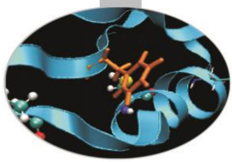


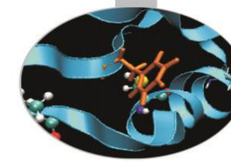
HPC Architectures – past ,present and emerging trends

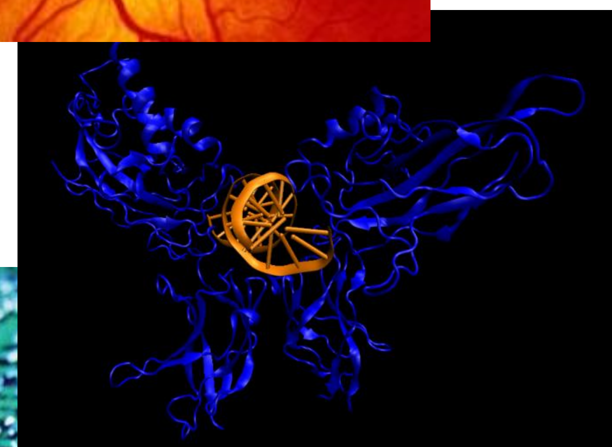
Author: Andrew Emerson, Cineca
a.emerson@cineca.it
Speaker: Alessandro Marani, Cineca
a.marani@cineca.it

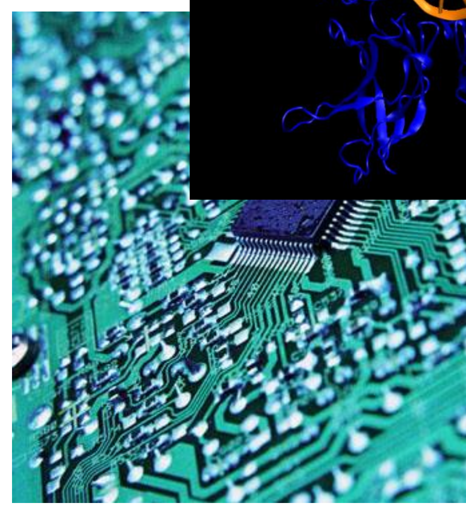
Agenda

- Computational Science
- Trends in HPC technology
- Trends in HPC programming
    - Massive parallelism
    - Accelerators
    - The scaling problem
- Future trends
    - Memory and accelerator advances
    - Monitoring energy efficiency
- Wrap-up

Computational Science



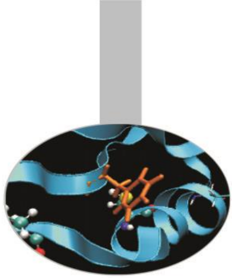


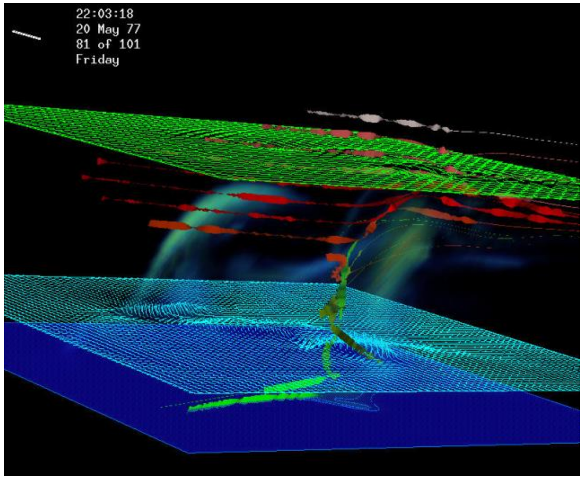


"Computational science is concerned with constructing mathematical models and quantitative analysis techniques and using computers to analyze and solve scientific problems. In practical use, it is typically the application of computer simulation and other forms of computation from numerical analysis and theoretical computer science to problems in various scientific disciplines." (Wikipedia)

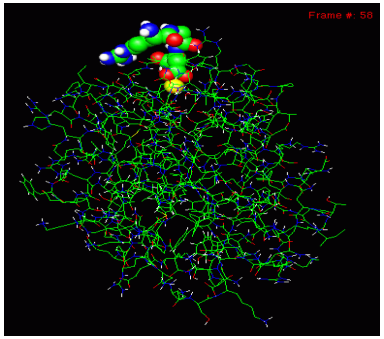
Computational science (with theory and experimentation), is the "third pillar" of scientific inquiry, enabling researchers to build and test models of complex phenomena.

Computational Sciences





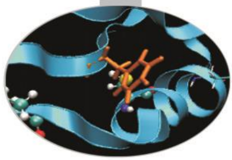
Computational methods allow us to study complex phenomena, giving a powerful impetus to scientific research.



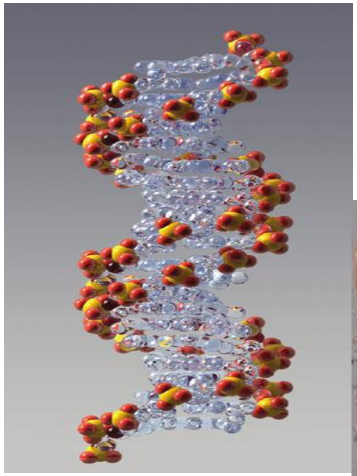
The use of computers to study physical systems allows to manage phenomena

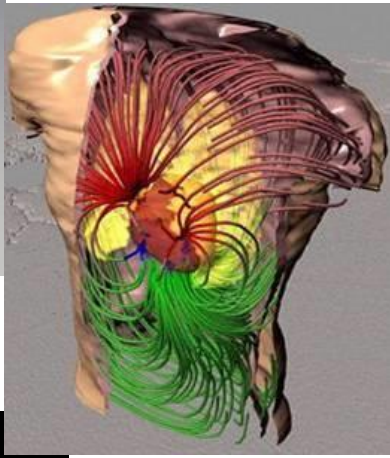
- **very large**
  (meteo-climatology, cosmology, data mining, oil reservoir)
- **very small**
  (drug design, silicon chip design, structural biology)
- **very complex**
  (fundamental physics, *fluid dynamics, turbolence)*
- **too dangerous or expensive**
  *(fault simulation, **nuclear** tests, crash analysis)*

Technology Evolution



More data everywhere:
Radar, satellites, CAT scans, weather models, the human genome, mobile devices.
The size and resolution of the problems scientists address today are limited only by the size of the data they can reasonably work with.
There is a constantly increasing demand for faster processing on bigger data.





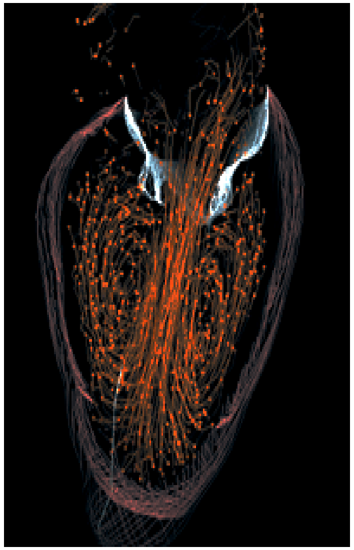
Increasing problem complexity:
Partly driven by the ability to handle bigger data, but also by the requirements and opportunities brought by new technologies. For example, new kinds of medical scans create new computational challenges.

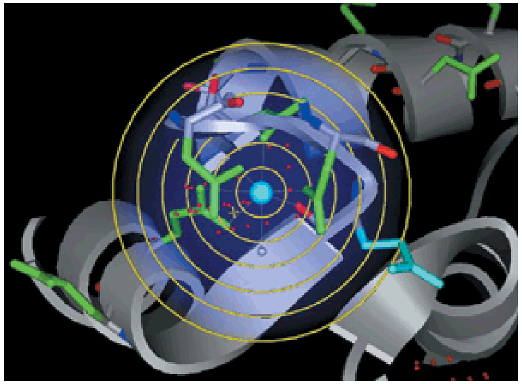
HPC Evolution
As technology allows scientists to handle bigger datasets and faster computations, they push to solve harder problems.
In turn, the new class of problems drives the next cycle of technology innovation.

Computational Science Today

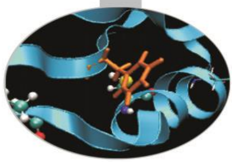
Multidisciplinary and multiscale problems using coupled applications include:

- Full simulation of engineering systems
- Full simulation of biological systems
- Astrophysics
- Materials science
- Bio-informatics, proteomics, pharmaco-genetics
- Scientifically accurate 3D functional models of the human body
- Biodiversity and biocomplexity
- Climate and Atmospheric Research
- Energy
- Digital libraries for science and engineering

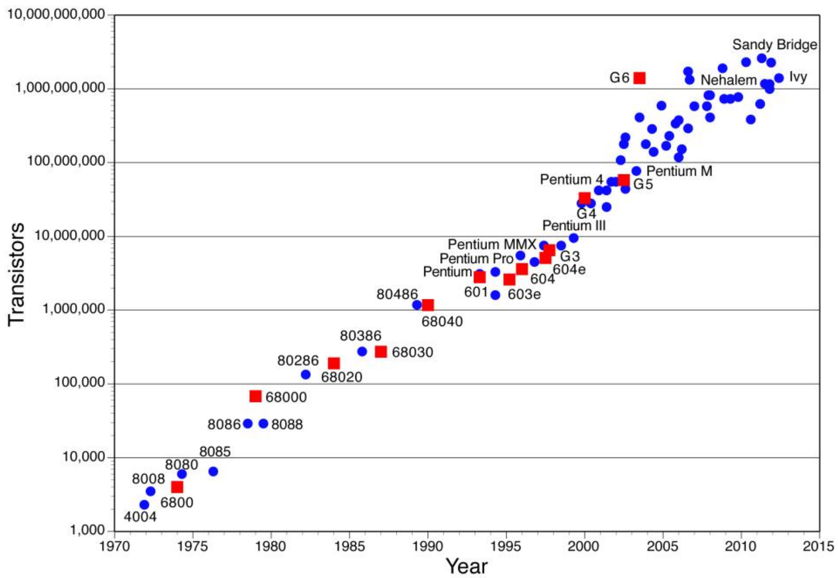




Which factors limit computer power?

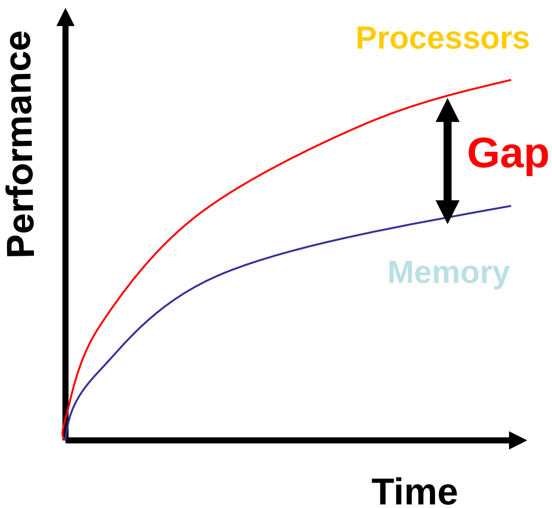


we can try and increase the speed of microprocessors but ..

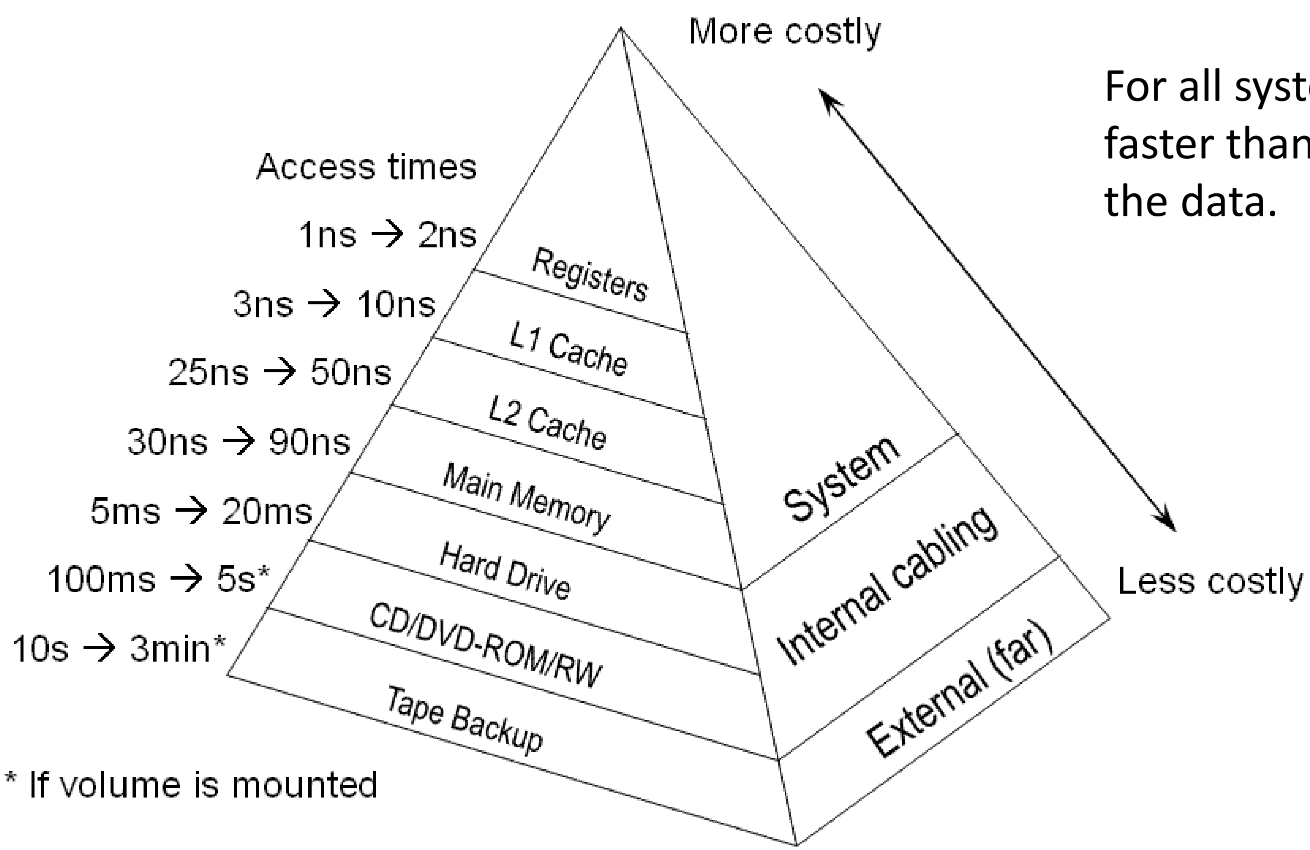


Moore’s law gives only a slow increase in CPU speed.
(It is estimated that Moore's Law will still hold in the near future but applied to the number of cores per processor) and ..

.. the bottleneck between CPU and memory and other devices is growing



Memory Hierarchy

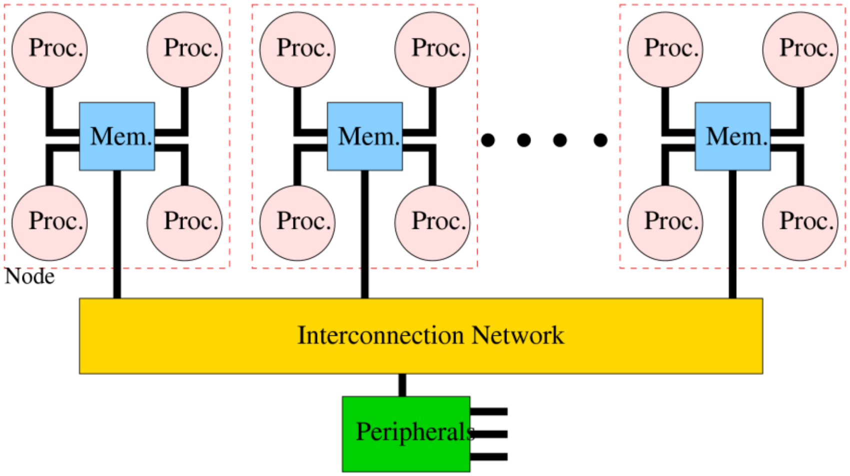


For all systems, CPUs are much faster than the devices providing the data.

HPC Architectures

The main factor driving performance is *parallelism*. *This can be on many levels:*

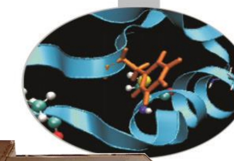
- *Instruction level parallelism*
- *Vector processing*
- *Cores per processor*
- *Processors per node*
- *Processors + accelerators (for hybrid)*
- *Nodes in a system*



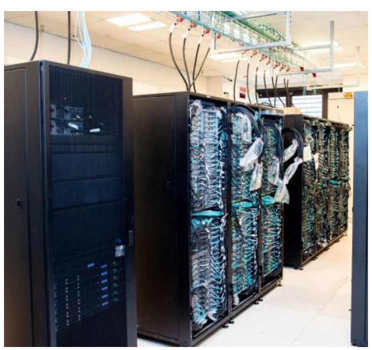
Performance can also derive from device technology

- *Logic switching speed and device density*
- *Memory capacity and access time*
- *Communications bandwidth and latency*

HPC systems evolution in CINECA



1969: CDC 6600 1st system for scientific computing
1975: CDC 7600 1st supercomputer
1985: Cray X-MP / 4 8 1st vector supercomputer
1989: Cray Y-MP / 4 64
1993: Cray C-90 / 2 128
1994: Cray T3D 64 1st parallel supercomputer
1995: Cray T3D 128
1998: Cray T3E 256 1st MPP supercomputer
2002: IBM SP4 512 1 Teraflops
2005: IBM SP5 512
2006: IBM BCX 10 Teraflops
2009: IBM SP6 100 Teraflops
2012: IBM BG/Q 2 Petaflops
2016: Lenovo NextScale 20 Petaflops



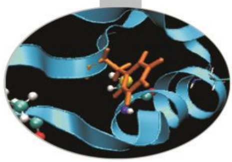
SuperComputing Applications and Innovation

HPC architectures/1

The are several factors that have an impact on the system architectures including:

1. Power consumption has become a primary headache.
2. Processor speed is never enough.
3. Network complexity/latency is a main hindrance.
4. Access to memory.

HPC architectures/2



Two approaches to increasing supercomputer power, but at the same time limiting power consumption:

1. Massive parallelism (IBM Bluegene range).
2. Hybrids using accelerators (GPUs and Xeon PHIs).

IBM BG/Q

- BlueGene systems link together tens of thousands of low power cores with a fast network.
- In some respects the IBM BlueGene range represents one extreme of parallel computing

Name: Fermi (Cineca)
Architecture: IBM BlueGene/Q
Model: 10 racks
Processor Type: IBM PowerA2, 1.6 GHz
Computing Cores: 163840
Computing Nodes: 10240, 16 core each
RAM: 16 GB/node, 1GB/core
Internal Network: custom with 11 links -> 5D Torus
Disk Space: 2.6 PB of scratch space
Peak Performance: 2PFlop/s

Hybrid systems

- Another approach is to “accelerate” normal processors by adding more specialised devices to perform some of the calculations.
- The approach is not new (maths co-procs, FPGAs, video-cards etc) but became important in HPC when Nvidia launched CUDA and GPGPUs.
- Capable of more Flops/Watt compared to traditional CPUs but still relies on parallelism (many threads in the chip).
- In the last few years Intel has introduced the Xeon PHI accelerator based on MIC (Many Integrated Core) technology.
- Aimed as an alternative to NVIDIA GPUs in HPC.

Model: IBM NextScale (GALILEO)
Architecture: Linux Infiniband Cluster
Nodes: 516
Processors: 8-cores Intel Haswell 2.40 GHz (2 per node)
Cores: 16 cores/node, 8256 cores in total
GPU: 2 NVIDIA Tesla K80 per node (80 in total)
MIC: 2 Intel Phi 7120p per node on 344 nodes (688 in total)
RAM: 128 GB/node, 8GB/core
Internal Network: Infiniband with 4x QDR switches
Disk Space: 2,500 TB of local scratch
Peak Performance: 1 PFlop/s

Hybrid Systems/2

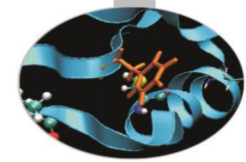
MARCONI A1

Model: Lenovo NeXtScale
Architecture: Intel Omnipath Cluster
Processors Type: 18-cores Intel Xeon E5-2697 v4 (Broadwell) 2.30 Ghz (2 per node)
Number of nodes: 1512 Compute
Number of cores: 54432
RAM: 128 GB/node, 3.5 GB/core
Internal Network: Intel Omnipath Architecture 2:1
Peak Performance: 2 Pflop/s

MARCONI A2

Model: Lenovo Adam Pass
Architecture: Intel Omnipath Cluster
Processors Type: 68-cores Intel Xeon Phi 7250 CPU (Knights Landing) 1.40 Ghz
Number of nodes: 3600 Compute
Number of cores: 244800
RAM: 108 GB/node, 96 of DDR4 and 16 of MCDRAM
Internal Network: Intel Omnipath Architecture 2:1
Peak Performance: 11 Pflop/s

Top500 – November 2014



Rank	Site	System	Cores	(TFlop/s)	(TFlop/s)	(kW)
1	National Super Computer Center in Guangzhou China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
2	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209
3	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1,572,864	17,173.2	20,132.7	7,890
4	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705,024	10,510.0	11,280.4	12,660
5	DOE/SC/Argonne National Laboratory United States	**Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM	786,432	8,586.6	10,066.3	3,945
6	Swiss National Supercomputing Centre (CSCS) Switzerland	**Piz Daint** - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect , NVIDIA K20x Cray Inc.	115,984	6,271.0	7,788.9	2,325
7	Texas Advanced Computing Center/Univ. of Texas United States	**Stampede** - PowerEdge C8220, Xeon E5-2680 8C 2.700GHz, Infiniband FDR, Intel Xeon Phi SE10P Dell	462,462	5,168.1	8,520.1	4,510
8	Forschungszentrum Juelich (FZJ) Germany	**JUQUEEN** - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect IBM	458,752	5,008.9	5,872.0	2,301
9	DOE/NNSA/LLNL United States	**Vulcan** - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect IBM	393,216	4,293.3	5,033.2	1,972
10	Government United States	Cray XC30, Intel Xeon E5-2697v2 12C 2.7GHz, Aries interconnect Cray Inc.	225,984	3,143.5	4,881.3	

BG/Q
GPU
Xeon PHI

Top500 – November 2016

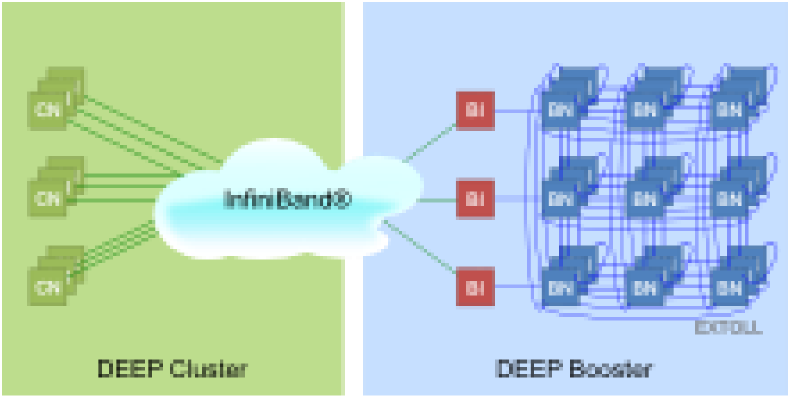
Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	National Supercomputing Center in Wuxi China	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway NRCPC	10,649,600	93,014.6	125,435.9	15,371
2	National Super Computer Center in Guangzhou China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
3	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209
4	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1,572,864	17,173.2	20,132.7	7,890
5	DOE/SC/LBNL/NERSC United States	**Cori** - Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect Cray Inc.	622,336	14,014.7	27,880.7	3,939
6	Joint Center for Advanced High Performance Computing Japan	**Oakforest-PACS** - PRIMERGY CX1640 M1, Intel Xeon Phi 7250 68C 1.4GHz, Intel Omni-Path Fujitsu	556,104	13,554.6	24,913.5	2,719
7	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705,024	10,510.0	11,280.4	12,660
8	Swiss National Supercomputing Centre (CSCS) Switzerland	**Piz Daint** - Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100 Cray Inc.	206,720	9,779.0	15,988.0	1,312
9	DOE/SC/Argonne National Laboratory United States	**Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM	786,432	8,586.6	10,066.3	3,945
10	DOE/NNSA/LANL/SNL United States	**Trinity** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect Cray Inc.	301,056	8,100.9	11,078.9	4,233

DEEP (Dynamical Exascale Entry Platform)

DEEP is an Exascale project funded by the EU 7th framework programme. The main goal is to develop a novel, Exascale-enabling supercomputing platform.

Prototype based on multi-core cluster linked to a “booster” part based on Intel’s MIC technology.

Cluster-booster comm handled by Parastation MPI OmpSs to ease application deployment

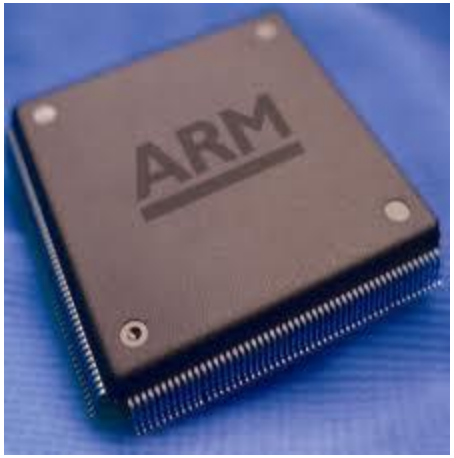


The Challenge of Exascale

The aim of the Mont Blanc project is to confront the problem of energy efficiency in Exascale systems by designing HPC systems based on low power components used in embedded systems and mobile devices such as ARM processors.

One objective is to design system using 30x less power than current systems.

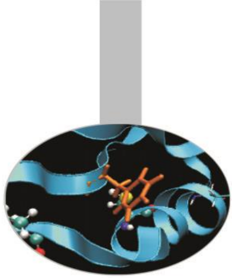
http://www.montblanc-project.eu/



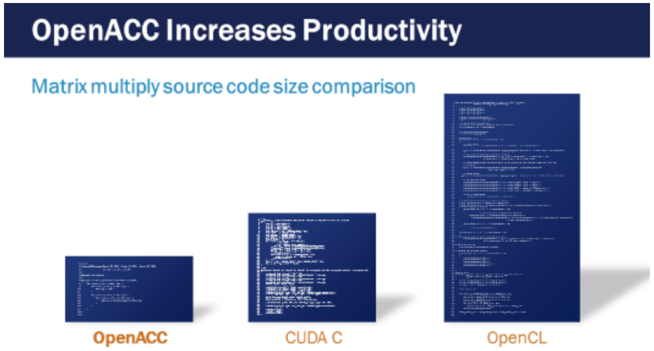
Roadmap to Exascale (architectural trends)

Systems	2009	2011	2015	2018
System Peak Flops/s	2 Peta	20 Peta	100-200 Peta	1 Exa
System Memory	0.3 PB	1 PB	5 PB	10 PB
Node Performance	125 GF	200 GF	400 GF	1-10 TF
Node Memory BW	25 GB/s	40 GB/s	100 GB/s	200-400 GB/s
Node Concurrency	12	32	O(100)	O(1000)
Interconnect BW	1.5 GB/s	10 GB/s	25 GB/s	50 GB/s
System Size (Nodes)	18,700	100,000	500,000	O(Million)
Total Concurrency	225,000	3 Million	50 Million	O(Billion)
Storage	15 PB	30 PB	150 PB	300 PB
I/O	0.2 TB/s	2 TB/s	10 TB/s	20 TB/s
MTTI	Days	Days	Days	O(1Day)
Power	6 MW	~10 MW	~10 MW	~20 MW

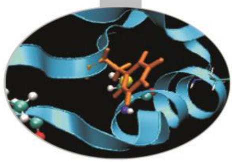
Parallel Software Models



- How do we program for supercomputers?
- C/C++ or FORTRAN, together with one or more of
  - Message Passing Interface (MPI)
  - OpenMP, pthreads, hybrid MPI/OpenMP
  - CUDA, OpenCL, OpenACC, compiler directives
- Higher Level languages and libraries
  - Co-array FORTRAN, Unified Parallel C (UPC), Global Arrays
  - Domain specific languages and data models
  - Python or other scripting languages



Message Passing: MPI



Main Characteristics

- Implemented as libraries
- Coarse grain
- Inter-node parallelization (few real alternatives)
- Domain partition
- Distributed Memory
- Long history and almost all HPC parallel applications use it.

Open Issues

- Latency
- OS jitter
- Scalability
- High memory overheads (due to program replication and buffers)

Debatable whether MPI can handle millions of tasks, particularly in collective calls.

```
call MPI_Init(ierror)
call MPI_Comm_size(MPI_Comm_World, size, ierror)
call MPI_Comm_rank(MPI_Comm_World, rank,ierror)
call MPI_Finalize(ierror)
```

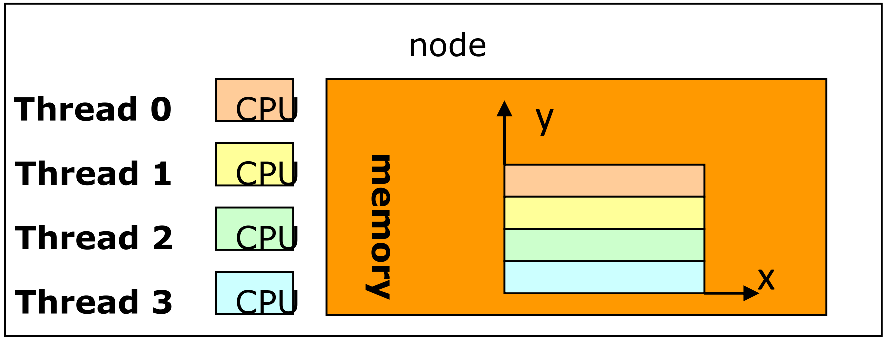
Shared Memory: OpenMP

Main Characteristics

- Compiler directives
- Medium grain
- Intra-node parallelization (p-threads)
- Loop or iteration partition
- Shared memory
- For Many HPC Applications easier to program than MPI (allows incremental parallelization)

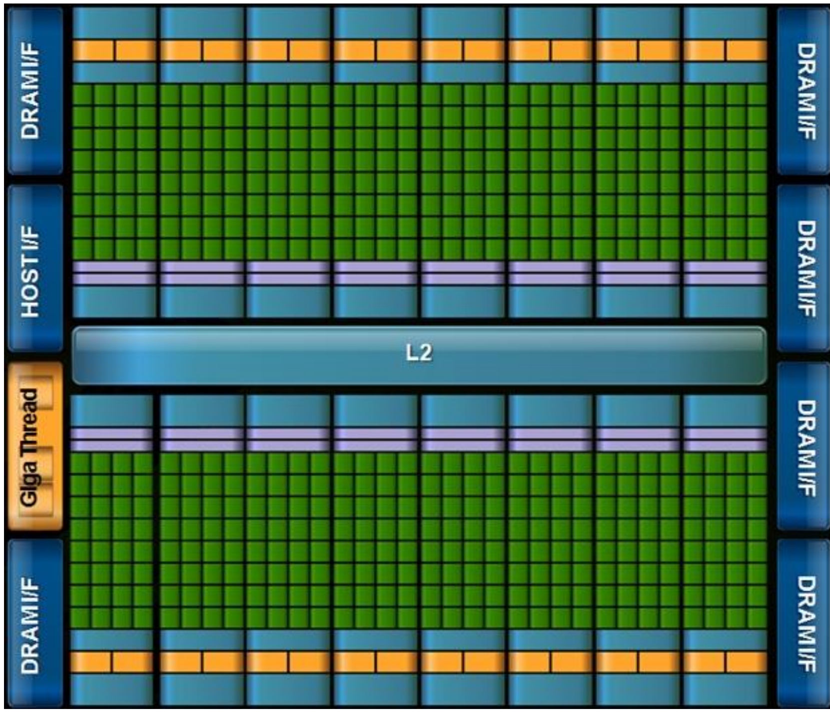
Open Issues

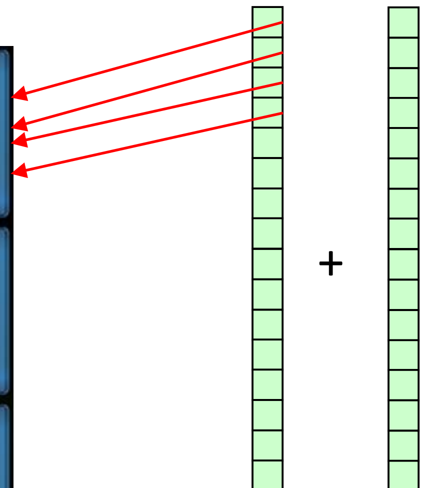
- Thread creation overhead (often worse performance than equivalent MPI program)
- Memory/core affinity
- Interface with MPI



Threads communicate via variables in shared memory

Accelerator/GPGPU





Exploit massive stream processing capabilities of GPGPUs which may have thousands of cores

Sum of 1D array

```
global__void  GPUCode( int* input1,
int*input2, int* output, int length)
{
  int idx = blockDim.x * blockIdx.x +
threadIdx.x;
  if ( idx < length ) {
     output[ idx ] = input1[ idx ] +
input2[ idx ];
               }
}
```

NVIDIA/CUDA

Main Characteristics

- Ad-hoc compiler
- Fine grain
- offload parallelization (GPU)
- Single iteration parallelization
- Ad-hoc memory
- Few HPC Applications

Open Issues

- Memory copy (via slow PCIe link)
- Standards
- Tools, debugging
- Integration with other languages

Accelerator/Xeon PHI (MIC)

The Xeon PHI co-processor based on Intel's Many Integrated Core (MIC) Architecture combines many cores (>50) in a single chip.

Main Characteristics

- Standard Intel compilers and MKL library functions.
- Uses C/C++ or FORTRAN code.
- Wide (512 bit) vectors
- Offload parallelization like GPU but also "native" or symmetric modes.
- Currently very few HPC Applications

```
ifort –mmic –o exe_mic prog.f90
```

Open Issues

For Knight's Corner:

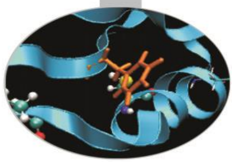
- Memory copy via slow PCIe link (just like GPUs).
- Internal (ring) topology slow.
- Wide vector units need to be exploited, so code modifications probable.
- Best also with many threads

Putting it all together -Hybrid parallel programming (example)

- Python: Ensemble simulations
  - MPI: Domain partition
    - OpenMP: External loop partition
      - CUDA: assign inner loops Iteration to GPU threads

Quantum ESPRESSO http://www.qe-forge.org/

Software Crisis



Real HPC Crisis is with Software

A supercomputer application and software are usually much more long-lived than a hardware

- Hardware life typically four-five years at most.
- Fortran and C are still the main programming models

Programming is stuck

- Arguably hasn't changed so much since the 70's

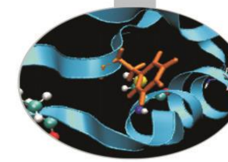
Software is a major cost component of modern technologies.

- The tradition in HPC system procurement is to assume that the software is free.

It's time for a change

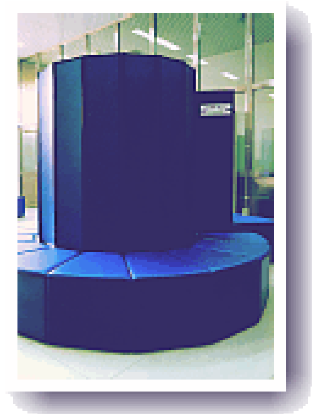
- Complexity is rising dramatically
- Challenges for the applications on Petaflop systems
- Improvement of existing codes will become complex and partly impossible.
- The use of O(100K) cores implies dramatic optimization effort.
- New paradigm as the support of a hundred threads in one node implies new parallelization strategies
- Implementation of new parallel programming methods in existing large applications can be painful

Hardware and Software advances comparison



PERFORMANCE

1975



400 Mflops

2015

173 Gflops (GPU)

1965

8Mb

2015

128Gb

STORAGE

1970

```
      PROGRAM HELLO
C
       REAL A(10,10)
       DO 50 I=1,10
          PRINT *,'Hello'
50      CONTINUE

       CALL DGEMM(N,10,I,J,A)
```

2015

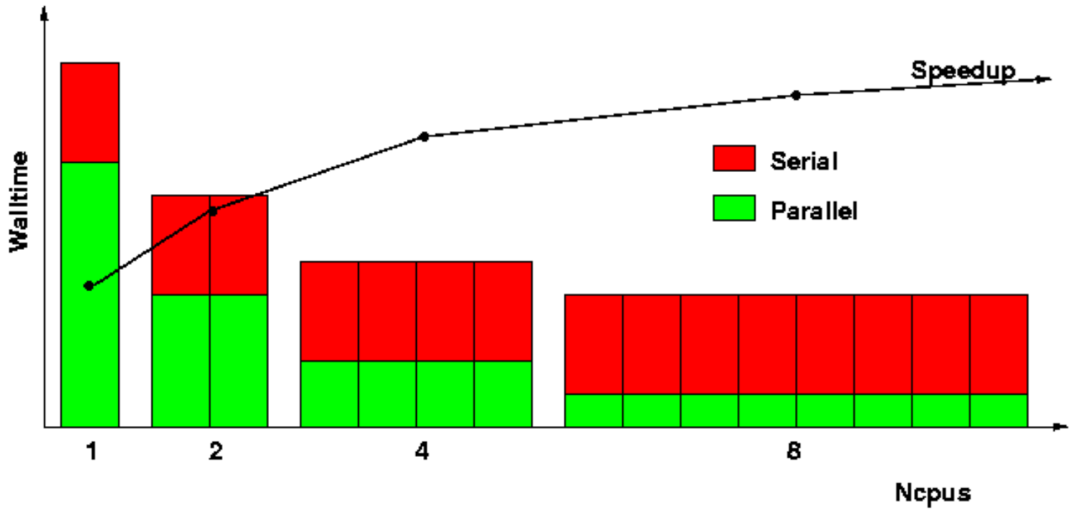
```
      PROGRAM HELLO
C
       REAL A(10,10)
       DO 50 I=1,10
          PRINT *,'Hello'
50      CONTINUE

       CALL DGEMM(N,10,I,J,A)
```

SOFTWARE

The problem with parallelism…

In a massively parallel context, an upper limit for the scalability of parallel applications is determined by the fraction of the overall execution time spent in non-scalable operations (Amdahl's law).



For N=no. of procs and
P=parallel fraction
max. speedup S(N) is given by

$$S(N) = \frac{1}{(1-P) + \frac{P}{N}}$$

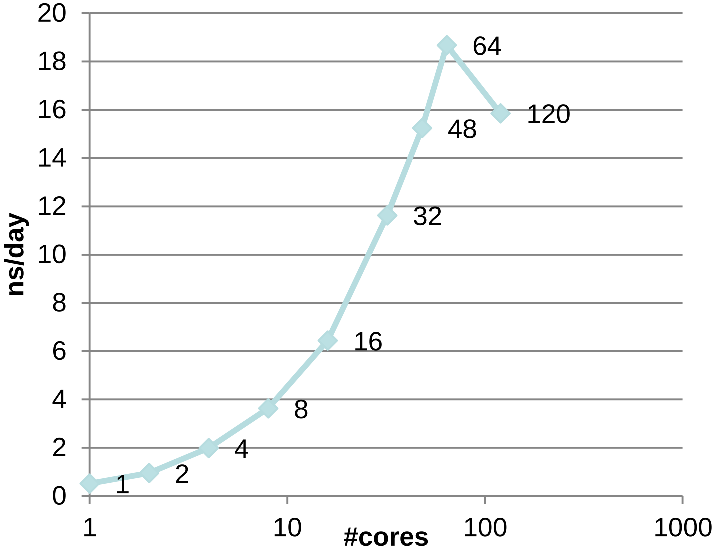
$$N \to \infty,$$

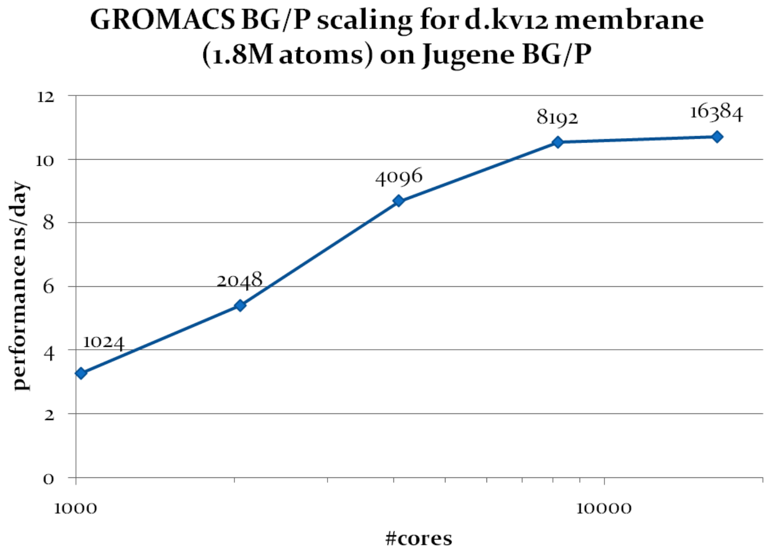
$$S(N) = \frac{1}{1-P}$$

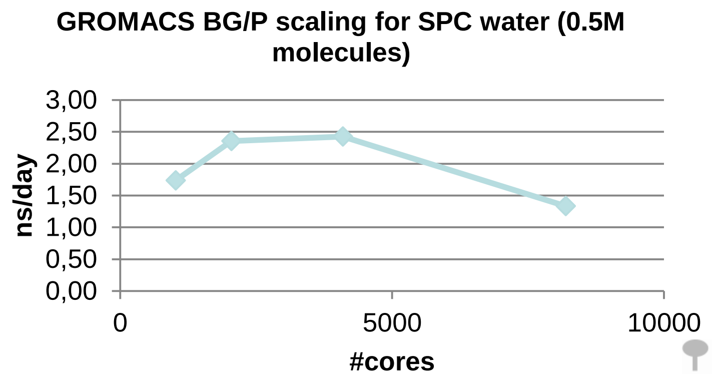
i.e. the max speedup is not dependent on N. Must minimize P if we want to work with many processors.

The scaling limit

- Most application codes do not scale up-to thousands of cores.
- Sometimes the algorithm can be improved but frequently there is a hard limit dictated by the size of the input.
- For example, in codes where parallelism is based on domain decomposition (e.g. molecular dynamics) no. of atoms may be < no. of cores available.







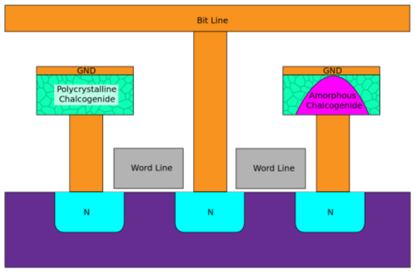
SCAI
SuperComputing Applications and Innovation

Other software difficulties

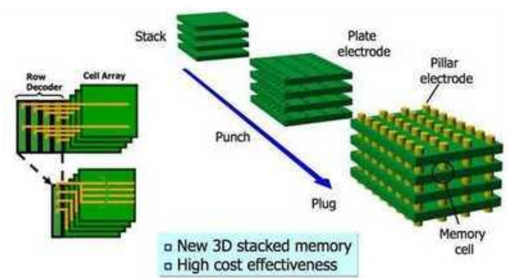
- Legacy applications (includes most scientific applications) not designed with good software engineering principles. Difficult to parallelise programs with many global variables, for example.
- Memory/core decreasing.
- I/O heavy impact on performance, esp. for BlueGene where I/O is handled by dedicated nodes.
- Checkpointing and resilience.
- Fault tolerance over potentially many thousands of threads.
  - In MPI, if one task fails all tasks are brought down.

Memory and accelerator advances things to look out for

- Memory
  - In HPC memory is generally either fast, small cache (SRAM) close to the CPU or larger, slower, main memory (DRAM).But memory technologies and ways of accessing it are evolving.
    - Non-volatile RAM (NVRAM). Retains information when power switched off. Includes flash and PCM (Phase Change Memory).
    - 3D Memory. DRAM chips assembled in “stacks” to provide a denser memory packing (e.g. Intel, GPU).



- NVIDIA GPU
  - NVLINK, high-speed link (80 Gb/s) to replace PCI-E (16 Gb/s).
  - Unified Memory between CPU and GPU to avoid separate memory allocations.
  - GPU + IBM Power8 for new hybrid supercomputer (OpenPower).



- Intel Xeon PHI (Knights Landing)
  - Upgrade to Knights Corner. More memory and cores, faster internal network and possibility to boot as standalone host.

Energy Efficiency

- Hardware sensors can be integrated into batch systems to report the energy consumption of a batch job.
- Could be used to charge users according to energy consumed instead of resources reserved.

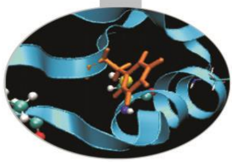
PowerDAM commands

Measures directly the energy in kWh (=3600 kJ).
Current implementation still very experimental.

```
ets --system=Eurora --job=429942.node129

EtS is: 0.173056 kWh
Computation:     99 %
Networking:      0 %
Cooling:         0 %
Infrastructure: 0 %
```

Wrap-up



- HPC is only possible via parallelism and this must increase to maintain performance gains.
- Parallelism can be achieved at many levels but because of limited code scalability with traditional cores increasing role for accelerators (e.g. GPUs, MICs). The Top500 is becoming now becoming dominated by hybrid systems.
- Hardware trends forcing code re-writes with OpenMP, OpenCL, CUDA, OpenACC, etc in order to exploit large numbers of threads.
- Unfortunately, for many applications the parallelism is determined by problem size and not application code.
- Energy efficiency (Flops/Watt) is a crucial issue. Some batch schedulers already report energy consumed and in the near future your job priority may depend on predicted energy consumption.