Prof. Simon McIntosh-Smith
Head of the HPC research group
University of Bristol, UK

Twitter: @simonmcs
Email: simonm@cs.bris.ac.uk

Beyond MPI+X

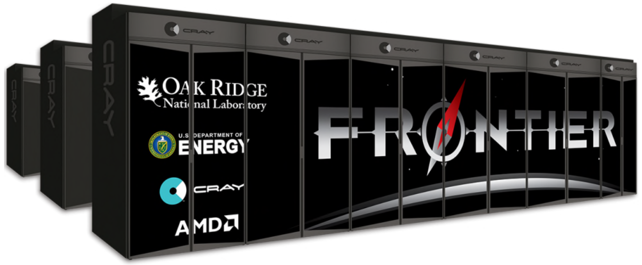
University of Bristol

http://uob-hpc.github.io

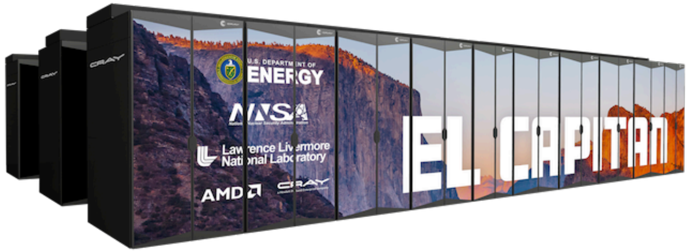
ASIMOV

Challenge: the diverse growth of parallelism

The coming generation of Exascale systems will include a diverse range of architectures at massive scale:



- **Fugaku**: Fujitsu A64FX Arm CPUs
- **Perlmutter**: AMD EYPC CPUs and NVIDIA GPUs
- **Frontier**: AMD EPYC CPUs and Radeon GPUs
- **Aurora**: Intel Xeon CPUs and Xe GPUs
- **El Capitan**: AMD EPYC CPUs and Radeon GPUs



University of BRISTOL

The Next Platform, Jan 13th 2020: "HPC in 2020: compute engine diversity gets real"
https://www.nextplatform.com/2020/01/13/hpc-in-2020-compute-engine-diversity-gets-real/

Three of the big issues facing parallel programming

1. **Massive parallelism**
   - Fugaku has over 7.63 **million** cores, each with 2x 512-bit wide vectors

2. **Heterogeneity**
   - CPUs and GPUs, both from multiple vendors
     - Intel, AMD, NVIDIA, Fujitsu, Marvell, IBM, Amazon, …
   - Non traditional architectures
     - Graphcore IPUs, Google TPUs, FPGAs, …

3. **Complex memory hierarchies**

So is there anything beyond "MPI+X" ?

Let's face it, **MPI+X** is going to be the most widely used programming model for scientific applications at Exascale

- It can be made to work
- We don't have to throw everything away
- MPI continues to evolve and can directly target GPUs
- The choices for "X" are becoming increasingly attractive:
  - **OpenMP** widely used and now also supports GPUs
  - Various dialects of **parallel C++** are maturing:
    - **SYCL**/DPC++, **Kokkos/RAJA**, …

But what else is there?

There are a few alternatives to MPI+X whose time might be right:

1. Partitioned Global Address Space (**PGAS**) languages
   - E.g. Chapel, Unified Parallel C (UPC), Coarray Fortran, …
2. **Julia**
   - In 2017 the Celeste project used Julia to achieve "peak performance of 1.54 PFLOP/s using 1.3 million threads" on 9,300 Knights Landing (KNL) nodes of the Cori II (Cray XC40) supercomputer (then 6th fastest system in the world)
3. And a few more exotic options:
   - **Rust**
   - **Go**
   - **…**

Who is using what?

Based on how often language tutorials are searched on Google:

Worldwide, Jun 2020 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	31.6 %	+4.3 %
2		Java	17.67 %	-2.4 %
3		Javascript	8.02 %	-0.2 %
4		C#	6.87 %	-0.4 %
5		PHP	6.02 %	-0.9 %
6		C/C++	5.69 %	-0.2 %
7		R	3.86 %	-0.1 %
8		Objective-C	2.5 %	-0.3 %
9		Swift	2.24 %	-0.1 %
10	↑	TypeScript	1.86 %	+0.2 %

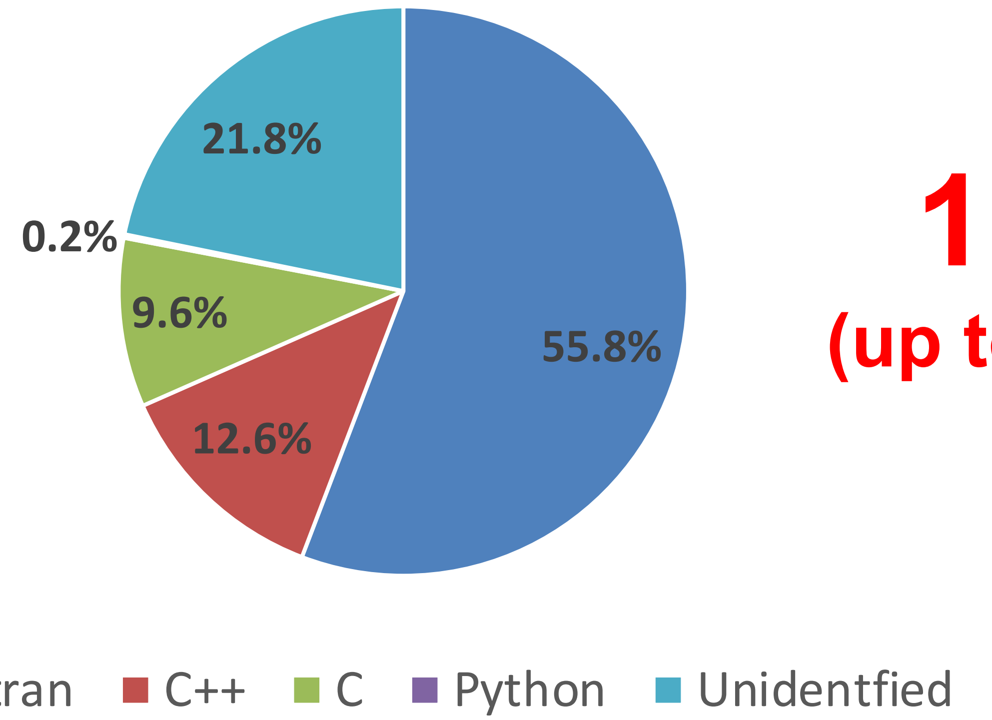
(Notice no Fortran!)

13	↑↑	Go	1.29 %	+0.2 %
18	↑↑	Rust	0.78 %	+0.2 %
25	↑↑↑	Julia	0.39 %	+0.1 %

Source: http://pypl.github.io/PYPL.html

But what about in HPC?

Data on language usage from the UK's national supercomputer service, ARCHER:



100% MPI

(up to 18% MPI+OMP)

Fraction of node hours over 1 year from May 2019 to April 2020

University of BRISTOL

http://uob-hpc.github.io

ASIMOV

Beyond MPI+X in the USA's ECP program

8 programming model and run-time projects funded in ECP:

- Two focus on MPI at Exascale (MPICH, OpenMPI)
- Two focus on task-level parallelism approaches (Legion, PaRSEC)
- One focuses on PGAS approaches (UPC++, GASNet)
- One focuses on parallel C++ (Kokkos, RAJA)
- Two focus on low-level on-node parallelism (ARGO, SICM)

Source: https://www.exascaleproject.org/research-group/programming-models-runtimes/

Challenges for beyond MPI+X approaches

- **Scalability**
  - Parallelism at the data, instruction, thread, core, socket, node, system, …
- **Portability**
  - Across CPUs, GPUs, different vendors, compilers, …
- **Performance portability**
  - Adapt to the best machine(s) available at the time

- **Many other issues too:**
  - Ease of use, availability, fault tolerance, longevity, …

Key takeaways for scientific software developers

- **Orders of magnitude more parallelism** at Exascale, ≥ $O(10^9)$
- **Increased heterogeneity** (CPU+X)
- **MPI+X likely to remain the most widespread solution**
- **If starting from scratch**, worth evaluating some of the alternatives
  - Julia, parallel task frameworks etc.

Exascale is not "business as usual"!

For more information

Bristol HPC group: https://uob-hpc.github.io/

Email: **S.McIntosh-Smith@bristol.ac.uk**

Twitter: **@simonmcs**

- **High Performance *in silico* Virtual Drug Screening on Many-Core Processors**
  S. McIntosh-Smith, J. Price, R.B. Sessions, A.A. Ibarra, IJHPCA 2014
- **On the performance portability of structured grid codes on many-core computer architectures**
  S.N. McIntosh-Smith, M. Boulton, D. Curran, & J.R. Price
  ISC, Leipzig, June 2014. DOI: 10.1007/978-3-319-07518-1_4
- **Assessing the Performance Portability of Modern Parallel Programming Models using TeaLeaf**
  Martineau, M., McIntosh-Smith, S. & Gaudin, W.
  Concurrency and Computation: Practice and Experience (Apr 2016)
- **GPU-STREAM v2.0: Benchmarking the achievable memory bandwidth of many-core processors across diverse parallel programming models**
  Deakin, T. J., Price, J., Martineau, M. J. & McIntosh-Smith, S. N.
  First International Workshop on Performance Portable Programming Models for Accelerators (P3MA), ISC 2016
- **The Productivity, Portability and Performance of OpenMP 4.5 for Scientific Applications Targeting Intel CPUs, IBM CPUs, and NVIDIA GPUs**
  M. Martineau and S. McIntosh-Smith, IWOMP 2017, Stony Brook, USA.

- **Evaluating Attainable Memory Bandwidth of Parallel Programming Models via BabelStream**
  Deakin, T, Price, J, Martineau, M, and McIntosh-Smith, S
  International Journal of Computational Science and Engineering (special issue), vol 17., 2018
- **Pragmatic Performance Portability with OpenMP 4.x**
  Martineau, Matt, Price, James, McIntosh-Smith, Simon, and Gaudin, Wayne
  Proceedings of the 12th International Workshop on OpenMP, 2016
- **Performance Analysis and Optimization of Clang's OpenMP 4.5 GPU Support**
  Martineau, Matt, McIntosh-Smith, Simon, Bertolli, Carlo, et al
  Proceedings of the International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS), 2016, SC'16
- **Exploiting auto-tuning to analyze and improve performance portability on many-core architectures**
  Price, J. & McIntosh-Smith, S., P^3MA, ISC High Performance 2017 International Workshops, Revised Selected Papers. Springer, Cham, p.538-556, vol. 10524 LNCS
- **Performance Portability across Diverse Computer Architectures**
  T. Deakin, S. McIntosh-Smith et al., 2019 IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC), Denver, CO, USA, 2019, pp. 1-13