Lecture 15: Introduction to High-Performance Computing (HPC)

Roadmap

Applied Concepts

- Introduction to meshing
- The finite volume method
- Jameson's scheme
- Solution storage approaches and their implications
- Advanced implicit methods
- **Today: Introduction to computer hardware and high performance computing**
  - **Technology trends: hardware & software**
  - **Bristol HPC resources**
  - **Parallelisation strategies**
- Next lecture: Parallel decomposition and efficiency

Background

Computational Fluid Dynamics has traditionally drawn on the latest developments in high performance computing as a result of the large problems encountered.

Jameson (1984) noted that "*A solution to the Navier-Stokes equations for a turbulent flow can at best be attempted for something like a rectangular cavity with one moving wall*". Moreover, discussing the prospect of solving the potential flow equations for a swept wing, he noted there was "*a failure in the aeronautical community to recognize that such calculations were possible*"

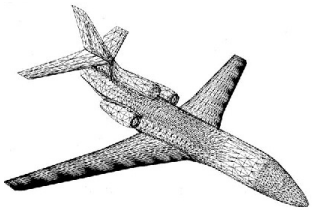
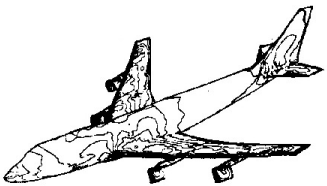


Image of surface mesh on Falcon aircraft

Source: Bristeau *et al.*, Proc. IMA conference on Numerical Methods in Applied Fluid Dynamics (1980) Work was reported in 1978, see also Comp. Meth. Appl. Mech. and Eng. V51, pp. 363-394, 1985



The world's first Euler calculation of a complete aircraft (Boeing 747-200) was done on a Cray XMP. #12038 nodes, #57914 tetrahedra[1] #24685 nodes, #132793 tetrahedra[2]

Source[1]: Jameson, Baker & Weatherill, AIAA-86-0103 (Jan 1986)

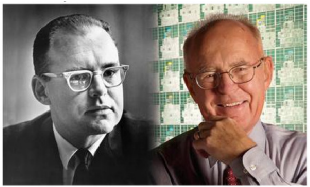
Source[2]: Jameson, ASME Symposium (June 1986)

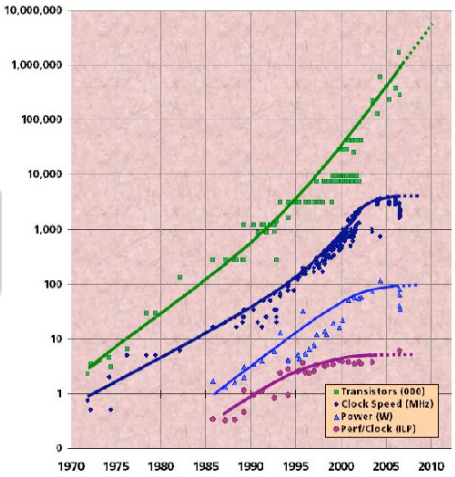
Technology Development

In 1965, Intel co-founder Gordon Moore published an article where he predicted that the number of components would double every 12 months due to improvements in design and manufacture. This article led to the establishment of what is now known as *Moore's law*.

Moore's Law

For a fixed price, the performance of a computer will double every 18 months.



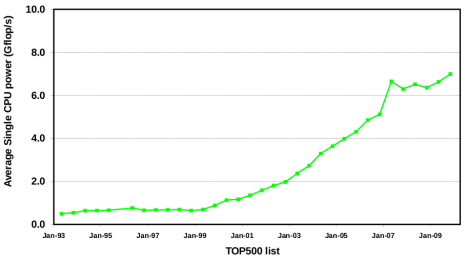


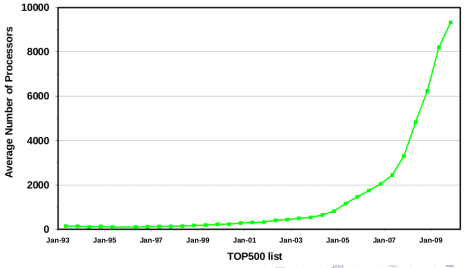
Technology Development

Up to late 1990's progression was due to increases in hardware - technology development resulted in higher clock speed and more operations clock cycle of CPUs.

- Single-core CPU development levelling off; primarily power issues and silicon manipulation limits. Machine power now rising rapidly due to increasing CPUs/cores.
- Nowadays the number of CPUs is more important than expensive developments of them. Lead to larger and larger machines of cheap 'commodity' hardware.

Latest supercomputer listings at: `http://www.top500.org`





HPC Trends

HPC machines continue to chase increased parallelism — *i.e.* increased core counts

Individual processors are becoming increasingly parallel

- AMD EPYC (2018): 64 cores (128 threads), 256 MB cache
- Intel Xeon (2019): 56 cores (112 threads), 77MB cache
- Fujitsu ARM A64FX (2020): 48 cores, 32 GB on-chip memory

HPC Trends

HPC machines continue to chase increased parallelism — *i.e.* increased core counts

Individual processors are becoming increasingly parallel

- AMD EPYC (2018): 64 cores (128 threads), 256 MB cache
- Intel Xeon (2019): 56 cores (112 threads), 77MB cache
- Fujitsu ARM A64FX (2020): 48 cores, 32 GB on-chip memory

Increased use of accelerator cards (GPUs)

- NVIDIA 'Ampere' (2020): 8704 cores
- GPUs offer significantly better performance (FLOPs) per Watt and per pound (£)
- But many limitations on fully exploiting their performance

Parallel Computing Strategies (1)

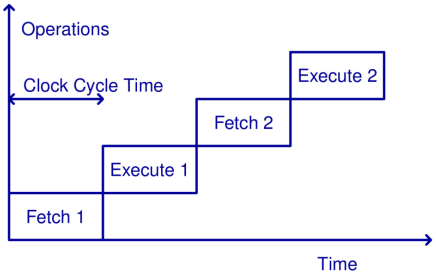
Instruction-level parallelism

- Pipelining — keep all resources in processor busy during clock cycles by overlapping elements of the fetch-decode-execute cycle
- Superscalar processing — execute multiple independent instructions in one clock cycle
- Implemented in hardware on all modern processors

Parallel Computing Strategies (1)

Instruction-level parallelism

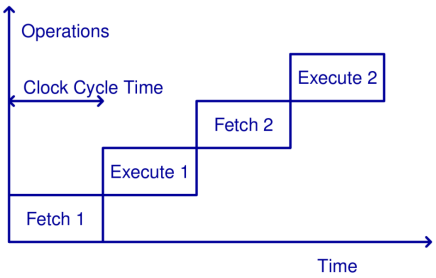
- Pipelining — keep all resources in processor busy during clock cycles by overlapping elements of the fetch-decode-execute cycle
- Superscalar processing — execute multiple independent instructions in one clock cycle
- Implemented in hardware on all modern processors

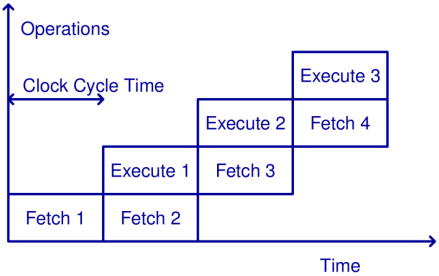


Parallel Computing Strategies (1)

Instruction-level parallelism

- Pipelining — keep all resources in processor busy during clock cycles by overlapping elements of the fetch-decode-execute cycle
- Superscalar processing — execute multiple independent instructions in one clock cycle
- Implemented in hardware on all modern processors





Parallel Computing Strategies (2)

Data-level parallelism (SIMD/Vectorisation)

- Use extra-wide registers and arithmetic units to perform the same instruction on multiple data elements simultaneously
- Implemented in all modern processors
- Vector instructions generated by most compilers — but limited in where it can be applied, may still need some code modification
- (Related: GPUs are essentially highly thread-parallel vector processors)

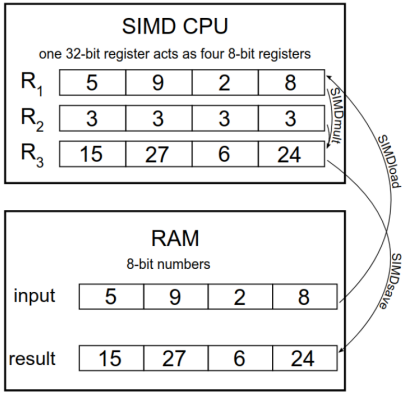


Diagram by 'Decora': https://commons.wikimedia.org/wiki/File:SIMD_cpu_diagram1.svg

Parallel Computing Strategies (3)

Shared-memory (thread-level/multi-core) parallelism

- Execute multiple threads on multiple cores within a single processor
- All threads have access to the same memory
- Limited scalability to number of cores on one machine
- Requires manual implementation from programmer (`pthreads`, `openmp`)

Parallel Computing Strategies (3)

Shared-memory (thread-level/multi-core) parallelism

- Execute multiple threads on multiple cores within a single processor
- All threads have access to the same memory
- Limited scalability to number of cores on one machine
- Requires manual implementation from programmer (`pthreads`, `openmp`)

Distributed-memory parallelism

- Execute multiple copies (instances) of program on multiple cores — instances do not have to be on the same machine
- Memory is segregated between instances: explicit communication required to transfer data or sychronise instances
- Scalable to very large number of instances
- Complex programmer implementation (`MPI`)

High Performance Computing (HPC)

HPC

"High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business." - https://insidehpc.com/hpc-basic-training/what-is-hpc/

High Performance Computing (HPC)

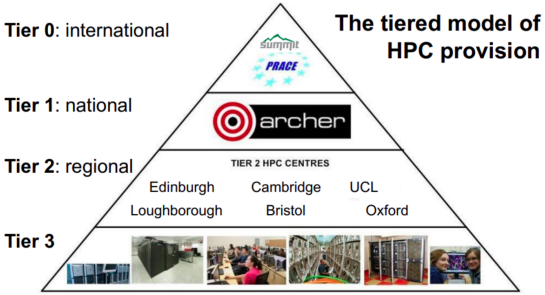
HPC

"High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business." - https://insidehpc.com/hpc-basic-training/what-is-hpc/

- HPC clusters are composed of many individual computers (or 'compute nodes')
- Compute nodes are connected by a high-speed internal network (or 'interconnect')
- Compute nodes cannot be accessed directly; access to the cluster is made by connecting to a login node
- Work is run on the cluster by submitting it to a 'queue' - this enforces various fair-use policies to share the computing resources among users.

HPC Facilities

The ‘tiered’ model describes different classes of HPC machine.



- Image: Dr James Price, University of Bristol / GW4 https://fortran.bcs.org/2018/AGM18FP_Price.pdf

- **TIER 1**: ARCHER2 was commissioned in 2021 with ¿ 700,000 cores (Rank: 22nd)
- Competitive application to secure computing time on ARCHER2

HPC Facilities at Bristol

TIER 2 (Regional)

- **Isambard 2** (2020)
  - GW 4 Alliance — Bath, Bristol, Cardiff & Exeter
  - £6.5M cost: 21,504 cores across 336 nodes
  - The largest ARM-based supercomputer in Europe
  - `https://gw4.ac.uk/isambard/`

HPC Facilities at Bristol

TIER 2 (Regional)

- **Isambard 2** (2020)
  - GW 4 Alliance — Bath, Bristol, Cardiff & Exeter
  - £6.5M cost: 21,504 cores across 336 nodes
  - The largest ARM-based supercomputer in Europe
  - `https://gw4.ac.uk/isambard/`

TIER 3 (Local) `https://www.bristol.ac.uk/acrc/`

- **Bluecrystal**
  - Phase 1 (2007 - 2015): £1M, 384 cores
  - Phase 2 (2009 - 2017): £2M, 3360 cores, ranked 67 in world, 4th in the UK

HPC Facilities at Bristol

TIER 2 (Regional)

- **Isambard 2** (2020)
  - GW 4 Alliance — Bath, Bristol, Cardiff & Exeter
  - £6.5M cost: 21,504 cores across 336 nodes
  - The largest ARM-based supercomputer in Europe
  - `https://gw4.ac.uk/isambard/`

TIER 3 (Local) `https://www.bristol.ac.uk/acrc/`

- **Bluecrystal**
  - Phase 1 (2007 - 2015): £1M, 384 cores
  - Phase 2 (2009 - 2017): £2M, 3360 cores, ranked 67 in world, 4th in the UK
  - Phase 3 (2013): £2M, 5600 cores + 76 GPUs, ranked 431 in world
  - Phase 4 (2017): £3M, 12,000 cores + 64 GPUs, ranked 302 in world

HPC Facilities at Bristol

TIER 2 (Regional)

- **Isambard 2** (2020)
  - GW 4 Alliance — Bath, Bristol, Cardiff & Exeter
  - £6.5M cost: 21,504 cores across 336 nodes
  - The largest ARM-based supercomputer in Europe
  - `https://gw4.ac.uk/isambard/`

TIER 3 (Local) `https://www.bristol.ac.uk/acrc/`

- **Bluecrystal**
  - Phase 1 (2007 - 2015): £1M, 384 cores
  - Phase 2 (2009 - 2017): £2M, 3360 cores, ranked 67 in world, 4th in the UK
  - Phase 3 (2013): £2M, 5600 cores + 76 GPUs, ranked 431 in world
  - Phase 4 (2017): £3M, 12,000 cores + 64 GPUs, ranked 302 in world
- **Blue Pebble** (2019) — throughput cluster (many small jobs)
- **Catalyst** — An ARM-based system

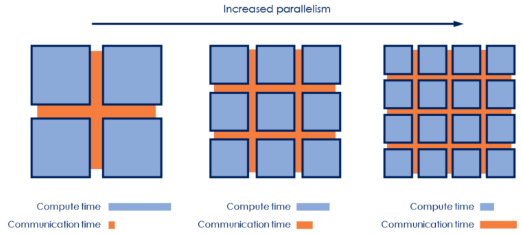
Bluecrystal Phase 3

In the ‘tank room’ on top of the Physics Building

Parallel Computing Challenges for CFD

- CFD codes based on domain decomposition are the standard
- Scaling of existing CFD is poor — efficient use of more than $O(1000)$ cores is only possible for very large niche problems
  - Traditional domain decomposition relies n high-speed inter-node communication
  - Communication costs quickly become dominant as parallelism increases — leads to poor utilisation of computing resources
  - Fine-grain (thread) parallelism limited by available memory bandwidth

"CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences" Tech. rep., NASA, 2014. NASA/CR-2014-218178



Summary

- CFD is computationally expensive, so demands use of high performance computing (HPC) resources.
- Modern HPC involves massively parallel clusters made up of many thousands of cores
- Single-core development has stalled due to physical limitations
  - Performance improvements primarily from increased core counts per processor
- HPC is now a heavy user of highly-parallel GPU accelerator cards
  - Very fast for some problems, but difficult to fully exploit for general problems
- Scalable parallelisation is commonly implemented using distributed-memory parallelism with **domain decomposition**

Next Lecture: Consider domain decomposition applied to CFD meshes and the consequent efficiency of parallelisation