Exascale for mid-scale applications ?

- Simon Burbidge (DiRAC) sburbidge@btopenworld.com
- DiRAC Director Prof Mark Wilkinson (DiRAC) miw6@leicester.ac.uk

- ISC 2024

About DiRAC

- https://dirac.ac.uk
- Funded by STFC, UKRI and DSIT
- Supports research in the STFC theory community (particle physics, astrophysics and cosmology, solar system physics, particle astrophysics and nuclear physics) and the STFC science challenges https://www.ukri.org/publications/stfc-science-challenges/
- Compute services co-designed around science workflows

The DiRAC HPC Facility

HPC for theoretical astrophysics, cosmology, particle & nuclear physics in the UK

Memory Intensive “COSMA 8” (Durham)

DELL EMC

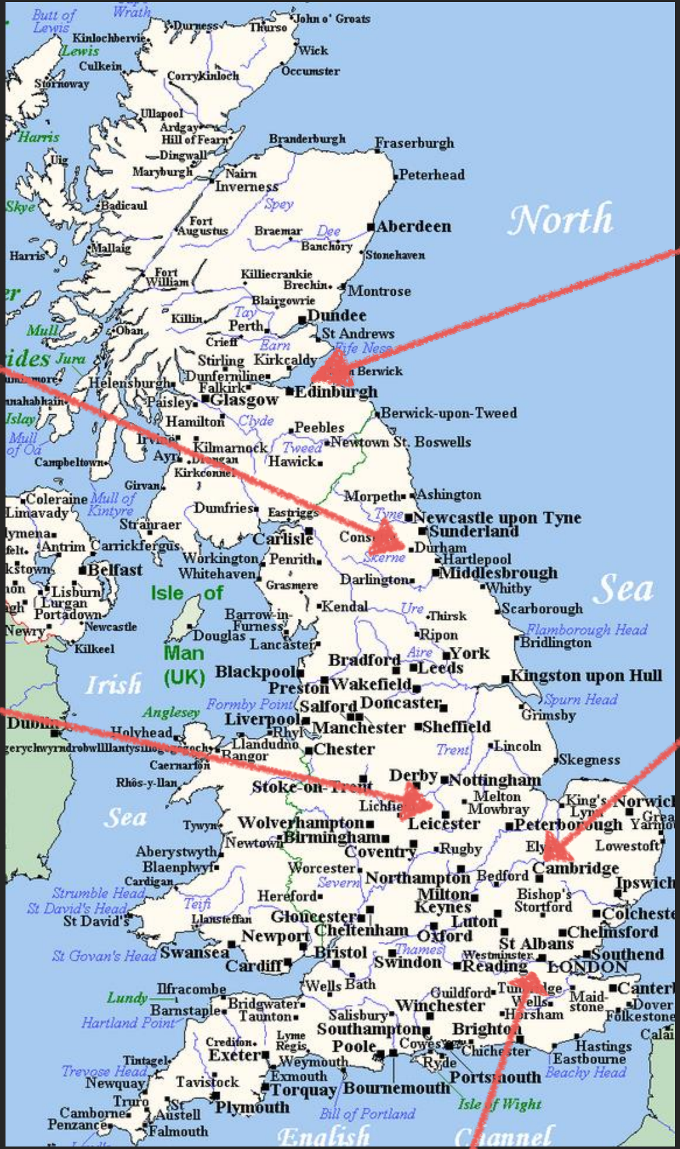
528 TB RAM
Large-scale cosmological simulations

Extreme Scaling “Tursa” (Edinburgh)

Atos

GPU-based system

Large lattice-QCD simulations



Data Intensive “CSD3” (Cambridge)

DELL EMC

Heterogeneous architecture for complex simulation and modelling workflows

Data Intensive “DIaL” (Leicester)

Heterogeneous architecture for complex simulation and modelling workflows

Hewlett Packard Enterprise

Project Office (UCL)

ExCALIBUR

- UK Exascale programme lead by the Met Office and UKAEA, with UKRI
- https://excalibur.ac.uk for all the details
- Comprised of a large number of projects under 5 themes.
- Projects are widely distributed across UK Universities and Research Organisations, many are collaborative across multiple institutions
- One more year to run on project

Lattice QCD success

- Code is called GRID – well established.
  Has been ported very successfully to GPU . But took several personyears of work to do this. Both RSEs and considerable collaboration with code author and group (Prof Peter Boyle, Antonin Portelli, James Richings University of Edinburgh). Extensive refactoring of the code was needed to get good performance on the GPUs, which required detailed knowledge and understanding of the code and algorithms used.
- This code is widely used and runs at large scale.

Mid-scale applications

- Many HPC runs are not of the exascale, and don’t require large proportions of our HPC systems to deliver results.
- We kind of expected that exascale benefits would trickle down through to mid- and small-scale applications
- *However:*
- We need to use accelerators to achieve exascale and :
- Many are mid-scale codes are CPU based and the code owners ask why would we want to port them to accelerated systems? They are not motivated to do the porting. Faster runs may not be of great value to them, as they make many runs in each study, that can be run concurrently.
- Is there anything we can do?

So can exascale benefit these users?

- More efficient runs in terms of power consumption to due to newer processors
- Potential speed up due to these new processors or more cores per node/fewer nodes per run
- More (and different) accelerators now available
- New tools to support all the different hardware types
- But - this won’t make their code more scalable though … still need to do hard work to scale up

How can we take advantage - software?

- Wide parallelization – its easy, for some apps we haven't tried yet, but there have to be inherent characteristics like huge matrices/enormous loops
- New tools, (eg kokkos , scyl, psyclone, compilers) can generate code for different accelerators, so if you can use these you have a choice of platforms, where cost and performance may be different for your runs. And you can switch platforms more easily.
- Use Libraries . If your code uses libraries, then there may be implementations that run on different accelerators, making it easier to transfer your code. In particular domain specific libraries may give you a notable (niche) benefit.

How can we take advantage – hardware?

- Some runs, traditionally using single HPC nodes, may fit on a single accelerator node – this is a sweet spot, not requiring a hefty interconnect or use of MPI or similar to partition the runs.
- New interconnect technology can give benefits. MPI collectives in hardware, scatter gather technology and better block transfers.
- Unified memory. In certain apps, the availability of a unified memory address space allowing access from one node to the memory of another may expose programming models supporting large memory and easy distribution of work (avoiding MPI)

Learn from the past

- A modern CPU node can have 256 or more cores per node– this is comparable capability to a 128 node $20m SGI Origin 2000 system of 20+ years ago!
- So each HPC Cluster CPU node is now a large SMP, a new kind of hybrid – didn’t we dream of these in the past?
- Also as well as being an SMP, these nodes are MIMD architecture, rather than the SIMD we have on accelerastors - again, in the past didn’t we dream of this?

So, where are we now?

- We don’t have the Holy Grail!
  (ie exascale for everyone)

We do have

- More powerful hardware
- More types of different hardware to choose from
- New tools for generating code on different platforms
- Still hard work to use multiple nodes in most cases

Thank you

- Simon Burbidge
  sburbidge@btopenworld.com
  www.dirac.ac.uk