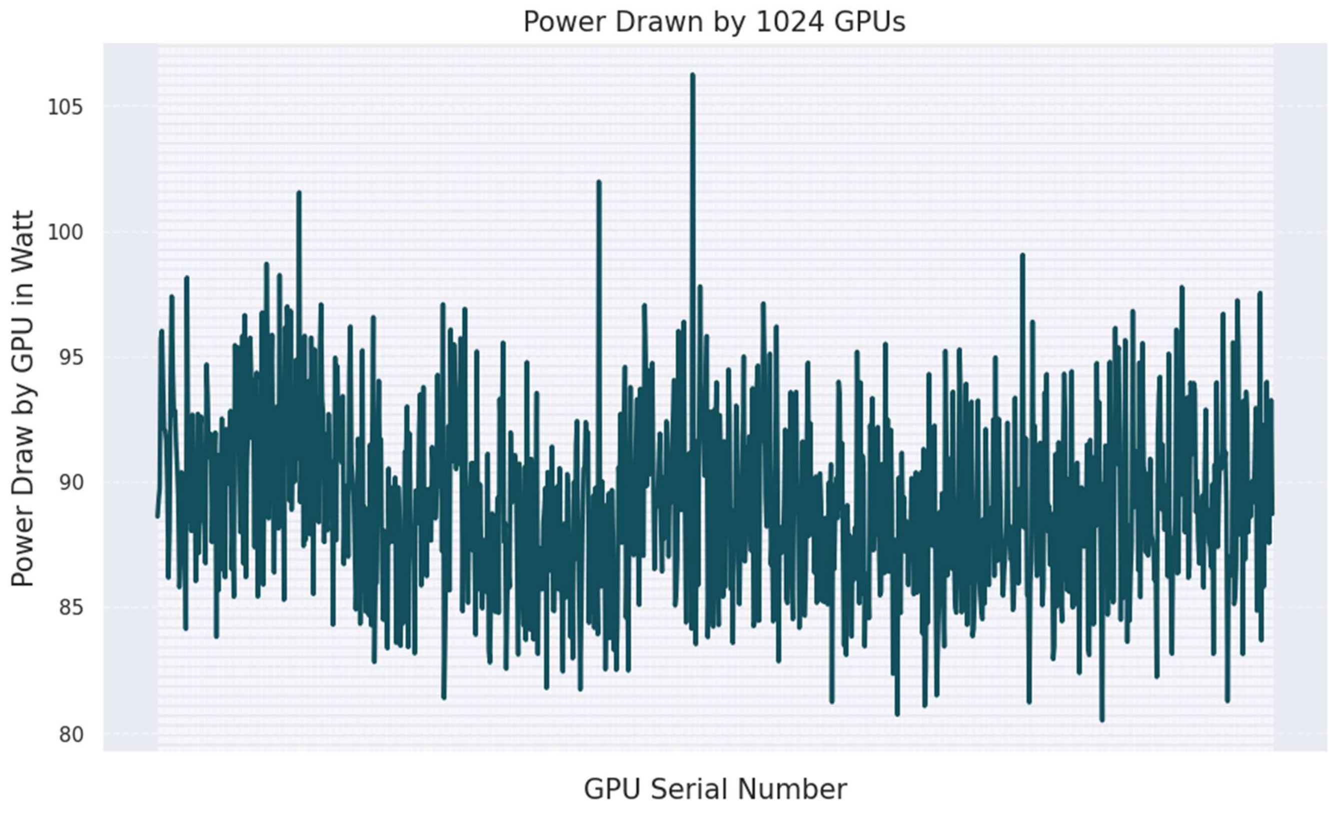


Fig. 7

Fig 7 is the line graph that depicts the power consumption that results from having 1024 GPUs in a system.

- The graph illustrates how the Graphics Processing Units (GPUs) vary in their power consumption, with some GPUs requiring more power than others.
- The results for power draw fluctuate, which indicates that different GPUs have varying requirements for how much power they need.
- The graph makes it possible to locate outliers and gain a better understanding of the system's power usage spectrum.
- The graph does not clearly illustrate the connection between the GPU serial number and the amount of power drawn.
- It is possible to conduct additional research to investigate the factors that influence the power consumption of the GPU system.

7.8 Box plot for GPU statistics: -

The box plot of GPU statistics shown below in Fig 8 shows the various GPU metrics that were recorded while the image was being rendered.

GPU Power Drawn

Power output ranges from 22.5W to 200W, with a typical range of 45 to 121W. These data can help engineers adjust the power supply unit for effective balance in a large-scale application. Additionally, it appears that the GPU utilisation averages at around 90% during the render runs, which is a positive sign that the processing resources are effectively deployed.

GPU Util Percentage

Since the average is between 85 and 90 percent, we can say that most graphics processors are used practically entirely, but because of the wide interquartile range, certain changes are still required for moreeffective handling.

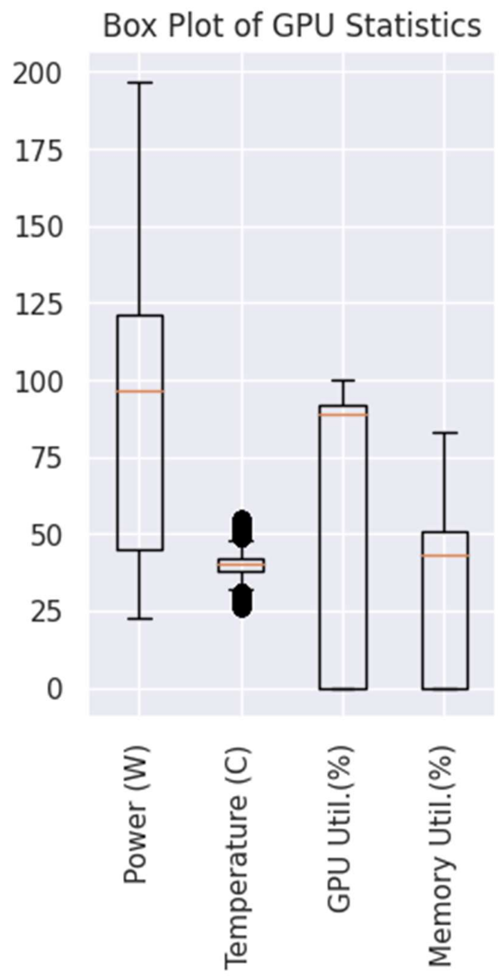


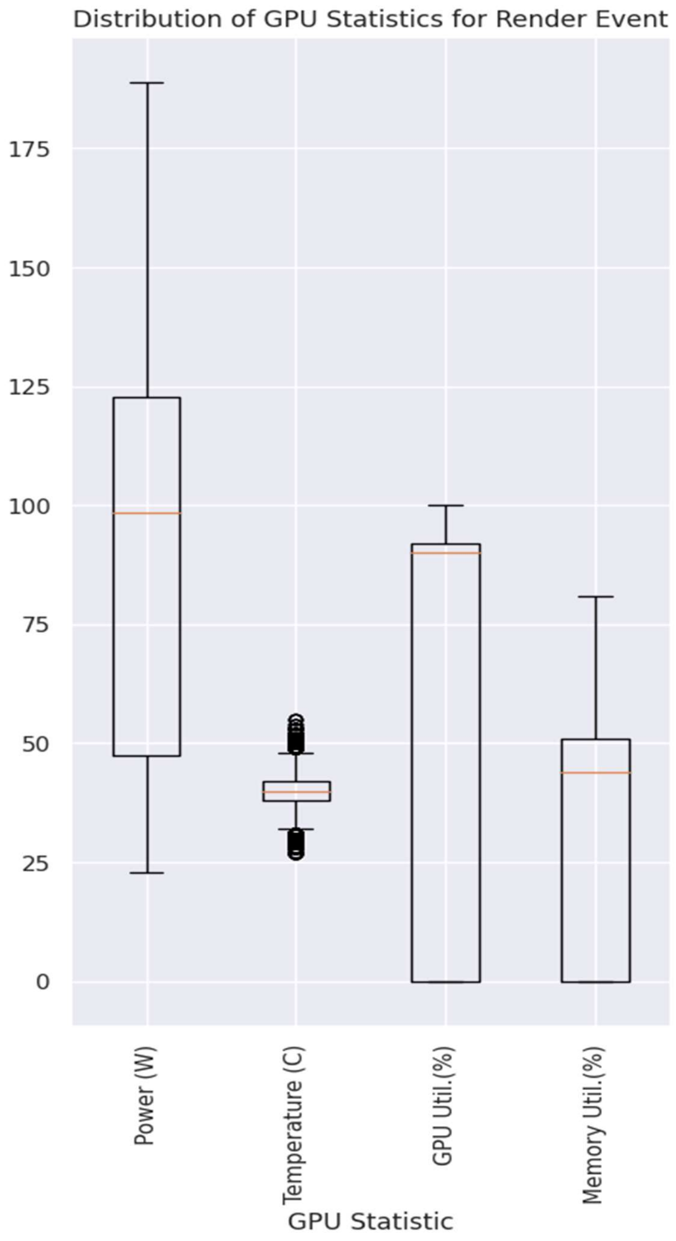
Fig. 8

Memory Utility Percentage

The median is approximately 50%, and the interquartile range is lower and closer, indicating that either the rendering process is not memory intensive or that the task is not well optimized to use the memory.

Another possibility is that GPUs, which use a lot of memory, are to blame for this waste.

7.9 Box Plot for GPU statistics for render event:



-

The box plot of GPU statistics for the render event shown above in Fig 9 shows the various GPU metrics thatwere recorded while the image was being rendered for render events only. The GPU statistics for the render event a are almost similar to the whole as shown in Fig 8.

Fig. 9

7.10 Histogram to visualize the power consumption range: -

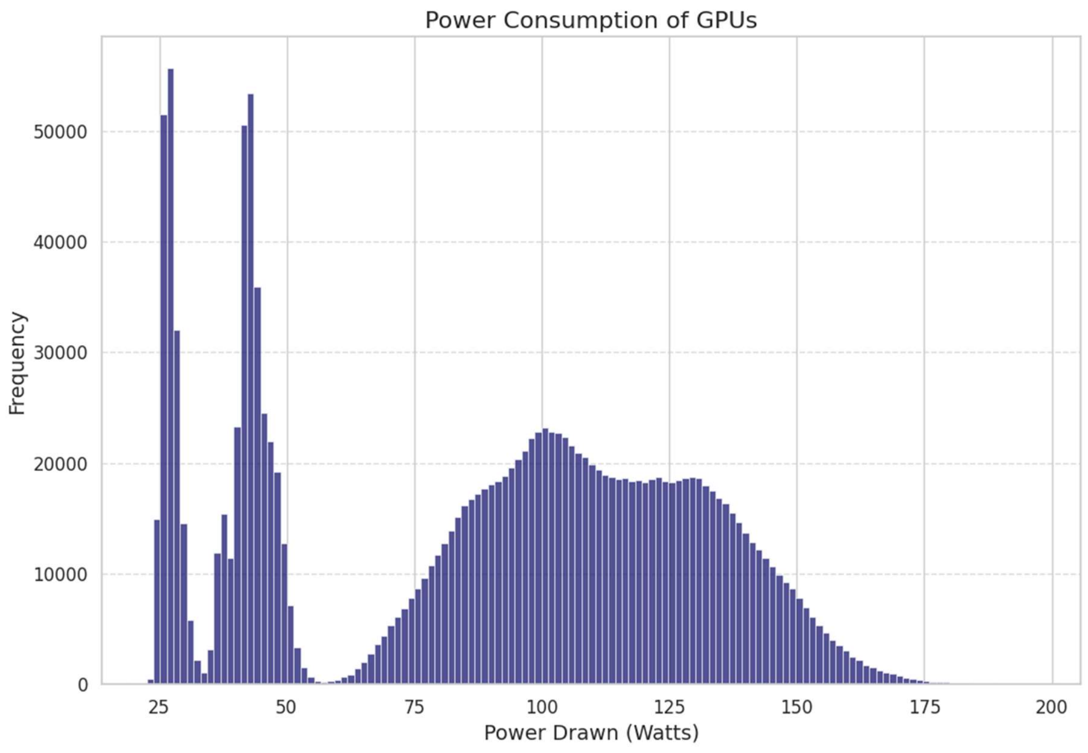


Fig. 10

We can see from the above histogram that the majority of GPU clusters use 20 to 50W of power. The rest use energy in a normal distribution.

7.11 Pair Plot of GPU Conditions: -

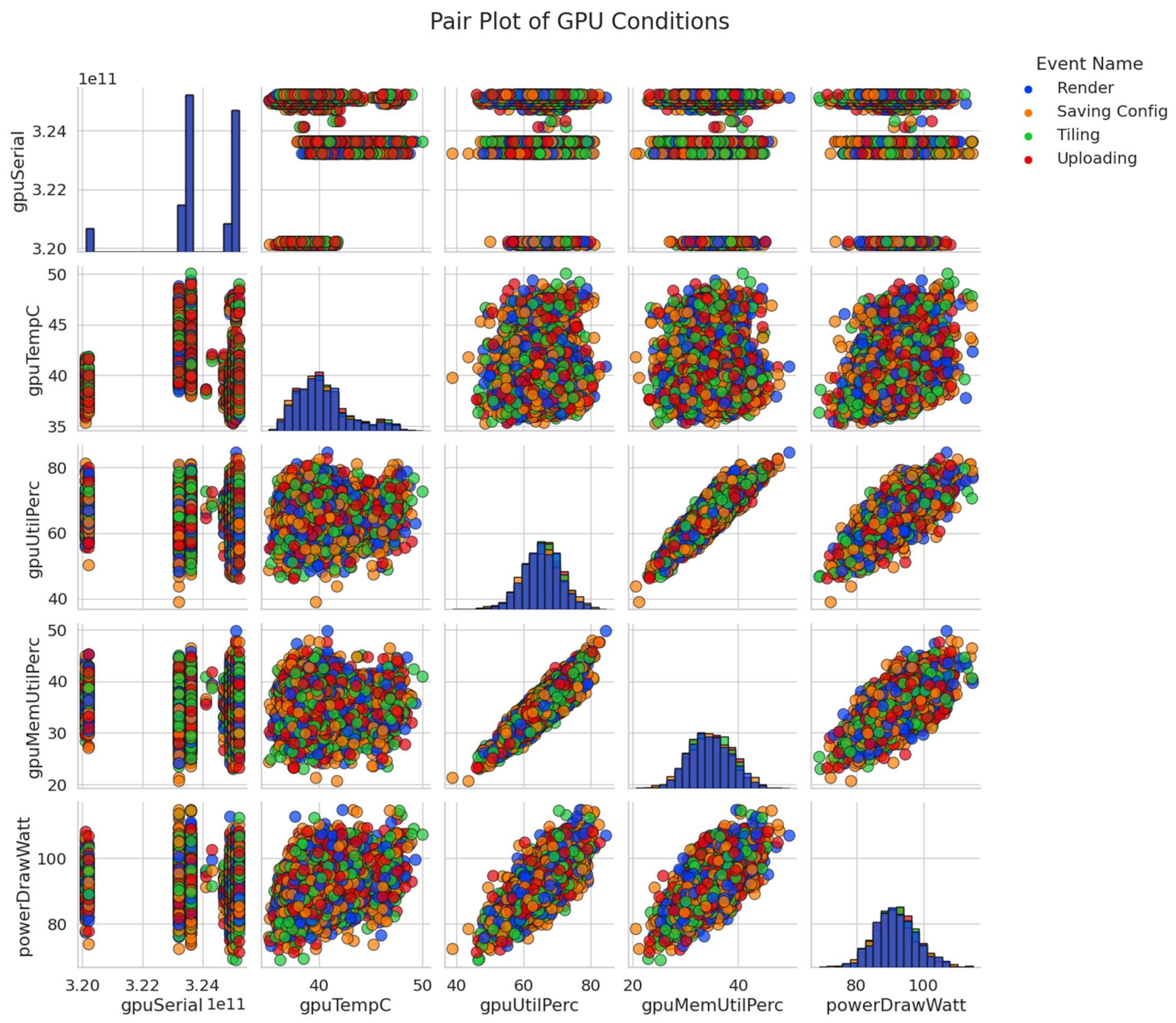


Fig. 11

The pair plot shown in *Fig. 11* depicts how the different GPU conditions are related to each other. The scatter plots in the matrix show how two variables are related, and the histograms along the line show how each variable is spread out. The figure is color-coded to show how the different events affect the GPU. In the text, the names of the events that go with each entry are written. This pair plot shows how all of the GPU variables are related to each other. This makes it easy to look for patterns and correlations between the variables.

8. Results and Conclusion: -

- • The amount of time that is consumed throughout the job runtime by Event render is greater than that of Save Config, Tiling, and Uploading. Even when the other three components of the rendering process are considered, it still accounts for around 95% of the entire process. There is a possibility that certain procedures will take less time given that the vast bulk of their work is concerned with metadata. Because of this, the processing time for the Render subtask will be significantly longer if the image that is going to be produced is of a larger size.
- The correlation among the various operating parameters of the GPU, such as performance time, is noteworthy despite lacking a linear association and instead forming a cluster. This observation presents an intriguing avenue for further exploration and potential insights. The duration of the rendering process is expected to increase in the presence of elevated memory utilisation rates, as a greater amount of memory will need to be processed. This is a reasonable expectation. There exists a linear correlation between the utilisation percentage of a GPU's memory and the performance time, which also exhibits a linear relationship with clustering.
- Furthermore, the level of magnification has an impact on the duration of image rendering. Based on this information, stakeholders have the ability to allocate appropriate hosts that possess the necessary GPU resources.
- The tasks of saving configuration and uploading can be executed concurrently with proper management and are considered to be the most efficient in terms of time consumption.
- The properties of an image can have an impact on how long the rendering process takes, as this investigation has revealed.

9. Future Scope: -

The work has potential future implications, as it may enable the use of prediction techniques for analyzing the dependability and performance of the GPU. The project has the potential to serve as a basis for scalability planning. The user suggests that better results could be achieved if more intensive processing resources were available, specifically 64GB of RAM and a faster GPU. The work has potential future implications, as it may enable the use of prediction techniques for analyzing the dependability and performance of the GPU. The project has the potential to serve as a basis for scalability planning. The user suggests that better results could be achieved if more intensive processing resources were available, specifically 64GB of RAM and a faster GPU.

10. Personal Reflection: -

Through this project, I have learned and gotten better at analyzing system speed and using a variety of tools. I have a better knowledge of the CRISP-DM method, which has helped me figure out how to analyze data. By taking advantage of the power of the Python programming language and using a literate programming structure, I have learned a variety of analytical methods that help me get useful information from data. The project's use of Python Notebooks and famous tools like Pandas, matplotlib, and Seaborn, along with version control on GitHub, has helped me learn these technologies much better. Also, the task has helped me improve my report writing and project management skills. Overall, this experience has helped me grow as a worker and has given me faith in my critical skills.

11. Citations: -

1. Brookes, T. (2021) What is thermal throttling?. Available at: https://www.howtogeek.com/739732/what-is-thermal-throttling/ (Accessed: May 31, 2023).

2. Nvidia Tesla V100 (2023). Available at: https://www.nvidia.com/en-gb/data-center/tesla-v100/ (Accessed: May 31, 2023).

3. Tesla v100 pcie GPU accelerator(2018). Available at: https://images.nvidia.com/content/tesla/pdf/Tesla-V100-PCIe-Product-Brief.pdf (Accessed: May 31, 2023).

12. References: -

Petascale Cloud Supercomputing for Terapixel Visualization of a Digital Twin (no date). Available at: https://arxiv.org/ftp/arxiv/papers/1902/1902.04820.pdf (Accessed: May 31, 2023).