GPUs and Emerging Architectures

Mike Giles

mike.giles@maths.ox.ac.uk

Mathematical Institute, Oxford University

e-Infrastructure South Consortium

Oxford e-Research Centre

CPUs and GPUs

Good news — Moore's Law will continue for next decade, with compute capability per chip doubling every two years.

Features can't shrink much more, so this will increasingly be achieved by going 3D with more layers.

However, individual cores will not go faster, so the increased throughput is from massive parallelism.

Trend is towards heterogeneous mix of fast scalar cores and powerful vector units.

CPUs and GPUs

Intel: current CPUs have 8 cores each with AVX vector unit (4 doubles or 8 floats)

MIC chips will have 50-64 cores with longer AVX units

next-generation CPUs will have integrated GPGPU unit

VIDIA: general purpose GPUs with up to 1536 cores, grouped in vectors of size 32 – in a year or two they plan to add ARM CPUs to avoid the need for a separate host CPU

AMD: CPU plans like Intel – already have integrated GPUs

discrete GPU plans similar to NVIDIA

ARM: very energy efficient CPUs / GPUs – could become a major HPC player in next 10 years

Memory Hierarchy

Memory speed / bandwidth has developed slower than CPUs/GPUs, so we have more and more memory levels:

registers

L1 / L2 cache (64-256 kB / core)

400 GB/s

L3 / LLC – last level cache on chip (10-20 MB)

150 GB/s

GDDR5 – fast, expensive graphics memory (3-6 GB)

75 GB/s

DDR3 – cheaper, slower conventional RAM (16-64GB)

10 GB/s

SSD – flash memory solid state disk (100-500GB)

5 GB/s

HDD – conventional hard disk (1-5TB)

HPC Networking

- Ethernet

  10 GigE $\Longrightarrow$ 40 GigE $\Longrightarrow$ 100 GigE

- Infiniband

  40 Gb/s QDR $\Longrightarrow$ 56 Gb/s FDR $\Longrightarrow$ 100 Gb/s EDR

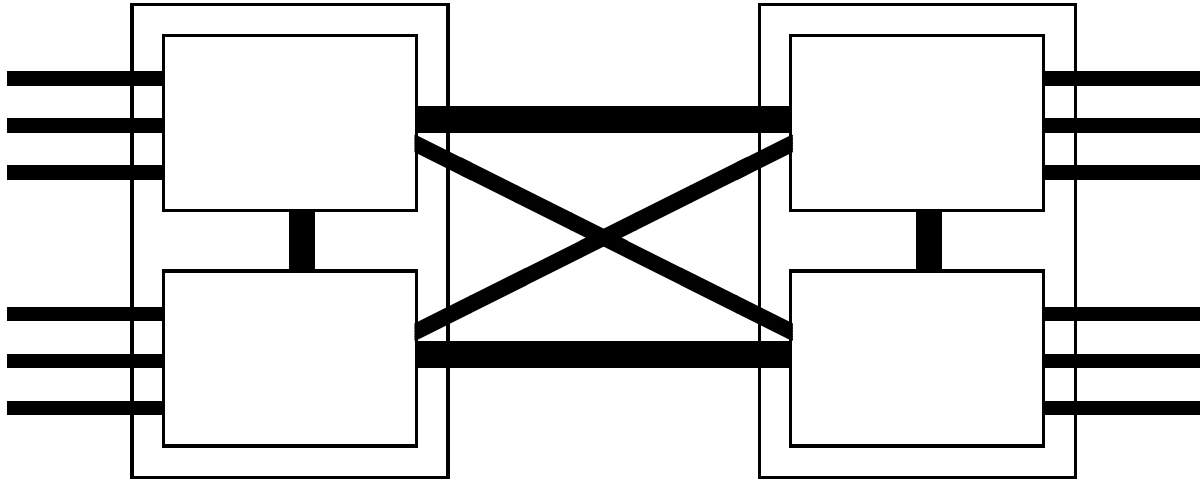
- trend towards same NICs / switches for both protocols

- move also towards integrating networking into CPUs (alongside PCIe support)

- I/O bandwidth into system often the limiting factor for real-time data processing – compare 100-500 GFlops to 1-5 GB/s bandwidth

Hardware summary

- generally, everything getting faster
- data movement an increasing bottleneck, leading to research on “communication avoiding algorithms”
- massive parallelism is a challenge to algorithms / programming
- NUMA (non-uniform memory access) features are increasingly painful



MPI

- still established standard for distributed memory systems
- PGAS (partitioned global address space) virtual shared memory approaches have failed to take off
- MPI also useful for NUMA systems, with separate MPI process for each die / memory zone
- MPI features (e.g. global reduction) being built into switches to address latency issues for enormous systems

OpenMP

- still established standard for shared-memory multithreading, but doesn’t give good performance with threads spread across NUMA system
- OpenACC extension aimed at “accelerators” like GPUs and MIC cards – still too early to assess but I’m not optimistic

GPU programming

CUDA

- proprietary extension of C/C++ for NVIDIA GPUs
- PGI has developed a FORTRAN CUDA compiler, and also one to convert to AVX executables for CPUs
- lots of HPC library support

OpenCL

- multi-platform (NVIDIA, AMD, Intel, ARM, Imagination)
- well-established standard at low end (e.g. smartphones)
- not clear whether it will eventually win out at high end

AVX vectorisation

Currently a mess – too many different approaches

- Intel low-level vector primitives
- Intel `icc` compiler
- Intel unsupported `spmd` compiler
- Intel TBB / ABB
- Intel Cilk
- OpenCL

Languages

Not much better:

- C99, C++
- Fortran 90 / 95 / 03 / 08
- C#, F#
- Java
- Python
- MATLAB

None well suited to massively parallel computing, but very tough for a new language to get established

Computer science challenges

Many-core research identified by EPSRC as a major priority

- energy-efficient computing
- high-level frameworks and domain specific languages
- make it simpler for application developers, and avoid them being tied to a particular hardware platform
- software has to last many times longer than hardware

Final comments

Key take-home points:

- hardware is continuing to get faster through massive parallelism
- energy consumption is an increasing concern
- software side is increasingly tough, needs substantial investment in people and software development