CSC8634_TeraScope Report

Morgan Frodsham (210431461)

10/01/2022

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

Understanding

Background

This report details the process and findings of an extended technical project on Newcastle University's cloud supercomputer 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 (Nicolas S. Holliman (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 supercomputer 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" (Nicolas S. Holliman (2019)).

In delivering this imitative, Newcastle University had three objectives:

1. "create a supercomputer 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 supercomputing 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 supercomputer architecture 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) (Nicolas S. Holliman (2019)).

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)), with data centres estimated to account for 19% of energy consumption (Tank (2020)). 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.

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 run times? The data mining success criteria is calculating the run time (duration) of each event to determine which has the longest duration.
2. What is the relationship between run time and GPU power draw? The data mining success criteria is identifying the power draw during event duration.
3. What is the relationship between GPU power draw and performance? The data mining success criteria is the **...**
4. Can we identify particular GPU cards (based on their serial numbers) whose performance differs to other cards? (i.e. perpetually slow cards). **...**
5. What is the variation in computation requirements for particular tiles? **...*
6. What can we learn about the efficiency of the task scheduling process? **...**

- we should see efficientl linear scalaing as we add more compute notdes - gustafosn-Baris' law Nicolas S. Holliman (2019) Rendering images is considered to be a good test of hardware performance as it is capable of absorbing all available compute resource, following Gustafson-Barsis' law that the problem size scales to fill the compute capacity (Nicolas S. Holliman (2019)).

##Data Understanding - What, concisely, did you do?

Initial data collection

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). 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 (levels 4, 8 and 12) of visualisation 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 will provides 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 quality

(add)

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. For example:

```
# How many unique hosts are there?
hosts <- AC1 %>%
  group_by(hostname) %>%
  summarise (n_distinct(hostname)) %>%
  count() %>%
  print()

# How many GPU serials are there?
gserials <- GPU1 %>%
  group_by(gpuSerial) %>%
  summarise(n_distinct(gpuSerial)) %>%
  count() %>%
  print()

# How many times does is a visualisation totally rendered?
R <- AC1 %>%
  filter(eventName == "TotalRender") %>%
  count() %>%
  print()
```

```
# How many taskIds are there?
ID <- AC1 %>%
  group_by(taskId) %>%
  summarise(n_distinct(taskId)) %>%
  count() %>%
  print() # This is the same in both application.checkpoints and task.x.y


# How many jobIds are there?
j <- AC1 %>%
  group_by(jobId) %>%
  summarise(n_distinct(jobId)) %>%
  count() %>%
  print() # This is the same in both application.checkpoints and task.x.y, and corresponds to the levels
```

To address the objectives and success criteria, the exploratory analysis outlined here starts with applications.checkpoints, moves to gpu, and finishes with the task.x.y in line with the data mining goals. The rest of this section is structured by those goals.

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" (Nicolas S. Holliman (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 dominate the run time AC1 is a copy of the application.checkpoints data set. 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.

```
# Display AC2 with START, STOP and duration as columns
print(AC2, n = 6, width = Inf)
```

To investigate the duration of each event, a boxplot was created of duration by event name.

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

AC3 is created by filtering out TotalRender from AC2. Another boxplot was created of duration by event name.

```
# Remove Total Render from results
AC3 <- AC2 %>%
  filter(eventName != "TotalRender")

# LABEL AND SAVE # Plot duration of eventName activities without total render
ggplot(AC3, aes(x = eventName, y = duration)) + geom_boxplot()
```

It is clear that 'Render' has the longest duration, and therefore is the event that dominates the run time. Further exploration related to this data goal can be viewed in EDA1.

Goal 2 (Power draw)

What is the relationship between run time and GPU power draw? The data mining success criteria is identifying the power draw during event duration.

second investigation in to understand the power usage of the top 5, average and bottom five event runtimes. (power draw and render time Q)

Create graphs.

Goal 3 (Performance)

Goal 4 (GPU performance)

Goal 5 (Tile computation)

Goal 6 (Efficiency)

Evaluation - How successful has it been? Provide evidence sing appropriate evaluation methodologies, and comment on the strengths/weaknesses of your evidence in answering this question.

- What are the future implications for work in this area? If applicable, which areas of extension work are now possible due to the foundational work you have performed in this project?
- execution time if often the best metric for accessing comuter performance (Hoefler)

Reflection

- A brief relfection on your personal and professional learning in undertaking this project. Here you can comment on how you found the process, what you learned about the technologies and methodologies you used, which aspects you found most difficult/straightforward, and any conclusions which will inform the way you undertake similar projects in future.

Project Plan

(reference approaches to understanding this - which data will be prioritised, and mention iterations.)

Costs and benefits

This not a resource intensive project (as outlined by the resource invitory below). If the insights generated by this extended technical project are significant, there could be benefits for both Newcastle University and the Newcastle Urban Observatory. For example, the analysis and findings could provide further evidence, or indeed highlight new evidence, to support Newcastle University in delivering its objective to evaluate cloud supercomputing for compute intensive visualisation applications.

Terminology

The terminology used throughout this report includes:

- R, which is a programming language for statistical computing and graphics.
- RStudio, which is an Interated Development Environment for R.
- 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.

- 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.
- Tibble, which is a data frame that stores data in R and appears as a table.

(is there anything else I use?)

Inventory of resources

This project has been undertaken by one part-time person, on a Dell G7 17" laptop. Git, RStudio and a selection of R packages (detailed in the next section) have been used to deliver the project. Newcastle University also provided monitoring data from its cloud supercomputer architecture.

Initial assessment of tools and techniques

This project utilises the following tools and techniques:

- CRISP-DM (Chapman et. al. and Wirth (2000)) is the methodology used to deliver data workflow best practice;
- 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;
- Git (specifically GitHub and GitBash) is used for version control;
- all written documentation is created with the R package, RMarkdown; and
- natbib is an R package used for the bibliography

Bibliography

"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.

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.

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

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

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.

Tank, The Shift Project: The Carbon Transition Think. 2020. "Implementing Digital Sufficiency." https://theshiftproject.org/wp-content/uploads/2021/07/TSP_DigitalSufficiency2020_Summary_corrige.pdf.