Getting the most out of your NeSI allocation

Chris Scott, Anthony Shaw

chris.scott@nesi.org.nz, anthony.shaw@nesi.org.nz

Outline

- What can you do to get the most of your NeSI allocation?
  - Improve your own job efficiency

- What does NeSI do to help you get the most out of your NeSI allocation?
  - Job Efficiency Reviews
  - Research Software Engineering Consultancy

Improving job efficiency

Utilisation

- Using the resources you request
- Requesting the optimal amount of resources

Performance

- Improving the efficiency of your code
- Improving how your jobs use available resources

Determining if your jobs are efficiently utilising resources

```
sacct
         JobID      JobName     Elapsed     TotalCPU Alloc   MaxRSS      State
-------------- ------------ ----------- ------------ ----- -------- ----------
1234567        TestJob         00:03:29    13:48.330     8           COMPLETED
1234567.batch  batch           00:03:29    13:48.330     8  249328K  COMPLETED
1234567.extern extern          00:03:29     00:00:00     8        0  COMPLETED
```

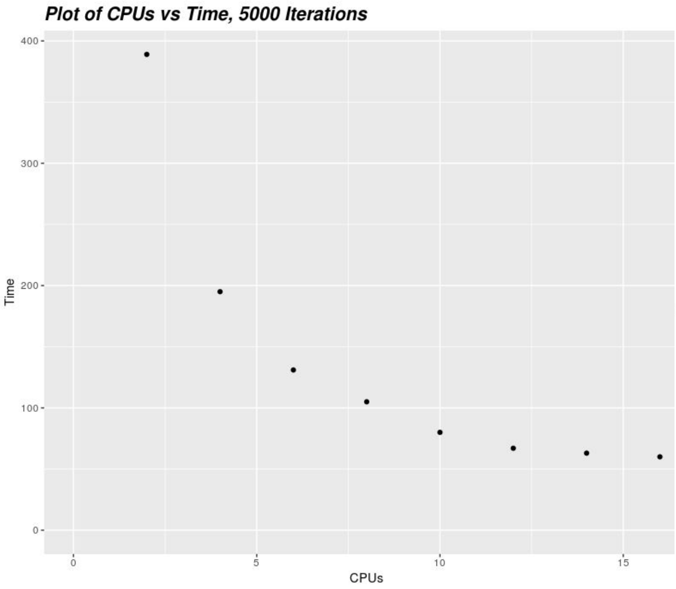
```
nn_seff 1234567
Job ID: 1234567
Cluster: mahuika
User/Group: user123/user123
State: COMPLETED (exit code 0)
Cores: 4
Tasks: 1
Nodes: 1
Job Wall-time:  34.83%  00:03:29 of 00:10:00 time limit
CPU Efficiency: 49.52%  00:06:54 of 00:13:48 core-walltime
Mem Efficiency: 47.56%  243.48 MB of 512.00 MB
```

Greater than 100% utilisation with nn_seff

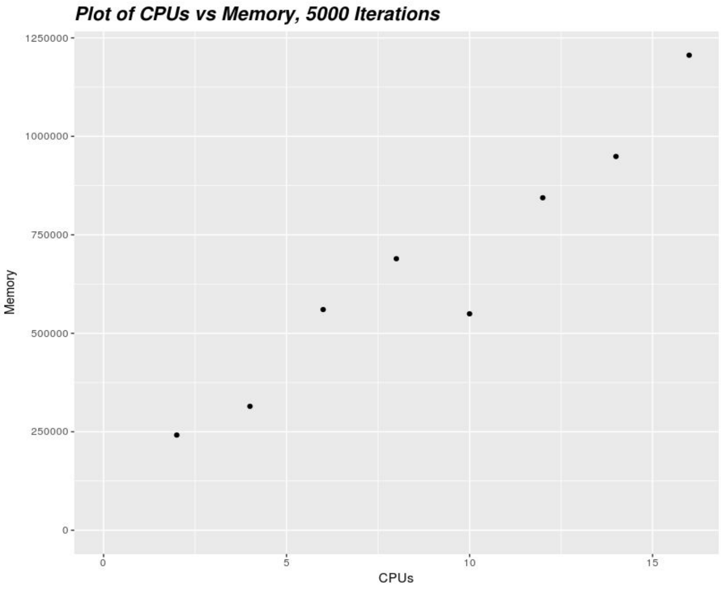
```
sacct
       JobID    JobName    Elapsed   TotalCPU Alloc   MaxRSS      State
------------ ---------- ---------- ---------- ----- -------- ----------
2345678      TestJob      00:03:03  11:54.940     4          COMPLETED
2345678.batch batch       00:03:03  11:54.939     4  411656K COMPLETED
2345678.extern extern     00:03:03   00:00:00     4        0 COMPLETED
```

```
nn_seff 2345678
Job ID: 2345678
Cluster: mahuika
User/Group: user123/user123
State: COMPLETED (exit code 0)
Cores: 2
Tasks: 1
Nodes: 1
Job Wall-time:  30.50%  00:03:03 of 00:10:00 time limit
CPU Efficiency: 195.36%  00:11:55 of 00:06:06 core-walltime
Mem Efficiency: 78.52%  402.01 MB of 512.00 M
```

How to scale your jobs



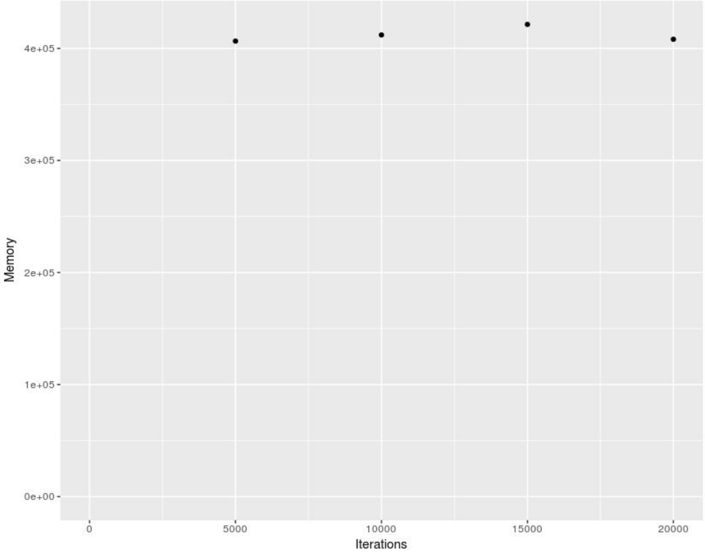
Plot of CPUs vs Time, 5000 Iterations



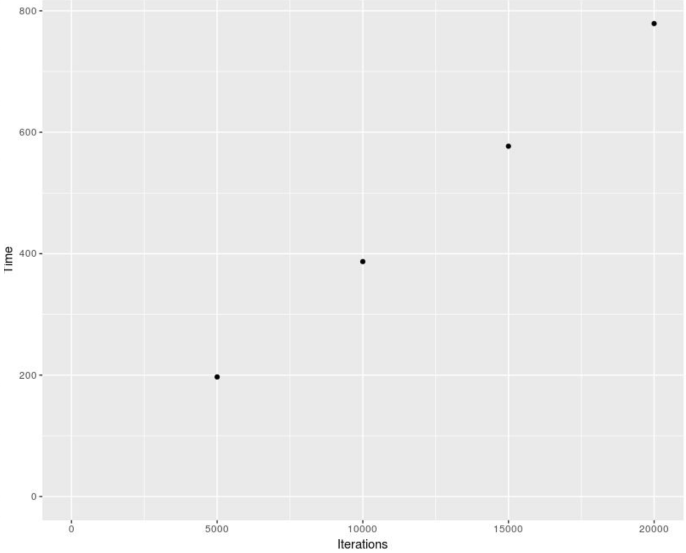
Plot of CPUs vs Memory, 5000 Iterations

How to scale your jobs

Plot of Iterations vs Memory, 4 CPUs

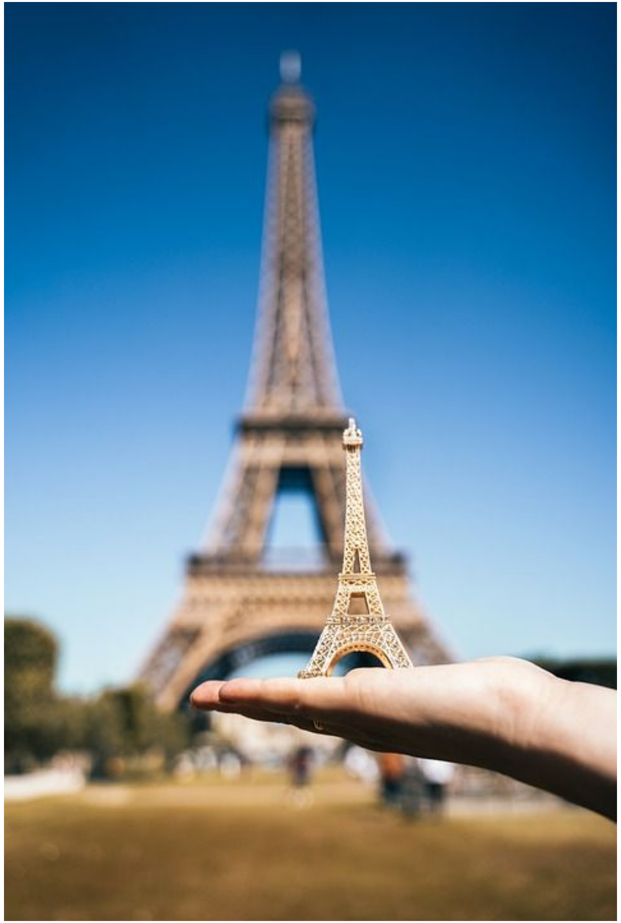


Plot of Iterations vs Time, 4 CPUs



Why should you scale your jobs

- Avoid job failure due to running out of resources
- Jobs usually don’t scale linearly and assuming they do will result in incorrect estimates
- Improve job queue time
- Resources you waste are resources other researches can’t use
- You can find a more details explanation of how to scale your jobs here: https://support.nesi.org.nz/hc/en-gb/articles/360000728016



How NeSI can help

NeSI Job efficiency reviews

- We have many projects with large allocations
- Worthwhile spending some time to “sanity check” the jobs from these projects
  - Even a small efficiency improvement could have a meaningful impact
- Job efficiency reviews
  - Compiling, running jobs, tweaking runtime options, …
  - Up to 20 hours
  - Looking for any quick wins or whether a larger project is worthwhile (see consultancy service)

https://support.nesi.org.nz/hc/en-gb/articles/360002327275

Computational Science Consultancy

Team of Research Software and Data Science Engineers, projects can last for 100s of hours…

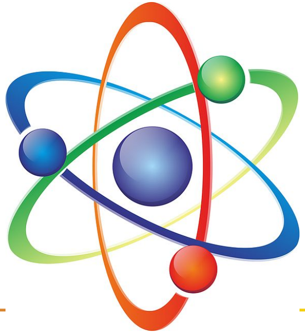
https://www.nesi.org.nz/services/consultancy

- Improve efficiency
- Faster time to solution
- Custom code development
- Data Science Engineering
- Bespoke solutions

Contact support@nesi.org.nz

Example - runtime tuning for VASP

- Enabling OpenMP at run time doubled the efficiency (twice as fast on same resources)
- 5-10 hrs effort, potentially saving millions of core hours
- Not transferable to other VASP users – highly dependent on the specific type of simulation, etc
- Didn’t reduce core hour usage – researcher just did more work in the same time



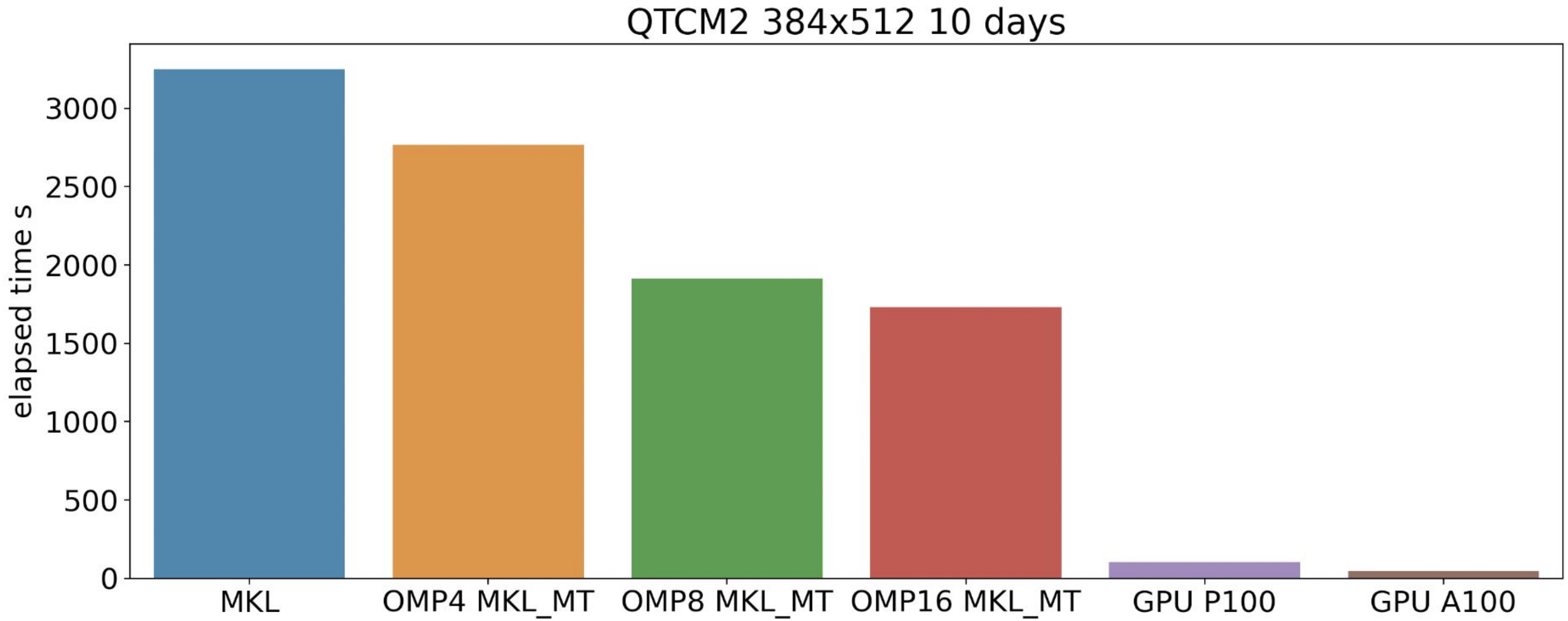
New Zealand Earth System Model optimisation

- Simulations were going to be very expensive
- Wolfgang Hayek worked on a consultancy project
- Tweaking compiler options
  - Optimised build vs debug, vectorisation, ...
- Optimising domain decomposition
- Outcome
  - ~30% faster and ~7x reduction in core hours

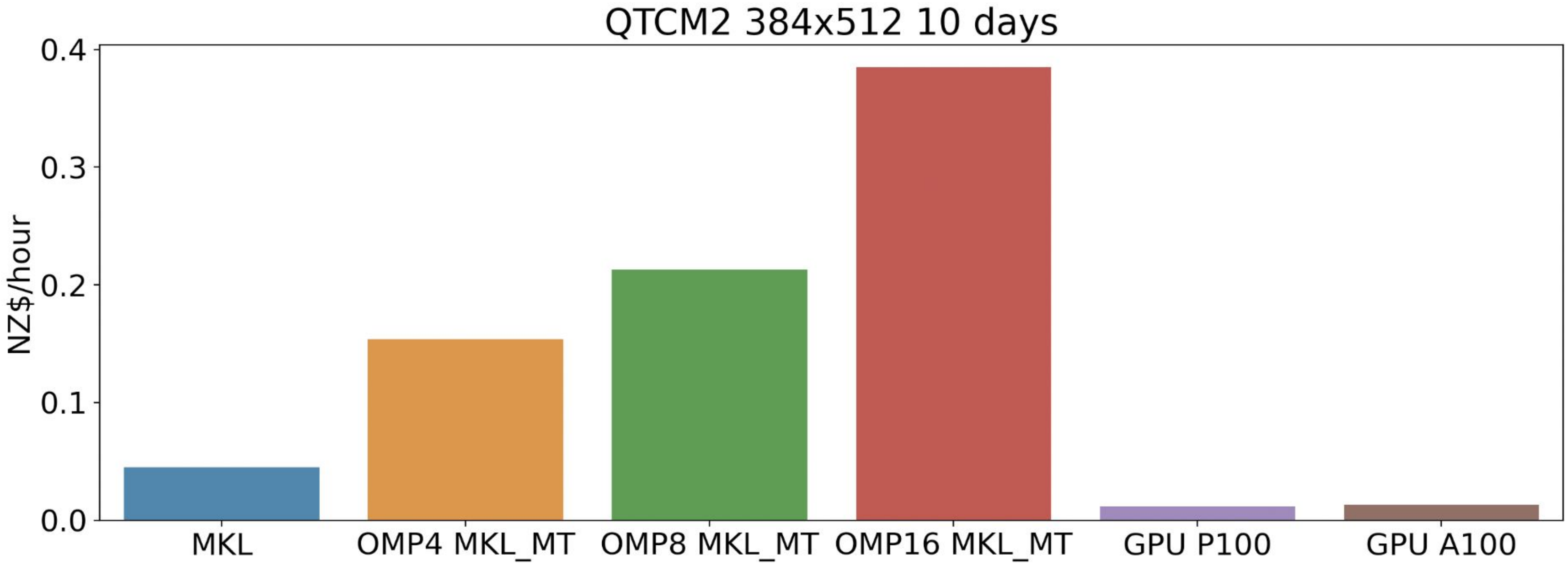
Example - porting to GPU

- Alex Pletzer led a consultancy project with Gilles Bellon (UoA), convert Fortran code to run on GPUs
- ~100 hrs effort, consultancy project
- Used OpenACC to offload computations to the GPU
  - key was data locality - making sure the data stays on the GPU vs copied back and forth

GPU version 66x faster than a single core



GPU version is 71-74% cheaper to run than single core version



Summary

Summary

- Utilisation: Make sure you are using what you ask for and asking for what you need
- Performance: Make sure you are making use of what is available and making the most of what you have
- Invest time early to make sure things are efficient and running well
- Job efficiency reviews and consultancy projects
  - Spend some time to sanity check big projects
  - RSE consultancy service available for bigger pieces of work: https://www.nesi.org.nz/services/consultancy
- Contact us by emailing support@nesi.org.nz