Durham University
Department of Computer Science

Large-Scale Computing

Recent research challenges in large-scale computing at Durham

T. Weinzierl

May 2, 2023

Large-scale computing as tool

Durham University
Department of Computer Science

Large-scale computing infrastructure (kit and people behind it) is **the** tool behind a lot of our research. However . . .

- programming is something every PhD student and PDRA does anyway;
- our code is so complex and powerful that only experts can maintain and extend it;
- we have to deliver new science not code.

> "Before the great discovery was the creation of a new tool!"
>
> [C. Johnson (SCI): "The Golden Age of Computing"]

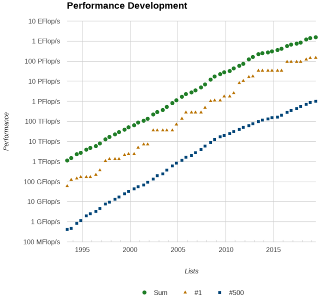
Creating the infrastructure (software and hardware) is research topic of its own*

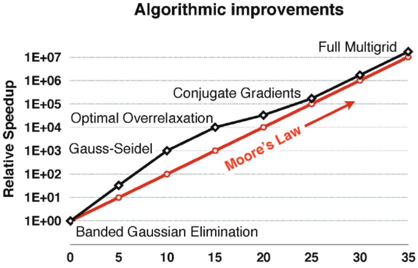
- Scientific codes are no byproduct (anymore)
  (traditional "by domain science"-approach has navigated lots of codes into dead end)
- Focus on tool is timely and urgent research
  (traditional codes do not become faster anymore)

* Cmp. UKRI's ExCALIBUR initiative, US ECP initiative, Europe's EuroHPC initiative and The President's Information Technology Advisory Committee (PITAC)

The two facets of “better tools”

Durham University
Department of Computer Science



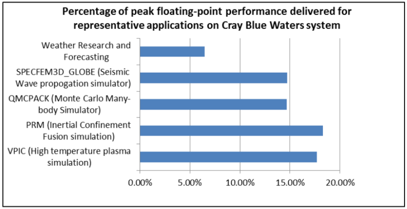


- Left: Machine (hardware) capabilities grow exponentially (`www.top500.org`, 2019-09-07)
- Right: New algorithms, numerics, algorithms contribute exponential capabilities (Ulrich Rüde, Karen Willcox, Lois Curfman McInnes, Hans De Sterck: *Research and Education in Computational Science and Engineering* SIAM Rev. 60-3 (2018), pp. 707–754)
- Third dimension of growth within application domains (not shown)

⇒ Facets **used to** team up

Efficiency vs. effectiveness

Durham University
Department of Computer Science



* Leland et al: *Performance, Efficiency, and Effectiveness of Supercomputers*. SANDIA report SAND2016-3730

- LinPack (matrix calculations) and similar mathematical core routines use 80-90% of machine potential
- Well-tuned scientific computing codes use 10–15% on average (HPCG only 2%, FFT only 1%)
- Most codes use way below 1% (includes ML)

> "There is substantial and well-founded concern that this trend [invariant achieved peak per machine generation] is now at risk due to the erosion of Moore's Law." *

$\Rightarrow$ Rethink algorithms for each new hardware generation

$\Rightarrow$ Tune codes

The ethical dimension

Durham University
Department of Computer Science

Consumption	CO_2e (lbs)
Air travel, 1 passenger, NY↔SF	1984
Human life, avg, 1 year	11,023
American life, avg, 1 year	36,156
Car, avg incl. fuel, 1 lifetime	126,000
Training one model (GPU)	
NLP pipeline (parsing, SRL)	39
w/ tuning & experimentation	78,468
Transformer (big)	192
w/ neural architecture search	626,155

Table 1: Estimated CO_2 emissions from training common NLP models, compared to familiar consumption.[1]

* Emma Strubell, Ananya Ganesh, Andrew McCallum: *Energy and Policy Considerations for Deep Learning in NLP*. 57th Annual Meeting of the Association for Computational Linguistics (ACL), 2019

- Electricity bill exceeds procurement cost
- Training an AI model has CO_2 impact of small car
- We waste more and more energy of more and more requested energy

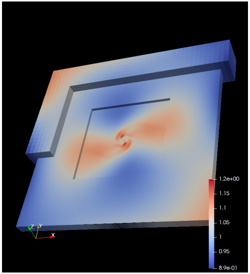
⇒ Compute/train only what's necessary

⇒ Reduce machine power envelope; done by others but . . .

⇒ Improve efficiency

Example: Binary black hole mergers

Durham University
Department of Computer Science



Joint work with Han Zhang and Baojiu Li (ICC).

- Code is too complex
  (memory layout, IEEE precision, map onto MPI, . . .)
- Hardware heterogeneity
  (GPGPUs and Intelligent NICs)

⇒ Implementation challenges

- Time step sizes constrain switch to finer resolution
- Many shots for UQ (off-topic here)
- . . .

⇒ Algorithmic challenges

Code complexity

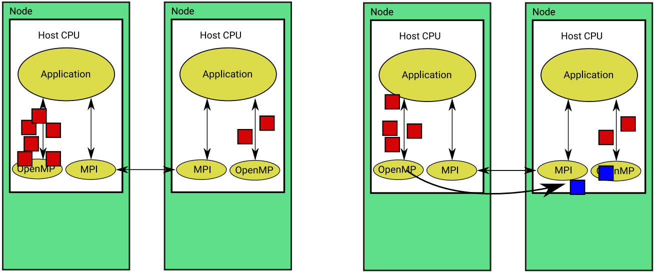
Durham University
Department of Computer Science

```
struct peano4::grid::AutomatonState {
  public:
    AutomatonState();
    void   flipEvenFlags(int index);

    [[clang::map_mpi_datatype]]
    static MPI_Datatype  getForkDatatype();
  private:
    [[clang::pack_range(0,63)]]
    int    _level;

    [[clang::truncate_mantissa(23)]]
    double    _x[Dimensions];
};
```

- Extend existing language
  (no introduction of new language, no rewrite, semantics-preserving)
- Language reduced to *what* to do
  (augmentation describes *how* to realise things)
- Migrate performance engineering to compiler

Heterogeneity

Durham University
Department of Computer Science



Tasks are automatically migrated between various nodes through NVIDIA BlueField NICs
(joint work with NVIDIA and A. Basden, DiRAC)

- Phrase algorithm in tasks
  (state-of-the-art paradigm nowadays)
- Identify tasks that fit to particular machine types
  (declarative rather than re-writes for machine parts)
- Write algorithms which handle task placement
  (GPUs via OpenMP with NVIDIA; GPUs via SYCL with Intel; SmartNICs with NVIDIA Networking)
- Optimise scheduling
  (on-the-fly vs. offline)

Time step sizes

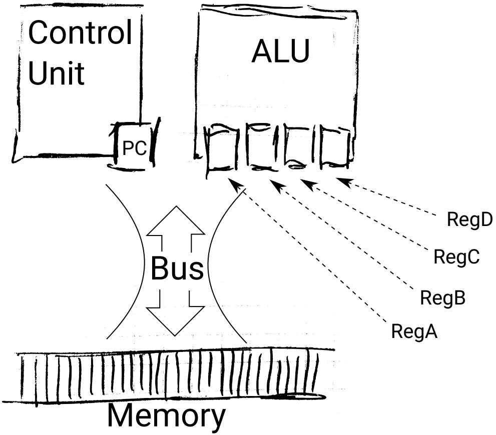
Durham University
Department of Computer Science

- Optimistic time stepping
  (bring tasks forward and hope that it turns out that they were valid)
- Implicit time stepping plus multigrid

$\Rightarrow$ t.b.h. we don't know yet what to do

The root of all evil: von Neumann

Durham University
Department of Computer Science



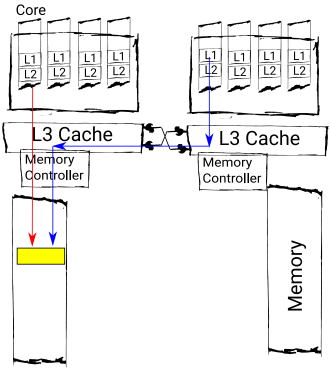
- Machine is dominated by calculations
- We just scale up the machine or
- We just make machine faster

⇒ Model is wrong

(always has been wrong, but that did not bother us previously)

Modern machines

Durham University
Department of Computer Science



- Speed depends on where the data sits
- Data movements are costly

  (time and electricity)

> Irony: We use a machine to model more precisely and correctly. Our model of that machine however is fundamentally inappropriate and even wrong.

Showstoppers and open problems—my summary

Durham University
Department of Computer Science

1. Right training (C and/or Fortran) has disappeared thanks to data science hype
   (both in CS, Maths and domain sciences)
2. Computer scientists solve toy problems, domain scientists stick to outdated algorithms
   (interdisciplinary often only sales pitch)
3. Hardware heterogeneity not solved but pushed due to machine learning success
   (pressure to migrate had historically not been high enough)
4. Some hard fundamental problems
   (time step sizes, many-parameter runs (UQ), . . .)

Showstoppers and open problems—my summary

Durham University
Department of Computer Science

1. Right training (C and/or Fortran) has disappeared thanks to data science hype
   (both in CS, Maths and domain sciences)
2. Computer scientists solve toy problems, domain scientists stick to outdated algorithms
   (interdisciplinary often only sales pitch)
3. Hardware heterogeneity not solved but pushed due to machine learning success
   (pressure to migrate had historically not been high enough)
4. Some hard fundamental problems
   (time step sizes, many-parameter runs (UQ), . . .)

$\Rightarrow$ 1. Introduce correct (professional) training and make PIs send their PhDs and PostDocs to it

$\Rightarrow$ 4. ???

$\Rightarrow$ 2.–3. Establish better cost models for calculations

PR

Durham University
Department of Computer Science

- Master in Scientific Computing and Data Analysis (MISCADA)
  (CS, Maths and Physics + Earth Sciences + Business School (Finances) + CS (Robotics) + . . .)
  - More than “just” data science (apply stats or ML models, e.g.)
  - For people with “right” background ⇒ PhDs, not converts
  - Led by experts

  https://miscada.webspace.durham.ac.uk/
- Durham HPC Days—Spring 2023
  (IDAS, ARC, Rockport, Dell, CS, . . .)
  - Dan Stanzione (TACC): energy issue
  - David E. Keyes (KAUST): algorithm’s role and co-design
  - ExCALIBUR projects: heterogeneous hardware

  https://tobiasweinzierl.webspace.durham.ac.uk/research/workshops/durham-hpc-days-spring-2023/
- Performance Analysis Workshop series
  (CS, ARC and DiRAC)
  https://zenodo.org/record/5155503.ZFE9yY3MLmg