CSC8634_TeraScope Report

Morgan Frodsham (210431461)

11/02/2022

Extended Technical Project: Performance Evaluation of Terapixel Rendering in Cloud (Super)computing

Context

Background

This report details the process and findings of an extended technical project on Newcastle University's cloud (super)computer architecture for visualising environmental data captured by the Newcastle Urban Observatory. It explores the data created during the computational production of this visualisation of Newcastle upon Tyne. Newcastle University and the Newcastle Urban Observatory started this initiative with the idea that "the only way to understand how to monitor a city is to try and monitor one" with a monitoring platform that matches "the scale, complexity and scope of our modern cities" (P James (2016)). It offers an unparalleled opportunity to improve decision making the city by accurately measuring and analysing our urban environment.

Objectives and success criteria

Urban Observatories attempt to replicate the "breadth, longevity and success of astronomical observatories"(P James (2016)) in understanding how cities operate. Funded in partnership with the UK Collabatorium for Research in Infrastructure and Cities ("Details of Grant" (n.d.)), the Newcastle Urban Observatory collects environmental data about the city of Newcastle-upon-Tyne (Observatory (n.d.)). This is considered to be the most expansive set of "environmental data in the UK, with: over 74 urban indicators; 7,000 observations every minute; 3,600 sensor streams; 9 billion data points; 540 CCTV sensors; and hundreds of millions of images" (Newcastle University (2021)). Newcastle University has created a scalable cloud supercomputer software architecture for visualising this data as realistic terapixels (Forshaw (2021)). Terapixels are images containing over one trillion pixels (Holiman et. al (2019)) that allow viewers to interactively browse big data in intense detail across multiple scales (Forshaw (2021)). The terapixel visualisation created by Newcastle University's cloud (super)computer allows the viewer to "zoom in from an overview of just one square kilometre of the city... to see detail within a single room in an office or a house" (Holiman et. al (2019)).

In delivering this initative, Newcastle University had three objectives:

1. "create a (super)computer architecture for scalable viualisation using the public cloud;
2. produce a terapixel 3D city viusalisation supporting daily updates; and
3. undertake a rigorous evaluation of cloud (super)computing for compute intensive visualisation applications" (Forshaw (2021)).

This extended technical project focuses on the third objective. It explores three datasets created from the data produced by the (super)computer during the creation of the terapixel visualisation, and aims to provide useful insights into the computational performance. The process and findings of this report explore the question: "How could the computation of a terapixel visualisation be more efficient?" The success criteria of the project is to identify useful insights about the stages of terapixel computation which are most resource intensive. Developing a better understanding of this is important as producing a high quality terapixel image is an intense process with significant computational costs (resource, energy, environmental and monetary) (Holiman et. al (2019)).

With a growing focus on doing "more with less" ("Efficiency" (n.d.)), cloud computing suppliers and users are trying to continue scaling with fewer computational costs (such as fewer servers and less energy). For humankind to fully reap the benefits of information provided by these visualisations, the computation needs to be efficient and associated costs must be sustainable. Improving the efficiency cloud computing has been a core focus for suppliers and users; indeed, Google returns 90.9 million results for "cloud computing efficiency." As the likelihood and potential impact of climate change is fully realised, users as well as wider society are increasingly concerned about the sustainability of digital technologies, such as cloud computing. Just five organisations - Amazon, Google, Microsoft, Facebook and Apple - use as much electricity annually as New Zealand (Kantor (2021)). Moreover, the Shift Project suggests that tech-related emissions are rising by 6 per cent annually (Kantor (2021)). Data centres are estimated to account for "1% of the total global energy demand" (Renee Obringer and Madani (2021)). Investigating the resource intensity of computing intensive visualisation applications, such as terapixels, is therefore a vital component of a rigorous evaluation and a necessary step towards greater sustainability.

Data mining goals and success criteria

To explore the question "How could the computation of a terapixel visualisation be more efficient?" this extended technical project seeks to identify useful insights about the stages of terapixel computation which are most resource intensive. To achieve this, the goals of data mining are:

1. Which events dominate task duration (run time)? The data mining success criteria is calculating the duration of each event to determine which has the longest duration.
2. What is the relationship between duration and GPU power draw? The data mining success criteria is identifying the power needed for duration time to better understand what demands more GPU power.
3. What is the relationship between GPU power draw and performance? The data mining success criteria is identifying the impact of GPU temperature, core utilisation and memory demand on GPU power to understand what demands more power.

Terminology, tools and techniques

This project utilises the following terminology, tools and techniques:

- R, which is a programming language for statistical computing and graphics.
- RStudio, which is an Interated Development Environment for R.
- Tibble, which is a data frame that stores data in R and appears as a table.
- R packages, which are extensions to the R statistcial programming language. R packages contain code, data, and documentation in a standardised collection format that can be installed by users of R. The R packages used by this report are listed in the section below.
- Tidyverse for R is the R package used for the analysis;

- ggplot2 for R is the R package used for visualisations;
- RColorBrewer is the R package used to provide accessible colour palettes for visualisations;
- ProjectTemplate is the R package used to improve the reproducibility of this analysis;
- all written documentation is created with the R package, RMarkdown; and
- natbib is an R package used for the bibliography.
- Git, which is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code. GitHub is a provider of internet hosting for software development and version control using Git. Git Bash is an application for Microsoft Windows environments which provides an emulation layer for a Git command line experience.
- CRISP-DM (Chapman et. al. and Wirth (2000)) is the methodology used to deliver data workflow best practice.

Data Understanding

Initial data collection

The data for this extended technical project was created during a run (from application checkpoint and system metric output) using 1024 GPU nodes, and this run processes three levels of visualisation (levels 4, 8 and 12) to render the terapixel (Forshaw (2021)). It is unclear whether there have been any problems in data acquisition or extraction as Newcastle University has provided the data.

The data provided shows the timings of producing the terapixel, GPU card performance, and the coordinates for the part of the terapixel image that was rendered in each task (Forshaw (2021)) in three different data sets. This data appears to be raw, but of sufficient quality to be tided and analysed. It will provide sufficient information for an initial investigation into the stages of terapixel computation process which are most resource intensive.

Data description

The run to compute the terapixel visualisation provides the following data sets:

- the application checkpoint events throughout the execution of the render job (timestamp, hostname, eventName, eventType, jobId, taskId);
- the metrics output relating to the status of the GPU on the virtual machines (timestamp, hostname, gpuSerial, gpuUUID, powerDrawWatt, gpuTempC, gpuUtilPerc, gpuMemUtilPerc); and
- the x,y co-ordinates of the part of the image being rendered for each task (jobId, taskId, x, y, level).

Data exploration

The data exploration outlined in this report began with initial questions to better understand what each data set contained and visual investigation to obverse any obvious patterns. Further details of this initial exploration can be seen in the file 'eda.R' in the src folder. Examples are shared below:

1. There are 1024 unique host names, which suggests that these are the names of each virtual machine containing the GPU nodes.

2. There are 1024 unique GPU serials, which suggests that each virtual machine has one GPU node with its own unique serial number.

3. It appears that some virtual machines complete more 'TotalRender' than others, suggesting that some are more productive.

4. There are 65793 taskIds in the applications.checkpoint data set (copied as AC1), which matches the number of taskIds in the task.x.y data set.

5. There are 3 distinct jobIds in the applications.checkpoint data set (copied as AC1), which is the same in task.x.y, and seems to reflect the three visualisation levels (level 4, 8 and 12).

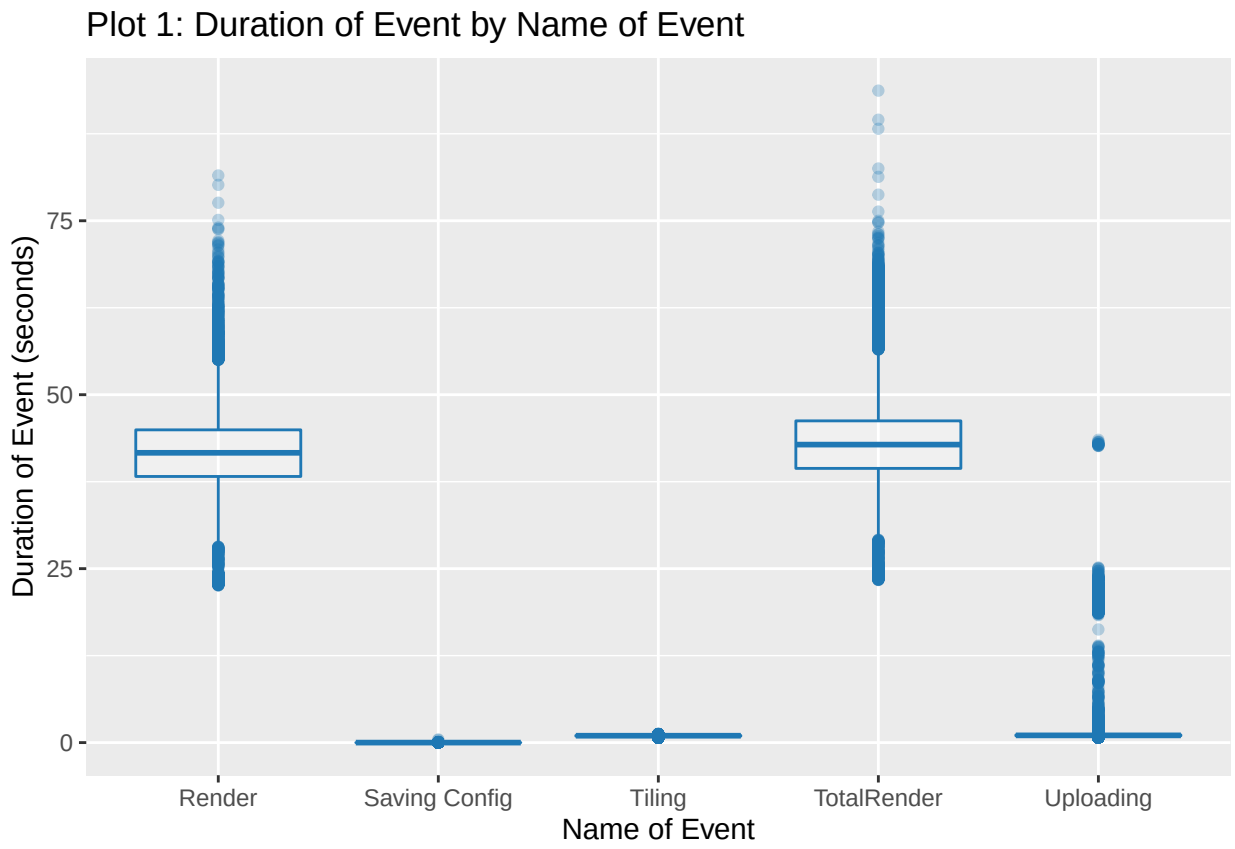
To address the key objective and success criteria, the exploratory analysis outlined in the rest of this section starts with applications.checkpoints and moves to gpu in line with the data mining goals and is structured by those goals. Each goal has it's own 'eda#.R' file in the Rmarkdown 'src' folder.

Goal 1 (Run time)

To create a terapixel, Newcastle University render levels 4, 8 and 12, "and subsample these to fill in each set of three intermediate levels" (Holiman et. al (2019)). There are 5 different events (eventName) that happen during the creation of a visualisation: Saving Config, Render, Tiling, Uploading and TotalRender. The data set application.checkpoints records the timestamp of when these events start and stop (eventType).

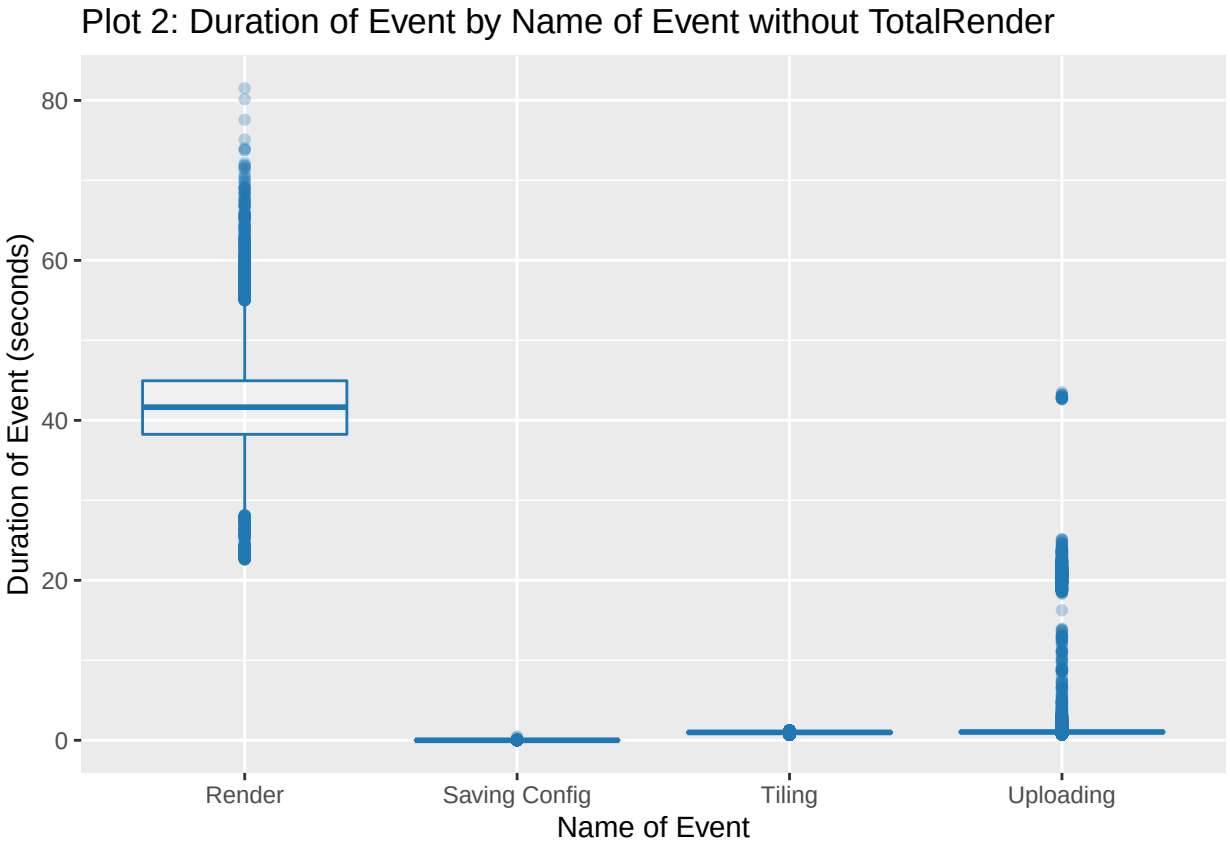
To determine which events have the longest duration (which means that they dominate the run time), the application.checkpoints data set is copied as AC1. AC2 applies a pivot wider function to AC1 so that eventType 'START' and 'STOP' become columns for the corresponding timestamp for each eventName. Both columns are mutated so that their values are date and time. In a new column, duration is calculated by subtracting START from STOP.

To investigate the duration of each event, a box plot was created of AC2 with event duration by event name (eventName).



The plot above highlights that 'TotalRender' has the longest duration. It is assumed that this is because it includes the duration of all the other events.

To better see the duration of the other events AC3 was created by filtering out 'TotalRender' from AC2. Another box plot is used to visualise the event duration by event name.



It is clear that 'Render' has the longest duration, and therefore is the event that dominates the run time. This is to be expected; rendering visualisations is considered to be a good test of hardware performance as it is capable of absorbing all available compute resource (Holiman et. al (2019)). It is also interesting to note that 'Saving Config' and 'Tiling' take close to zero seconds to execute.

The virtual machines with the longest duration for 'Render' are listed in the table below.

Table 1: Hostname of virtual machines with the longest run time (duration) for the event 'Render'

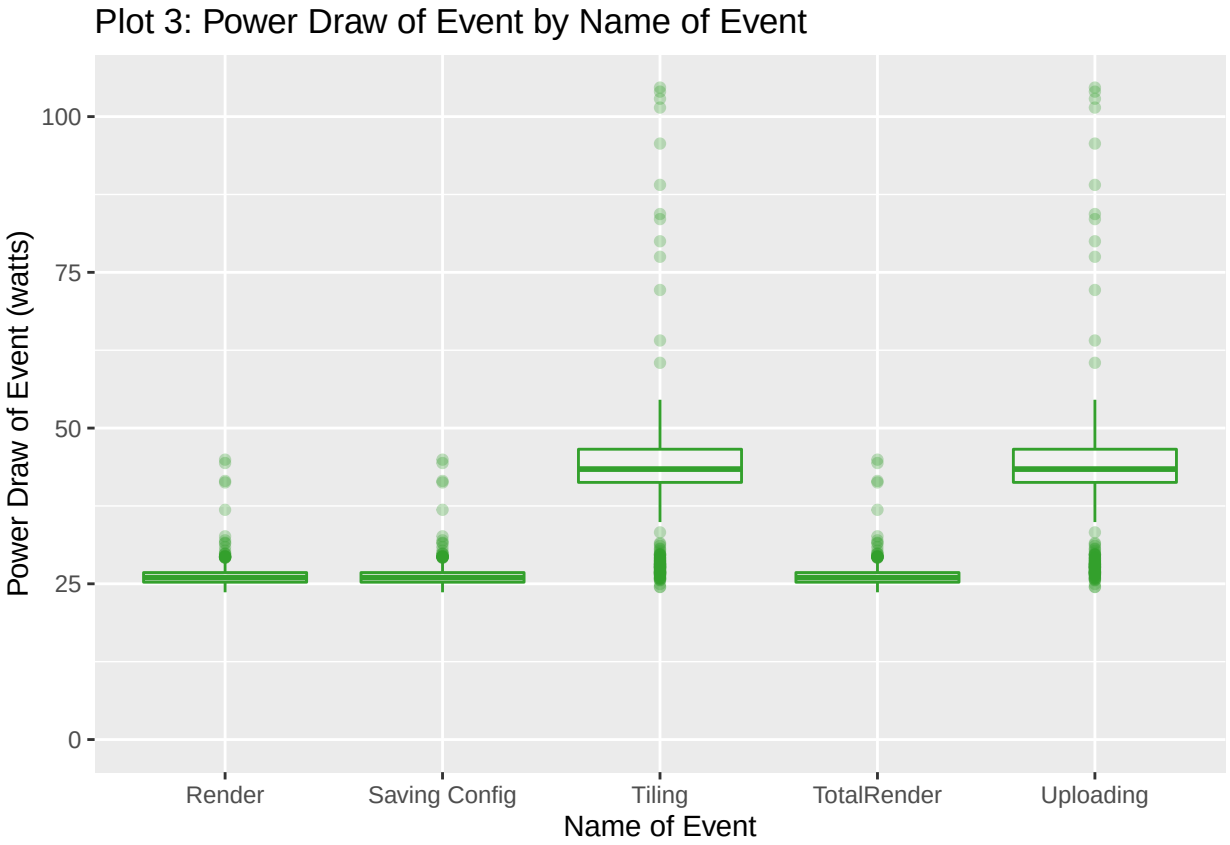
hostname	duration
0d56a730076643d585f77e00d2d8521a00000I	81.51
4a79b6d2616049edbf06c6aa58ab426a000003	80.16
95b4ae6d890e4c46986d91d7ac4bf08200000G	77.59
6139a35676de44d6b61ec247f0ed865700000I	75.12
5903af3699134795af7eafc605ae5fc700000U	74.02

Further exploration related to this data goal can be viewed in 'eda1.R.'

Goal 2 (Power draw)

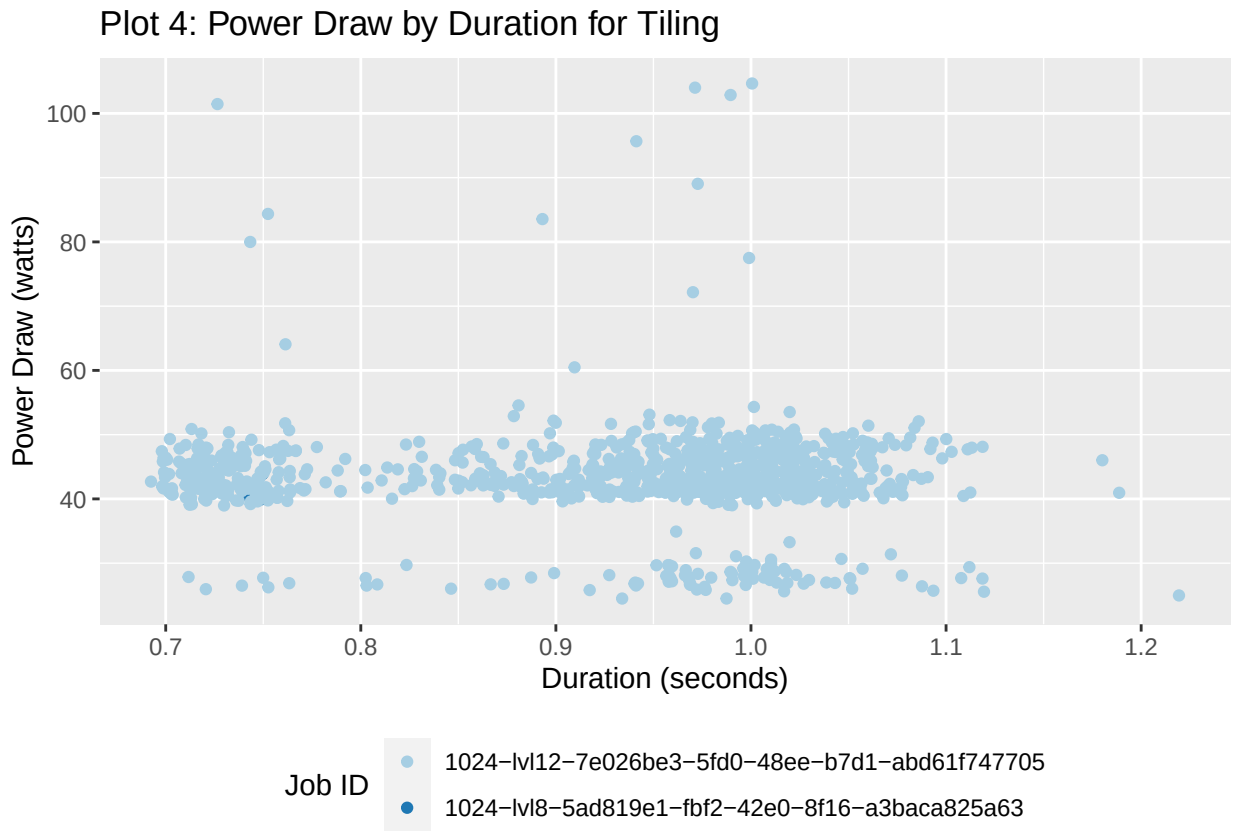
To identify the power needed for duration time, the application.checkpoints and gpu data were combined. This could be achieved by either 'timestamp' or 'hostname' as they are the common values across both data sets. Joining by hostname was unlikely to provide us with the desired result as hostnames appeared more than once in both data sets. However, there are more timestamps in the gpu data set than the application.checkpoints data set; when the two are joined (Power_Duration2), we explore a subset of the data becase the timestamp is rounded to the nearest two seconds; this places limitations on our results.

Initially, a box plot was created of event power draw by event name to see if there was an obvious correlation between duration and power consumption.



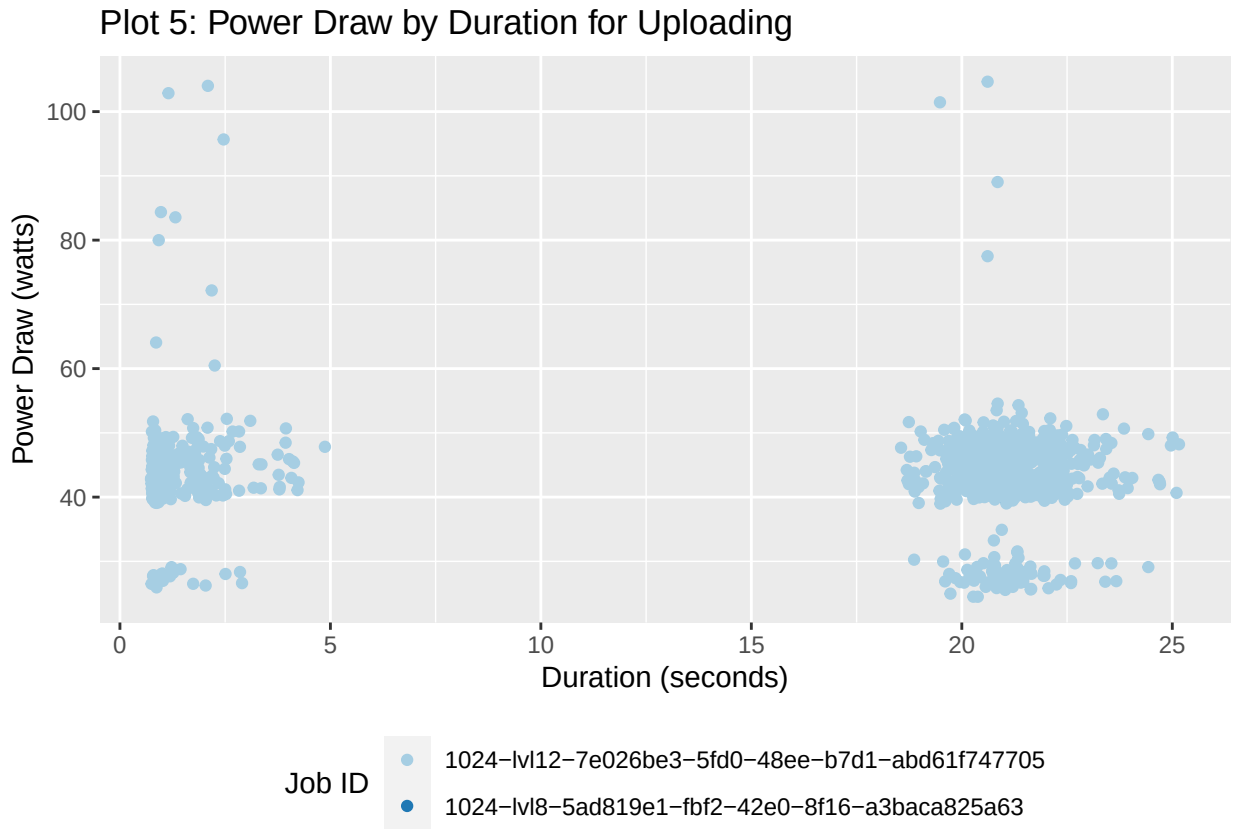
It was a surprise that the events with the highest duration ('TotalRender' and 'Render') do not draw (consume) the most power. The events that consume the most power are 'Tiling' and 'Uploading.'

To investigate further, Power_Duration2 was filtered to observe the power draw by duration for 'Tiling.' This was graphed in a scatter plot, coloured by 'jobId,' and is presented below.



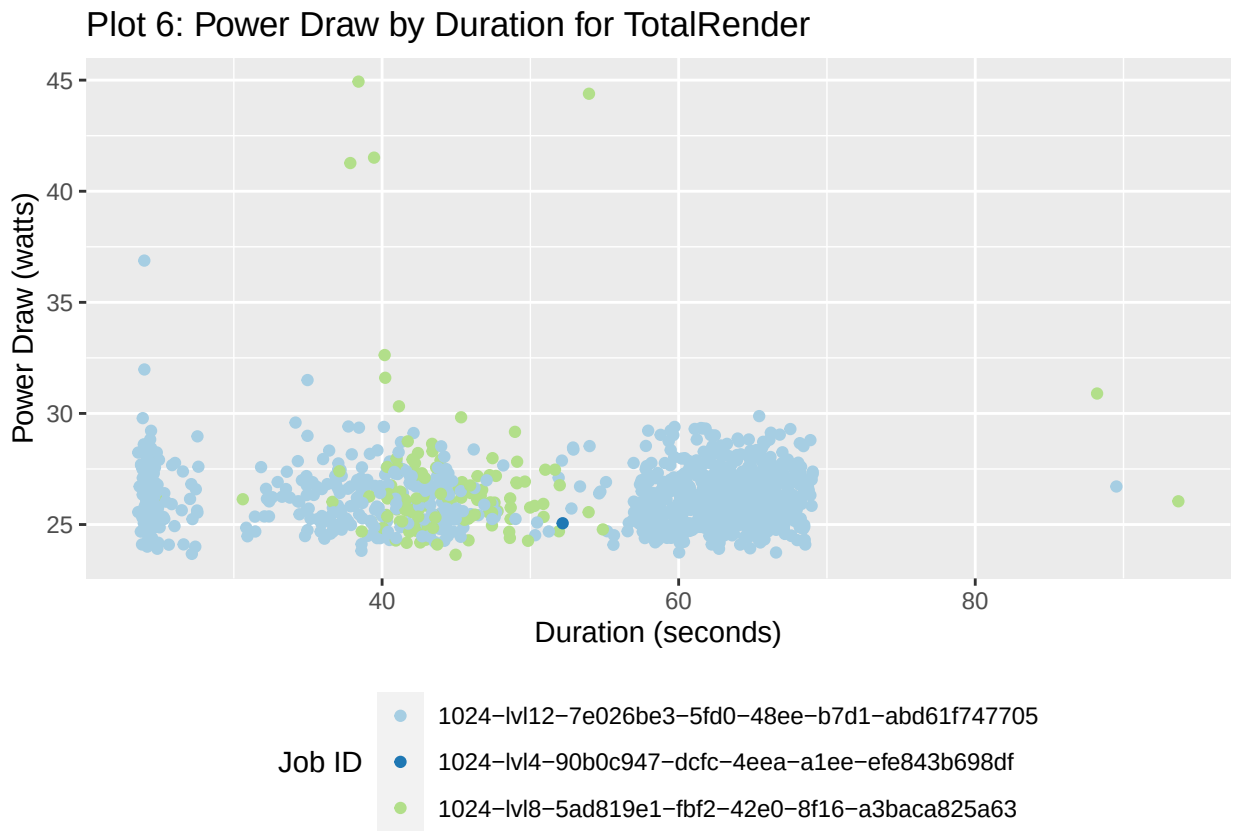
Given that there are two job IDs ('jobId') that correlate with 'Tiling,' it is interesting to observe that most of the tiling occurs for the job ID of level 12 (1024-lvl12-7e026be3-5fd0-48ee-b7d1-abd61f747705). It is unclear if this is a problem caused from the data wrangling, or a true result in the data so further investigation is required.

Power_Duration2 was also filtered to see the power draw by duration for 'Uploading.' This was graphed in a scatter plot, coloured by 'jobId,' and is presented below.



The same pattern is observed in 'Uploading' as for 'Tiling,' where there is only power draw and duration data for two job IDs (level 8 and 12). Interestingly, here there are also two distinct groups for the duration of 'Uploading,' and both groups split in their consumption of power (excluding some outliers, it appears almost that the software is testing the power needed for each task). This would benefit from further exploration.

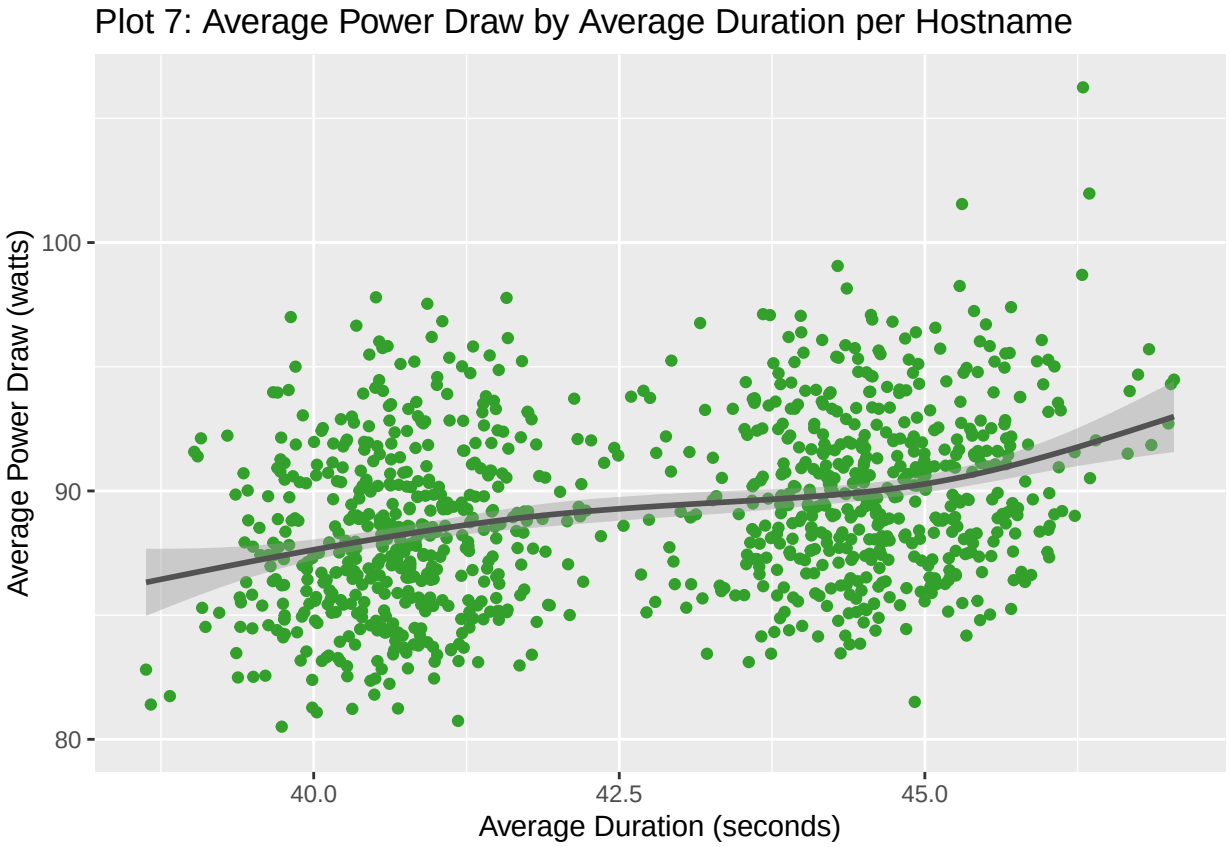
For comparison, Power_Duration2 was filtered to see the power draw by duration for 'TotalRender.' This was graphed in a scatter plot, coloured by 'jobId,' and is presented below.



There are three job IDs visible in the scatter plot for 'TotalRender,' although data for level 4 (1024-lvl4-90b0c947-dcfc-4eea-a1ee-efe843b698df) only appears once. It also seems that there are three groups for duration, but no group divide in power consumption. Tasking software or algorithm efficiency could account for why there are three distinct groups in the scatter plot for 'TotalRender' as more time intensive tasks could be allocated to specific virtual machines.

To better understand whether there is any observable pattern, the mean, standard deviation and coefficient of variation for the power draw and duration of each virtual machine ('hostname') was calculated.

These values are combined into a single data set and visualised with the scatter plot below.



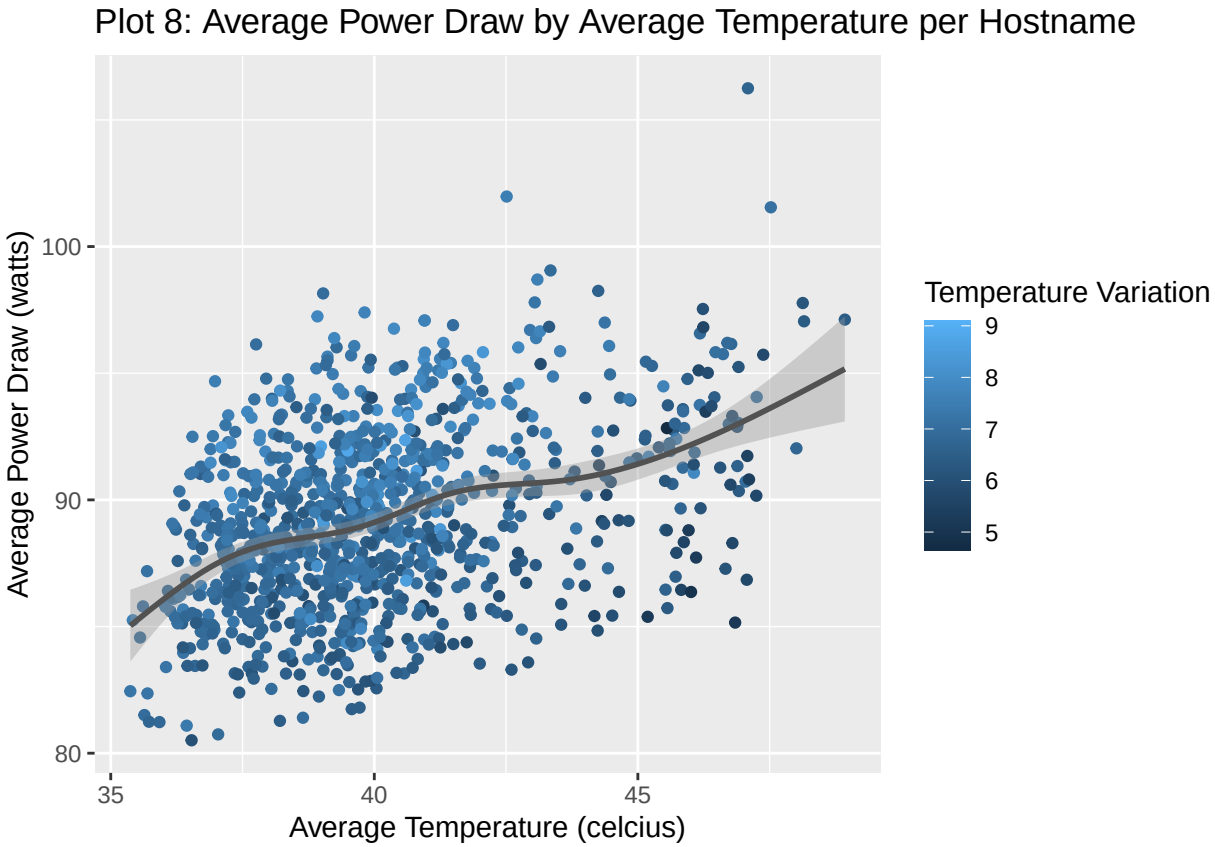
It does appear that there are two clear groups of virtual machines, but it is not clear why. Further investigation is required.

Further exploration related to this data goal can be viewed in 'eda2.R.'

Goal 3 (Performance)

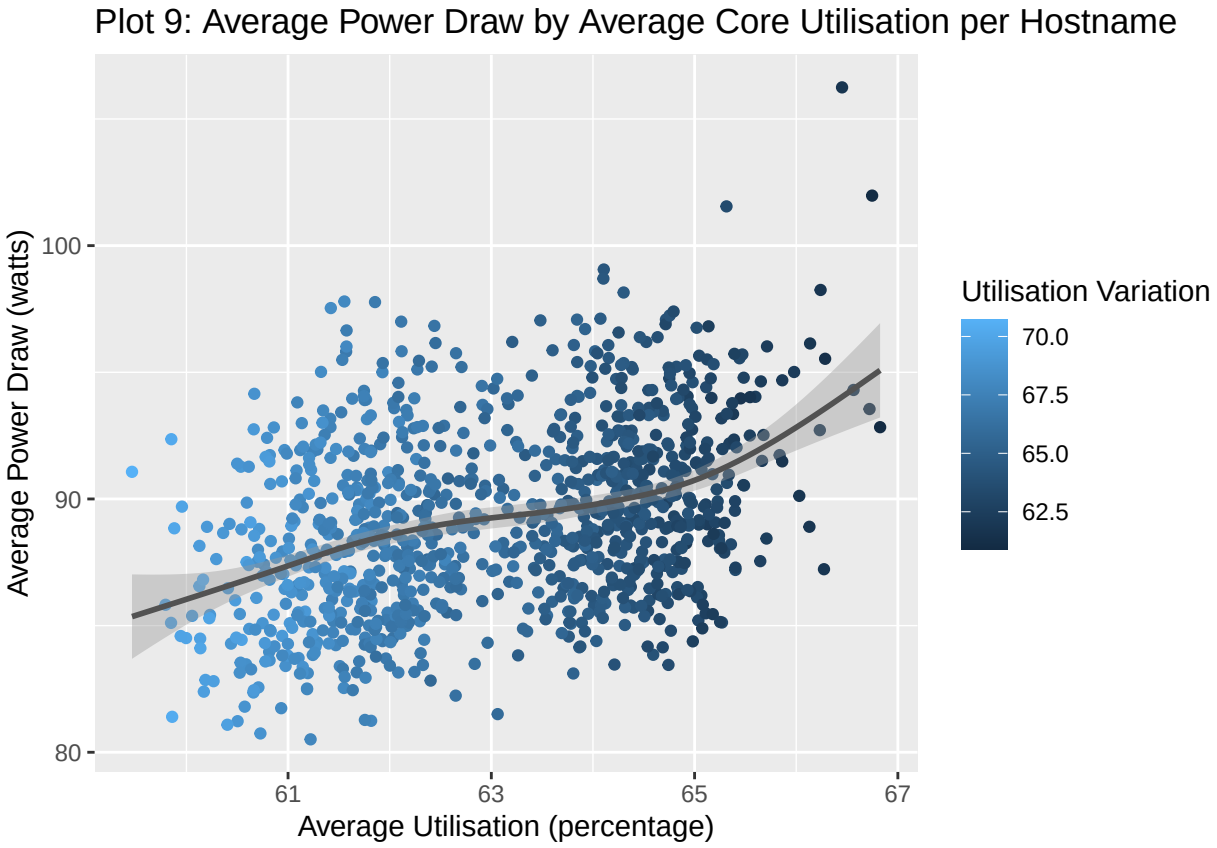
To understand the relationship between GPU power draw and performance, the impact of GPU temperature, core utilisation and memory demand on power consumption needed to be determined. The mean, standard deviation and coefficient of variation are calculated for all GPU card metrics.

The graph below visualises the average power draw by average temperature of each virtual machine and, therefore, each GPU card (as it was previously demonstrated that each virtual machine appears to have a unique one). The temperature of the virtual machine has a lower variation when the dot is darkest.



Temperature and power draw appear to be loosely coupled; it is not obvious that there is a strong relationship between these two metrics of performance. However, it does seem that the temperature of the virtual machine machine, on average, increases as more power is consumed.

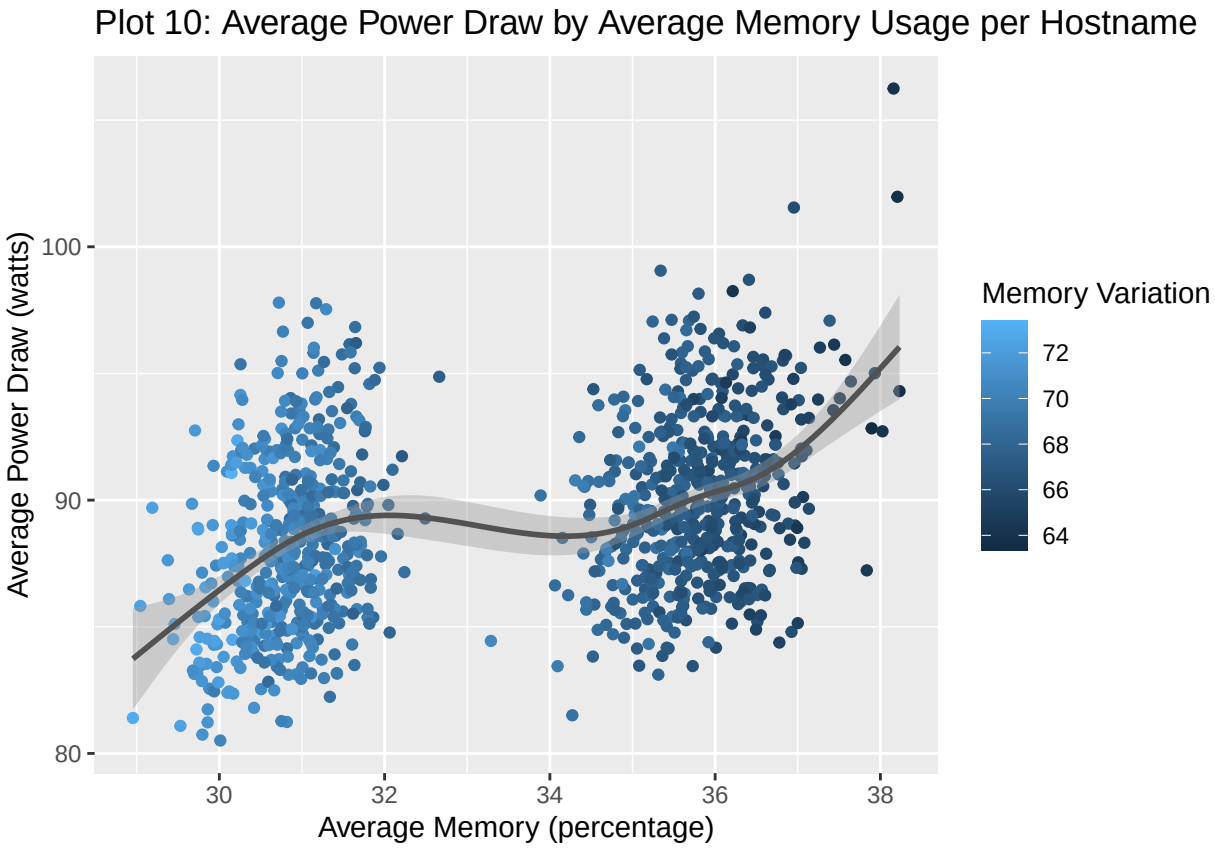
The graph below visualises the average power draw by average core utilisation of the GPU card from each virtual machine. The core utilisation of the GPU card varies less when the dot is darkest.



GPU core utilisation and power draw appear to be loosely coupled; it is not obvious that there is a strong relationship between these two metrics of performance. However, it does seem that more power is consumed, on average, as the GPU core utilisation increases.

In this scatter plot we begin to observe two groups emerge in the virtual machines' utilisation of the GPU core. It is interesting that the group to the right, which uses more of the GPU core, varies less.

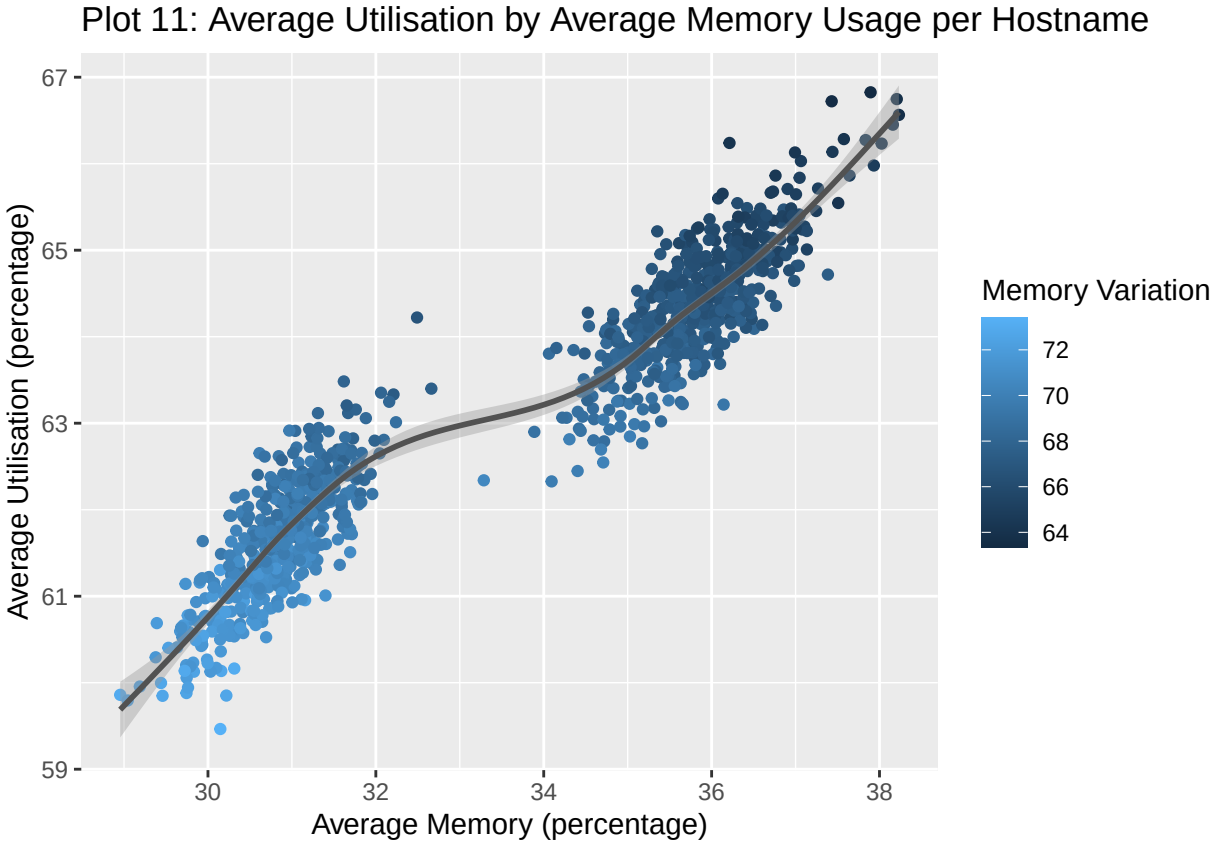
The graph below visualises the average power draw by average memory use of the GPU card from each virtual machine. The memory demand of the GPU card varies less when the dot is darkest.



GPU memory use and power draw appear to be loosely coupled; it is not obvious that there is a strong relationship between these two metrics of performance. However, it does seem that the more memory used by the GPU, on average, the more power is consumed.

In this scatter plot we observe two obvious groups between the virtual machines' use of GPU memory. It is interesting that the group to the right, which uses more GPU memory, varies less.

The graph below visualises the average GPU core utilisation by average memory use of the GPU card from each virtual machine. The memory demand of the GPU card varies less when the dot is darkest.



The average utilisation of the GPU core and average memory consumption appear to be tightly coupled; there appears to be a strong relationship between these two metrics of GPU performance. It does seem that, on average, the more GPU memory that is required, the more the GPU core is utilised. This is to be expected as increasing the memory demand will increase the load on the GPU core, requiring more of it to be utilised.

Further exploration related to this data goal can be viewed in 'eda3.R.'

Context (second iteration)

Objectives and success criteria

This extended technical project explores "How could the computation of a terapixel visualisation be more efficient?" to identify useful insights about the stages of terapixel computation which are most resource intensive. An initial investigation into the three data sets provided by Newcastle University suggests that rending the visualisation is the most time intensive, but that tiling and uploading the visualisation is the most power intensive. GPU performance also appears to be loosely correlated with power consumption. However, usilisation of the GPU core and memory consumption appear to be tightly correlated. A better understanding of what is driving the duration of the visualisation and the relationship of GPU cards performance is needed.

Data mining goals and success criteria

To better understand the resource intensity of computing a terapixel, the three data sets need to be explored as one whole picture. This will enable us to investigate what about the visualisation is driving the resource intensity and whether there are indicators of hardware issues that warrent further investigation. This second iteration of data mining aims to determine:

4. What is the variation in computation requirements for particular tiles? The data mining success criteria is identifying if there is a variation in computing requirements to determine which tiles are most intensive to produce.
5. Are there particular GPU cards (based on their serial numbers) whose performance differs to other cards? The data mining success criteria is identifying the outlier GPU cards (for example, cards that are perpetually slow) to determine which cards could benefit from further investigation to improve performance and resource efficiency.

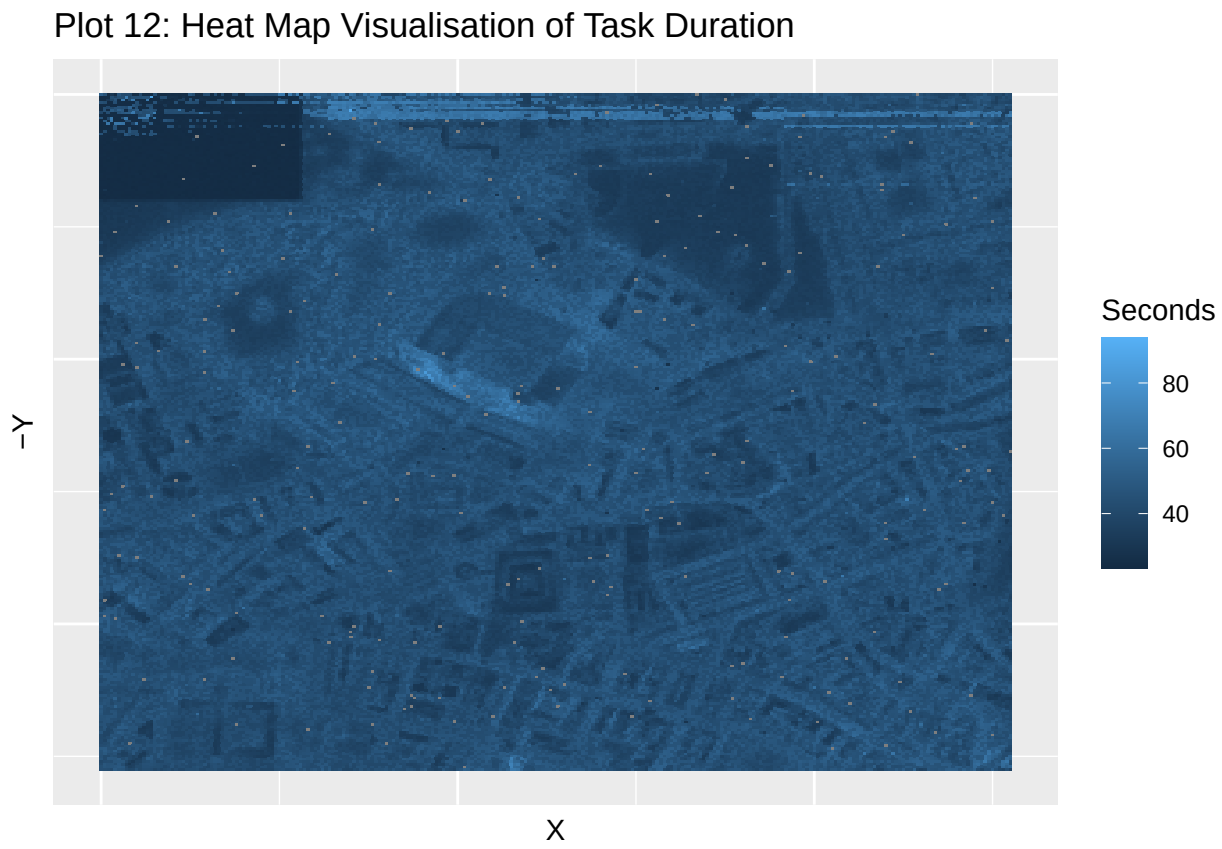
Data understanding

The details of data collection, description and quality outlined in the first iteration of data understanding also apply to this second iteration. This second iteration also uses the data set task.x.y.

Data exploration

Goal 4 (Tile computation)

There is a significant amount of variation across the power consumption and render duration for each virtual machine ('hostname'). To better understand why this might be occurring, a heat map was created from the x and y co-ordinates in task.x.y to identify if the intensity of the visualisation accounts for the variation in resource demand. The y co-ordinates needed to be inversed to match the heat map to the terapixel visual key.



As previously mentioned, rendering the visualisation in these tile coordinates ultimately generates this terapixel.

The heat map suggests that the more textured the content of the visualiation, the longer the visualisation takes to render. The tiles with the longest duration appear to be the more patterned edge of the stadium (perhaps seating), trees (with leaves), and roofs with more texture (such as satellites).

Goal 5 (GPU serials)

To identify potential hardware issues in the virtual machines that could improve performance and resource efficiency, we can identify the GPU cards which are perpetually slow.

This is achieved by filtering Power_Duration2 by the event name 'TotalRender,' grouping the data by the GPU serial number ('gpuSerial') and calculating either the median or mean. In this instance, both were calculated, and were very similar.

The median of the slowest GPU cards are presented below.

Table 2: The hostnames and GPU serials of the 10 slowest virtual machines

hostname	gpuSerial	med_time
95b4ae6d890e4c46986d91d7ac4bf082000010	320118119713	69.04
95b4ae6d890e4c46986d91d7ac4bf08200000G	320118119842	68.97
b9a1fa7ae2f74eb68f25f607980f97d7000002	325217086519	68.97
95b4ae6d890e4c46986d91d7ac4bf08200000V	325017018547	68.88
95b4ae6d890e4c46986d91d7ac4bf08200000Z	324917146854	68.88
95b4ae6d890e4c46986d91d7ac4bf08200000P	325217085268	68.77
4ad946d4435c42dabb5073531ea4f315000003	320218055524	68.74
95b4ae6d890e4c46986d91d7ac4bf08200000J	325017020353	68.72
95b4ae6d890e4c46986d91d7ac4bf08200000W	325017019501	68.54
95b4ae6d890e4c46986d91d7ac4bf082000013	324817027696	68.53

Interestingly, there is one virtual machine (95b4ae6d890e4c46986d91d7ac4bf08200000G) that has the longest duration to render the visualiation (as discovered in the first iteration of data exploration) and appears to be perpetually slow.

As it has been identified that the GPU core utilisation and memory use divides the virtual machines into two groups, it would be interesting to explore whether that divide can be observed in the medians calculated in Power_Duration3.

Evaluation

Assessment of findings

The process and findings of this report explore the question "How could the computation of a terapixel visualisation be more efficient?" The success criteria of the project was to identify useful insights about the stages of terapixel computation which are most resource intensive. The following results have been identified:

1. Run time (duration) of events in the creation of a terapixel is dominated by rendering the visualisation (eventName 'Render'). The five virtual machines with the longest duration have been identified.
2. The most power is consumed by 'Tiling' and 'Uploading' the terapixel. We also learnt that there are only two levels ('jobIds') for these events and that two distinct groups emerge between the virtual machines in the time it takes for the terapixel to be uploaded ('Uploading') as well as totally rendered ('TotalRender').
3. There does not appear to be a strong correlation between power consumption ('gpuPowerDraw') and the other performance metrics of temperature, GPU core utilisation and memory use. However, the relationship between GPU core utilisation and memory appears tightly coupled.
4. The time it takes to create a terapixel appears to be driven by the visual intensity of the content being visualised. In this example, the tiles with the longest duration have more texture, such as stadium seating and roofs with satellites.
5. The 10 most perpetually slow GPU cards and virtual machines have been identified. We learnt that only one of these virtual machines has one of the top five longest render duration.

The findings of the exploratory analysis in this extended technical report does produce useful insights. These results offer insights into how the computation process could be improved; run time, power consumption, GPU core utilisation and GPU memory demand are indicators of resource intensity (Holiman et. al (2019), Torsten Hoefler and Belli (2015)). Time, power and hardware demands increase the monetary and environmental costs of cloud computing (Microsoft and WSP (2020), "Efficiency" (n.d.), Kantor (2021)). It is vital that we understand what drives the demand for, and intensity of, these resources in order to realise a technologically sustainable future.

This project has been limited by the time available, programming capabilities, and focus on data exploration instead of dashboard or model creation. The process has been sufficiently effective to somewhat address the objective and success criteria of this extended technical project. If the extended technical project was continued, it would benefit from more rigerous statistical analysis. Possible actions to better understand the results identified in this report could also include:

- identifying why there are only two jobIDs (visualisation levels) for 'Tiling' and 'Uploading';
- determine why 'TotalRender' and 'Uploading' have distinct groups for run time (duration) and power consumption, and whether this applies for other events in the creation of the terapixel;
- determine why there are also distinct groups in the GPU core utilisation and memory usage to identify if it is the same driver as for the split groups of duration;
- explore why there is so much variation in the GPU performance measures of the virtual machine; and
- better understand the impact and efficiency of the tasking software in this cloud (super)computer architecture.

Reflection

This extended technical project was more enjoyable than previous projects in the MSc. From the beginning of the project I felt more confident that I could conduct the analysis necessary and was able to find most of my own answers to technical questions through research. I also felt more comfortable with the technical suite being used, and was able to experiment more with different types data tidying and visualisations. I also continued to use CRISP-DM my methodology and as the structure for my report; I felt more confident to adapt it to fit the scope and requirements of the project instead of following it precisely.

I also found this project more frustrating, however, on a personal level. I have a full-time, high-profile and fast-paced job at the same time as this MSc and, due to a short-term increase in my workload, I was unable to spend more than 2 to 3 hours at a time on this project during evenings and weekends. This meant that I felt unable to immerse myself as deeply in the project as I would have liked.

Going forward in similar projects, I would like to conduct further statistical analysis and am looking forward to undertaking the statistics module in the second year of this MSc. In the meantime, I would also like to focus more time and attention going beyond the exploratory analysis phase. For example, it may have been sufficient to not undertake a second iteration of data exploration and instead build a Shiny dashboard (which I would love to practice).

Bibliography

Chapman et. al., Juian Clinton, Pete Chapman, and Rudiger Wirth. 2000. *CRISP-DM 1.0: Step-by-Step Data Mining Guide*. https://www.the-modeling-agency.com/crisp-dm.pdf.

"Details of Grant." n.d. https://gow.epsrc.ukri.org/NGBOViewGrant.aspx?GrantRef=EP/P016782/1.

"Efficiency." n.d. https://www.google.co.uk/about/datacenters/efficiency.

Forshaw, Matthew. 2021. *Summary*. https://github.com/NewcastleDataScience/StudentProjects202122/blob/master/TeraScope/Summary.md.

Holiman et. al, Manu Antony, Nicolas S. Holliman. 2019. "Petascale Cloud Supercomputing for Terapixel Visualization of a Digital Twin." *CoRR* abs/1902.04820. http://arxiv.org/abs/1902.04820.

Kantor, Alice. 2021. "Big Tech Races to Clean up Act as Cloud Energy Use Grows," May. https://www.ft.com/content/c719f655-149c-4ce0-a7a5-18527c7776cf.

Microsoft, and WSP. 2020. "The Carbon Benefits of Cloud Computing." https://www.microsoft.com/en-gb/download/details.aspx?id=56950.

Newcastle University, Proffessor P James on behalf of. 2021. "Written Evidence Submitted by Newcastle University (Evp0115)." https://committees.parliament.uk/writtenevidence/22873/html/.

Observatory, Newcastle Urban. n.d. "Our Urban Observatory." https://www.ncl.ac.uk/who-we-are/vision/urban-observatory/.

P James, J Jonczyk, RJ Dawson. 2016. "The UK Urban Observatory Programme - Newcastle: Towards Slightly Less Dumb Cities." https://huckg.is/gisruk2017/GISRUK_2017_paper_15.pdf.

Renee Obringer, Debora Maia-Silva, Benjamin Rachunok, and Kaveh Madani. 2021. "The Overlooked Environmental Footprint of Increasing Internet Use." *Resources, Conservation & Recycling* 167. https://doi.org/10.1016/j.resconrec.2020.105389.

Torsten Hoefler, and Roberto Belli. 2015. "Scientific Benchmarking of Parallel Computing Systems." https://doi.org/10.1145/2807591.2807644.