Project Preparation Report

Investigating the Power Efficiency of Numerical Software

Siddharth Chandrashekar

March 26th 2021

Contents

1. Introduction

Supercomputers have had a major effect on everyone's lives over the years. There are countless uses such as weather prediction, molecular dynamics, automobile engineering, astronomy (Tynan, 2020) and many more that have made us reliant on supercomputing. Naturally, since there are so many benefits, there has been a constant demand for faster computing.

The Top500 list from just 10 years ago in November 2010 shows the fastest supercomputer, Tianhe 1-A, to have a peak at 4,701 TFlops/ and a power consumption of 4,040 kW (T500, 2010), in November 2020 Fugaku is the fastest supercomputer with a peak of 537,212 TFlops/s and power consumption at 29,899 kW (Top500, 2020). The top500 list was initially created in 1993 and only measured performance, but from in June 2013, they also started publishing the Green500 (Top500, 2013) which is ranked based on Flops/s per Watt. This suggests that the energy efficiency of these machines is already viewed as an important factor.

The energy effieciency of machines is improving because fo effiency improvements in CPUs over this period of time. In November 2010, the supercomputer that consumed the most power in the top 10 was IBM's Summit at 6,950 kW, in Novermber 2020, out of the 8 power readings provided, 4 are above 10,000 kW. This mean that although the power effiency of hardware is improving, supercomputers still use more chips and more power to hit higher performance numbers.

This has two important consequences; the cost of running supercomputers is increasing and the environmental impact of these supercomputers in worsening. In the case of Fugaku, we can find that the price per kWh in Japan for a household is $0.29 (Sönnichsen, 2020), the annual cost in power alone is approximately $75,000,000. It is unlikely that the household rates are paid for a large system like Fugaku, but it is a reasonable method of making an approxmation. When we look at the environmental impact, supercomputers have been scrutinized for their carbon emissions in the past. In 2009, the Met office supercomputer in London came under scrutiny for being a massive cause of carbon emission while modelling climate change (Johnson, 2009). If we take an estimate of amount of CO2 produced per kWh from US energy Information Administration of 0.417 kg/kWh (U.S. Energy Information Administration, 2020), the Carbon emissions per year over 100,000 tonnes just from Fugaku. This depends on the percentage of energy being produced by fossil fuels and renewables but it is a good way to estimate the emission from a modern supercomputer.

There is an environmental and monetary benefit to working on reducing the power consumption of supercomputers. This proposition is for an investigation into some methods of reducing the power consumption of these machines while they are working on intense workloads. The possible methods proposed include examining the impact on power consumption by using MPI, openMP or hybrid MPI/openMP, using the matrix free method for matrix multiplication and adjusting the data storage formats.

2. Previous Dissertation Review

The dissertation I reviewed was about Profiling and Optimisation of Meshing in the OpenFOAM for the Renault F1 Team by Daniel Hews written 2020.

2.1 Project Summary

The dissertation discusses the use of Computational Fluid Dynamic (CFD), its importance to an F1 team when designing components and how speeding up a component of this process, the meshing, by optimising the *snappyHexMesh* utility helps the F1 team. Renault implemented CFG using the OpenFOAM CFD software, the most expensive part of this computation is the *snappyHexMesh* utility which is why it is the focus of the project. Due to the rules of CFD usage in F1, only focus of this project was to reduce the runtime of this utility, not its parallel efficiency.

Initial testing showed the code scaled very poorly, the time reduce until 48 processes but at 144-192 processes the program was significantly slower. Profiling identified decomposition as the bottleneck. The decomposition method used by Renault was the Scotch decomposition library. The primary focus of the project was to try and improve the Scotch decomposition library and explore other decomposition methods.

The first is the hierarchical method, which requires some user input for each unique case but the actual decomposition takes significantly less time but the load balance was worse and inconsistent. The second method explored was the KaHIP(Karlsruhe High Quality Partitioning) library, this library (along with the scotch) was built into OpenFOAM. This method was to be more configurable and could improve performance. The author also improved the current Scotch library by adjusting the *maxLoadUnbalance* value, this was also a parameter for KaHIP. The value was too low causing the decomposition to repeat so much that it increased the overall runtime.

The author found that KaHIP scaled the best with increased processors, followed by the hierarchical and then the improved Scotch. The author concluded to allow the F1 team to choose between KaHIP and Scotch, despite KaHIP performing the best and Scotch performing the worse, since there was more risk to switch to KaHIP and the Scotch times were improved over the baseline.

The author goes on to try and use vectorization to optimise the code more. This showed minimal improvement, upon further testing with wide vector units the author concluded that to benefit from vectorization the codebase would need changes beyond the scope of the project. The author then attempted to optimise the codebase in places they had identified earlier. They looked to switch from a standard parallel sorting algorithm to a parallel replacement and switched the communication from blocking to non-blocking. Unfortunately, both changes had negligible effects on performance.

2.2 Does the dissertation adequately explain the context in which the problem is set?

This dissertation explains the context of the problem very well. It describes why CFD is important and time sensitive for F1 teams (Chapter 1). It mentions that there are restrictions on CFDs to ensure even competition in F1. This restriction only applied to some uses of CFDs though and it restricts the amount of performance per core and by clock speed of each core and luckily this project does not fall under that restriction (2.1, 2.1.1). But the team needs rapid turnarounds times because even though the rules don't restrict them from running as many tests as they want, they have limited time between races to make adjustments (2.1.2).

2.3 Has the student adequately explained the scope of the problem, the approach that they would take, how success would be measured and reflected on the outcomes?

In sections 2.2 and 2.3 the author explains the structure of OpenFOAM and explains that the *snappyHexMesh* utility is the only one that has not been optimised, referencing to past literature, and that is why they are choosing to work on this. Furthermore, they explain that the primary goal of this project is to reduce the time taken by the program overall. Any efficiency goals are secondary, and they should not increase the time taken.

In Chapter 3 they explain how they hardware they are required to use as well as the profiling tools they will use to analyse the current codebase and the *checkMesh* utility they are using to find compare the current program with any modifications to ensure that the program still functions correctly overall.

In Chapters 4 the author explains their findings from profiling and what they identified as the primary bottleneck in *snappyHexMesh* and then they go on to explain their approaches to improve this in chapters 5, 6 and 7. Each experiment they conduct is first justified with theory and is every variable is explained including the number of times the tests are conducted. Every decision is also justified, and any changed due to experiment results is explained in the analysis. This makes the experiments very reproducible.

2.4 Quality of presentation.

The structure of the dissertation was of high quality, although the chapters are a little confusing at the beginning, each section explains what the information presented means for the entire project as well as the next steps that need to be taken and therefore what the next chapter will focus on. This made the reading of the dissertation much easier since it was less jarring moving from one section to the next and from one chapter to the next.

The sections are also very consistent in how they present experiments, there is always a section to explain the theory to justify the experiment that will be conducted. This is then followed by what the experimental conditions are, how many times the experiments are conducted and if there are any parameters being left unchanged as well as justification for this. The next section discusses the results and explains what this means for the project as well as any anomalies, any changes from the initial plan.

The figures were very clear, the axis labels and figure names were consistent throughout. The format of the graphs used were easy to read and consistent and were formatted to make it easy to read the data. One criticism would be that the images from the ARM map profiler were not very easy to read and the explanations for them were not always good enough to understand the conclusions the author drew.

The language was very formal and consistent, for example the tense of the verbs was always the same in each section of the report. In experiment plans the author always used active verbs to describe how things will be conducted (despite the experiments possibly taking place in the past) and in the analysis the verbs were in the past tense.

2.5 Presentation of additional information.

The references and citations in the text were always used to backup certain theory or about some sort of hardware specification. Furthermore, the author also mentioned when the experiments would go against some references and explain why this could be the case. The references themselves were in a standard format.

The tables and figures in the dissertation were listed along with the contents page, meaning there was a list of tables and figures and the pages they were on along with the list of chapters. This seemed odd at first but in the paper, it became clear that this was because all the figures and tables were not always in the places they were referenced. This is understandable in a long paper as there are references to figures in different chapters, however all the figures were always on the top of a page and would be oddly placed. For example, in section 6.1 the table for the results of a vectorisation experiment is placed in the middle of the description of the experiment itself. The analysis is in the next section, on the same page. This would be an area for improvement.

3. Background and Literature Review

When it comes to reducing power consumption there are several actors that have control over the methods that can be used to reduce power consumption, the user, the application developer, the library developers, and the centre that runs the supercomputer.

It is important to have a benchmark for measuring performance and, in the context of this report, the power consumption of a system. The High Performance Conjugate Gradient (HPCG) (Heroux, Dongarra, & Luszczek, 2021) benchmark is commonly used in HPC systems and, as the name suggests, benchmarks with matrix-vector operations and uses the conjugate gradient algorithm.

The most straightforward method of reducing power usage, is to limit the power provided to the CPUs at the cost of speed. Centres have the option to cap power and there have been various approaches to try and adjust power usages to lower power consumption. Borghes et al. proposed using machine learning to predict the power consumption of HPC machines as well as a batch system which could schedule based with some bounds on power consumption (Andrea Borghes, 2015). Power management systems have been proposed as well. The propositon by Cao et al (Thang Cao, 2016) which monitors the power usage of a system and adjusts power consumption according to a system budget to meet the requirements of the jobs that are being scheduled as best as possible.

When it comes to library developers, one of the most comprehensive sources is the Addressing Energy in Parallel Technologies (Adept) report on Power Usage of Parallel Algorithms and Parallel Programming Models (N. Johnson (UEDIN), 2016). This explores various parallel technologies and its impact on power consumption. One of their comparisons looks at the affect on power consumption based on matrix format, testing three types of matrix storage formats: co-ordinate format (COO), compressed sparse row format(CRF) and the sparse skyline format(SSS). The results suggest that having a different storage format has an affect on performance as well as power consumption, this would be a possible change from a library developer. HPCG, by default uses a format similar to CRF (Ao, et al., 2018).

Sell-C-σ is a data format proposed by (Kreutzer, Hager, Wellein, & Fehske, 2014) that suggests performance improvement over CRF. This is done by modifying another format, sliced-ELLPACK. The problem that ELLPACK looks to address is using SIMD instructions for better performance, CRF is not good for this, sliced-ELLPACK then groups the data into chunks, Sell-C-σ takes sliced-ELLPACK and then sorts some of the data in order of size. Experiments comparing the formats suggest better performance for Sell-C-σ and could lead to better energy efficiency.

Another method to consider in the same vein is the matrix-free method for matrix multiplciation, this would have to be a decision made by an application developer. Them matrix-free method is known to reduce the memory usage (N. G. Burago, 2018) and provide better performance by evaluating the evaluating the matrix-vector

product directly without forming and storing the matrix itself. This is especialy useful for very large matrices that a difficult to store. The possible power save here comes from being able to use less storage and therefore requiring less writing of data from memory to cache, which consumes energy.

The ADEPT report also looks at the comparison between MPI and openMP suggests that in the majority of cases, for programs that are as similar as possible in implementation in the MPI and openMP, MPI uses more energy, this would be a change decided by the user. We can also find that the programming languages themselves require different amounts of energy. In experiments conducted by (Rui Pereira, 2017) on 27 different well known languages, the energy usage, performance and memory usage were vastly different. C stood out as the best in terms of energy usage and performance and was third best in memory usage. Whereas even a language as popular as python was second worst in both performance and energy usage. It should be noted that performance and energy usage are tied. Since E = Pt(where E is energy, P is power and t is time) taking less time to complete a computation at the same power level will lead to better energy efficiency. When the data format or usage is affected, it is possible the processors, no longer being bottlenecked by the slow memory, will run fasters and use more power. This will lead better performance but more energy consumption.

4. Preliminary Investigation

Two sets of experiments were conducted on the Fulhame machine for preliminary investigation using the High-Performance Conjugate Gradients (HPCG) benchmarking tool. The first experiment was conducted to find what problem size exceeded the cache memory levels and began using RAM, by increasing the problem size and comparing the system's speed during benchmarking. The second experiment was to measure the power consumption of the system during the benchmark with increasing problem sizes.

The purpose of these experiments is to find at what problem size the system is forced to use RAM, causing more data to be transmitted during the benchmark. This allows us to measure the impact this has on the power consumption of the system.

4.1 Cache Levels Experiment

The experiment was conducted on the Fulhame machine to identify the problem size at which RAM is used along with the cache levels. A single process is used to run the HPCG benchmark with a 3D problem of size starting at 16 (this means 16,16,16), which is the minimum allowed by default, and incrementing this by 8 until the memory usage exceeds the available cache memory.

Each test is conducted three times and then averaged to account for anomalies and to account for the system being under different loads at different times.

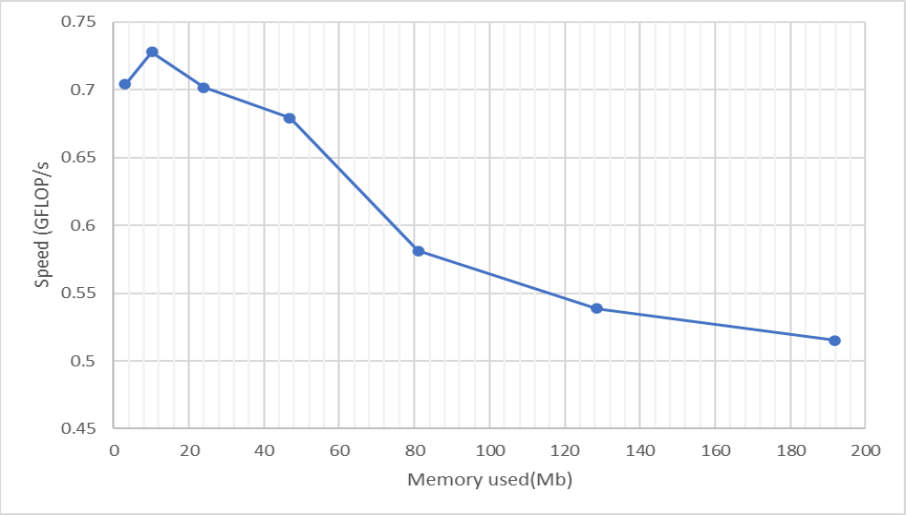

Figure 1 : Impact of speed as prolem size is increased

Cache Levels Analysis

Ideally, the experiment would allow us to find the point of every cache level, but even at the minimum problem size of 16, this was using L3 cache. The point where we see the performance drop was for problem of size 48 or when the memory usage is about 80 MB. When the problem size is 40, the majority of the 46MB of memory usage fits in

the 42MB of cache available (WikiChip, 2019) with very few cache misses. At this point, the memory requirement is double that of the cache memory available and so the reading and writing from RAM to cache happens more frequently. Identifying this threshold will help guide the setup of future experiments and help interpret the results.

4.2 Memory Power Consumption Experiments

The next experiment was to try and find what the impact of the system needing to use more memory to story the data. To test this, experiments were conducted on Fulhame with increasing problem sizes, again, and the tx2mon package was used to retrieve the power consumption during an experiment.

The code snippet below is an example of the experiments were setup, the results are stored in a .csv file (pwr.csv below). When tx2mon starts recording, the command `sleep 10` is used and again after the benchmark is run. This way the first 10s and the last 10s of the results can be ignored and the increase and decrease in power consumption helps identify at which point the benchmark is running.

```
sudo $(which tx2mon) -f pwr.csv &

                sleep 10

          mpirun -np 64 $hpcg --nx=16 --rt=60

                sleep 10

         sudo /bin/pkill tx2mon
```

The tests were conducted with 1 process and 64 processes and in each experiment the problem size was doubled in one of the dimensions. The codebase was adjusted to allow for smaller problem sizes as well, the smallest problem is of size 4x4x4 then 8x4x4, then 8x8x4 and so on until 64x64x64. Each set of tests is conducted three times to account for anomalies and volatile system loads.

The problem is set to run for 60s with a 10s wait on either side. This is so that it is clear in the results when the system has started running the benchmark as the CPU power usage will increase. The benchmark is such that it will keep running for 60s and then output the average speeds during that time.

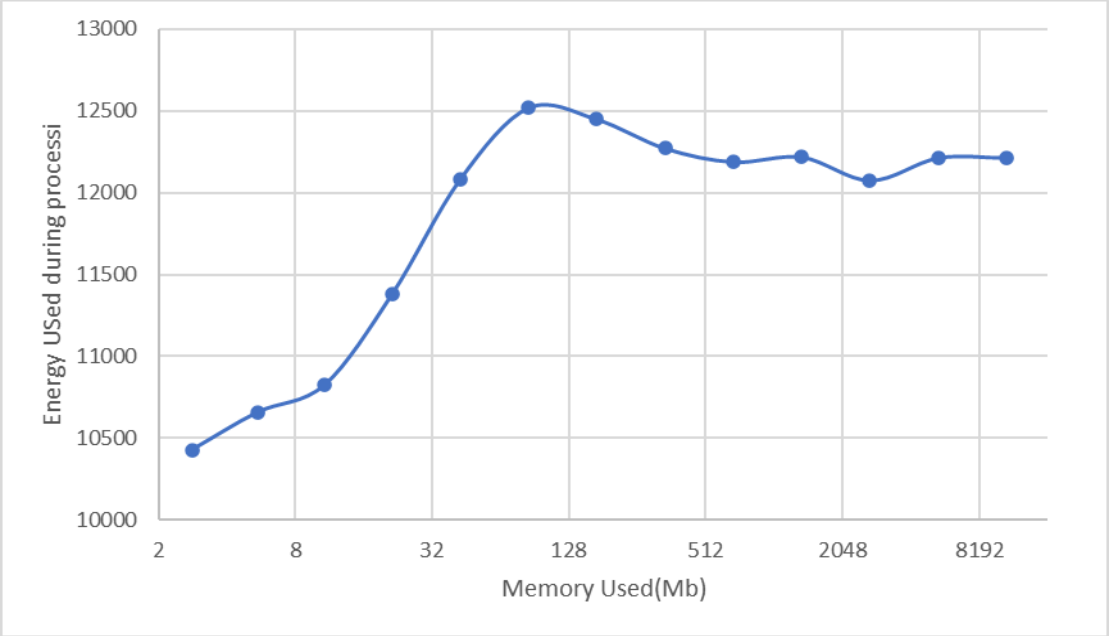

Figure 2: Energy Consumption as memory usage is increased on 64 processes

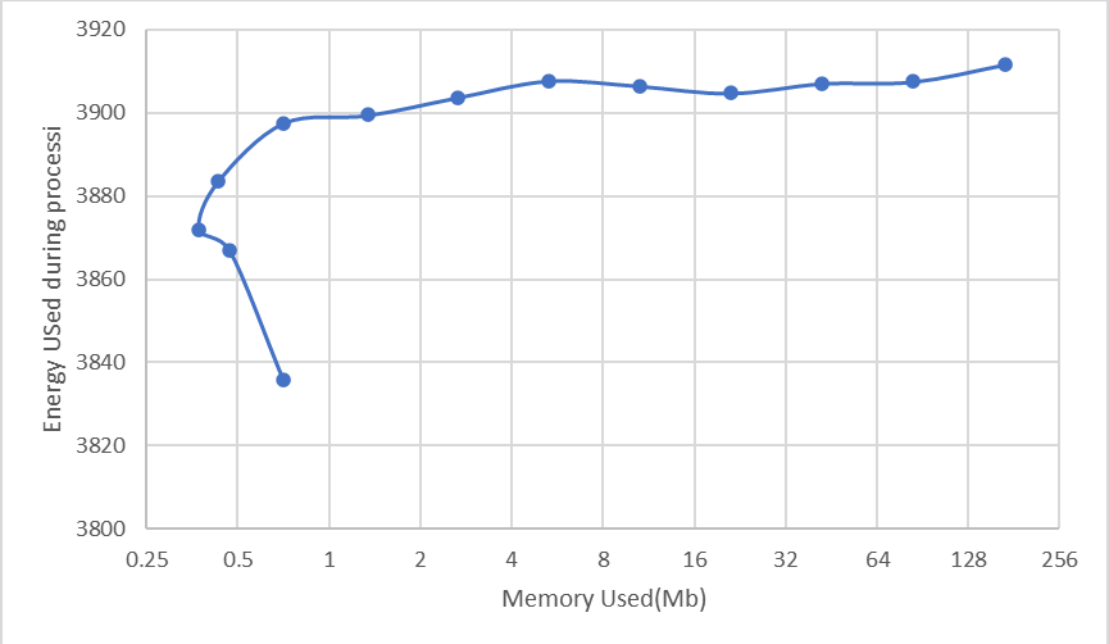

Figure 3: Energy Consumption as memory usage is increased on 1 process

Energy Usage Analysis

The results in Figure 2 are consistent with what is expected from the cache level experiment. The energy usage increases as the memory usage increases, we can see that at the beginning of the graph, the energy usage is rising rapidly. At 80Mb, the system reaches its peak energy consumed, then the energy usage reduced very slightly but remains close to the same amount throughout.

This is because at this point the problem is far larger than the cache storage available on these chips and has mostly being stored in RAM, the CPU is running at its peak clock speed and the volatility is caused by the changes in DRAM power consumption.

The results for the experiments conducted with 1 process, in Figure 3, had some unexpected results. The line does seem strange, this is because the first data point, which used problem size of 4x4x4 used approximately 0.71MB, the problem of size 8x4x4 used just 0.47MB. This was unexpected since a larger problem was expected to use more memory, and as the problem sizes increase, Figure 3 shows that this is the case.

HPCG has a minimum problem size of 16x16x16 and some changes were made in the code to allow larger sized problem, it is possible that the method of recording memory usage works incorrectly when the problem size is below that threshold and that is what skewed the results. The trend, however, was similar to the 64 process problem where there is a steep increase in energy usage and then the energy usage is very similar as the problem size increases.

5. Final Project Proposal

2.6 Proposal

Supercomputers. This project aims to find possible methods of making these systems more power efficient without acquiring better hardware and how much this would affect the performance of the system.

There are a few approaches that could result in better energy efficiency. One of the easier methods to test is the use of MPI or openMP or a combination of the two and compare their impact on performance and energy efficiency. The primary focus of this project will be to implement the matrix-free method, this directly reduces the memory usage of the system and is expected to improve performance and power efficiency, the experiments will show us if there is a reduction in energy efficiency as well. The primary focus of this project will be to look at data layout; transferring data uses power.

A secondary objective would be to try and implement different data layouts, the HPCG benchmark uses a storage format like compressed row storage (CRS). A data format that would be beneficial experiment with is Sell-C-σ, it has been shown to possibly have better performance than CSR, which the adept report suggested was an improvement over the inefficient COO and could lead to better energy efficiency.

A possible extension to this, would be to test the affects of performance and energy efficiency by adjusting CPU clock speeds and under-volting. There would be an expected loss of performance with lower power consumption, but this could still lead to better energy efficiency.

The overall objective of testing these various methods is to find which stake holder can most affect the energy efficiency of the system. The decision of openMP or MPI is controlled by the user, optimisations such as the matrix-free method would be controlled by the application developer, the data storage format is controlled by the library developer and the CPU clock speeds are controlled by the system administrators.

The experiments are all conducted using the HPCG benchmark, this uses the conjugate gradient method as a benchmark, which is a commonly used computation. Although the testing is on the conjugate gradient method, it is a benchmarking tool because of the wider implications it can have and so it is expected that the findings will have a wider impact as well.

2.7 Hardware Requirements (or similar)

The main hardware requirement for this project is a machine that can measure the power usage during benchmarks. There are two machines available for this purpose, Fulhame (EPCC, n.d.) and NextgenIO (Nextgenio, n.d.).

At this stage access to both machines has been acquired and some tests have been run on Fulhame already. Furthermore, access to the Tx2Mon package within Fulhame has been acquired, this is the package needed for power measurements. There is limited storage needed, just enough to store the benchmarks (in order of megabytes).

Additional costs inferred? – run tests and judge how long the tests will take and how many need to be conducted

6. Workplan

The project stage starts on Friday May 21st and ends on Friday August 20th, to ensure that the project is completed, the following workplan has been proposed. The matrix-free implementation is expected to take less time to implement, hence it is the primary focus which is why it will be implemented early on to ensure a minimum viable product in the event of unforeseen delays. The dates in the table refer to the beginning of the week it represents.

Task	24-May	31-May	07-Jun	14-Jun	21-Jun	28-Jun	05-Jul	12-Jul	19-Jul	26-Jul	02-Aug	09-Aug	16-Aug
Test User Settings (openMP/MPI)													
Implement Matrix Free Method in HPCG													
Complete Analysis of User Settings													
Run Jobs Tesing Matrix Free Implementation													
Implement Data Formating Changes for Matrices													
Complete Anaysis of Matrix Free Tests													
Write-up Results Obtained So Far													
Rerun Tests As Needed													
Run Experiments to Test Data Formating Changes													
Complete Analysis of Data Format Tests													
Complete First Draft of Final Report													
Project Presentation Preparation													
Complete Final Report													

Figure 4: Gantt chart representing the project plan

7. Risk Analysis

7. 1 Success Criteria

The aim of this project is to find which party is most responsible for the energy efficiency of HPC systems: the user, application developer, library developer or system administrator. The testing of user settings, such as MPI and openMP is straightforward to test. The application developer and library developer approaches are focused on adjusting the amount of memory used by the system during a benchmark. Preliminary investigation has helped find the threshold at which RAM is used during computation for the HPCG benchmark and it also demonstrates the increase in power consumption when this threshold is reached.

For the application developer, the matrix free method is being tested and for the library developer, adjustments to data storage formats are being tested. Ideally these would both be implemented and tested successfully so that a comparison can be made between the influence of the user, application developer and library developer.

Possible fallbacks

The minimum viable product is to have the matrix free method implemented and tested as well as the user settings tests with openMP and MPI. These results can then be compared so we can compare the user's impact with that of the application developers. The data format changes are expected to be more complicated, and it is possible they are not completed at all, so they are in the plan but not a necessity for the report.

Extensions

The stake holder that has not been mentioned is the system administrator. The experiments that could be conducted for this are adjusting CPU clock speeds to save power at the cost of performance. At this stage access to the required tools has not been confirmed, if there is enough time then this could be an extension to the project that is explored.

7.2 Risks and Mitigations

The main risk for this project is the availability of tools to measure power usage during experiments. Access to two different machines that can provide energy readings have been acquired, Fulhame and NextGenIO. The plan is to compare the performance on the two machines, but they are also a backup in case one of the machines becomes unavailable. If both machines become unavailable, Archer2's CrayPat profiler is also capable of providing energy readings, access will have to be acquired in this event.

Furthermore, if there is no method acquiring power readings available within the timescale of this project then it can be shifted to a performance comparison on the machines available of the matrix free method and the data format changes.

8. Outline of the Dissertation Report

The planned Structure of the dissertation report is as follows:

1. **Introduction**
2. **Background and Literature Review**
3. **Methods**
   - 3.1 *User settings*
   - 3.2 Matrix Free multiplication
   - 3.3 Alternative Matrix Storage
4. **Results and Analysis**
   - 4.1 User settings
   - 4.2 Matrix Free multiplication
   - 4.3 Alternative Matrix Storage
5. **Conclusion**
   - 5.1 Project Evaluation
   - 5.2 Future Work

References

Ao, Y., Yang, C., Liu, F., Yin, W., Jiang, L., & Sun, Q. (2018). Performance Optimization of the HPCG Benchmark. *Transactions on Architecture and Code Optimization*, *15*(1). Retrieved from https://dl.acm.org/doi/pdf/10.1145/3182177

Borghes, A., Bartolini, A., Lombardi, M., Milano, M., & Benini, L. (2015). Scheduling-based power capping in high performance computing systems. *Principles and Practice of Constraint Programming*, 524-540.

Burago, N. G., & Nikitin, I. S. (2018). Matrix-Free Conjugate Gradient Implementation of Implicit Schemes 58. *Computational Mathematics and Mathematical Physics*, 1247-1258.

EPCC. (2019). *Fulhame*. Retrieved from EPCC: https://www.epcc.ed.ac.uk/facilities/other-facilities/fulhame

Heroux, M., Dongarra, J., & Luszczek, P. (2021). *HPCG-Benchmark*. Retrieved from HPCG-Benchmark: https://www.hpcg-benchmark.org/index.html

Johnson, B. (2009). *Supercomputer in the firing line over carbon footprint*. (The Gaurdian) Retrieved 03 12, 2021, from https://www.theguardian.com/technology/blog/2009/aug/28/met-office-supercomputer

Kreutzer, M., Hager, G., Wellein, G., & Fehske, H. (2014). *A UNIFIED SPARSE MATRIX DATA FORMAT FOR EFFICIENT GENERAL SPARSE MATRIX-VECTOR MULTIPLY ON MODERN PROCESSORS WITH WIDE SIMD UNITS*. Retrieved from https://arxiv.org/abs/1307.6209

N. Johnson (UEDIN), M. W. (2016). *Addressing Energy in Parallel Technologies.*

Nextgenio. (2021). *NextgenIO*. Retrieved from http://www.nextgenio.eu

Rui Pereira, M. C. (2017). *Energy efficiency across programming languages: how do energy, time, and memory relate?* the 10th ACM SIGPLAN International Conference.

Sönnichsen, N. (2020). *Statista.* Retrieved 02 12, 2021, from https://www.statista.com/statistics/263492/electricity-prices-in-selected-countries/

Thang Cao, Y. H. (2016). Demand-Aware Power Management for Power-Constrained HPC Systems. *16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, 21-31.

Top500. (2010). *November 2010*. Retrieved from https://www.top500.org/lists/top500/2010/11/

Top500. (2013). *June 2013*. Retrieved from Top500: https://www.top500.org/lists/green500/2013/06/

Top500. (2020). *November 2020*. Retrieved from Top500: https://www.top500.org/lists/top500/2010/11/

Tynan, D. (2020). *Six ways the world's fastest computers have changed your life*. (Hewlett Packard Enterprice) Retrieved 03 10, 2021, from https://www.hpe.com/us/en/insights/articles/supercomputers-in-everyday-life-2011.html

U.S. Energy Information Administration. (2020). *How much carbon dioxide is produced per kilowatthour of U.S. electricity generation?* Retrieved 03 15, 2021, from https://www.eia.gov/tools/faqs/faq.php?id=74&t=11#:~:text=In%202019%2C%20total%20U.S.%20electricity,of%20CO2%20emissions%20per%20kWh.

WikiChip. (2019). *ThunderX2 CN9980 - Cavium*. Retrieved from WikiChip: https://en.wikichip.org/wiki/cavium/thunderx2/cn9980