



9th Advanced  
School on  
**PARALLEL  
COMPUTING**

February 11 - 15, 2013 - BOLOGNA

# Trends in HPC Architectures and Parallel Programming

Giovanni Erbacci - [g.erbacci@ Cineca.it](mailto:g.erbacci@ Cineca.it)

Supercomputing, Applications & Innovation Department - CINECA

**CINECA, February 11-15, 2013**





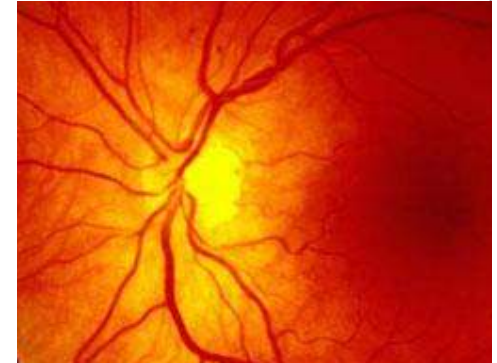
# Agenda

- Computational Sciences
- Trends in Parallel Architectures
- Trends in Parallel Programming
- HPC access offer: ISCRA and PRACE



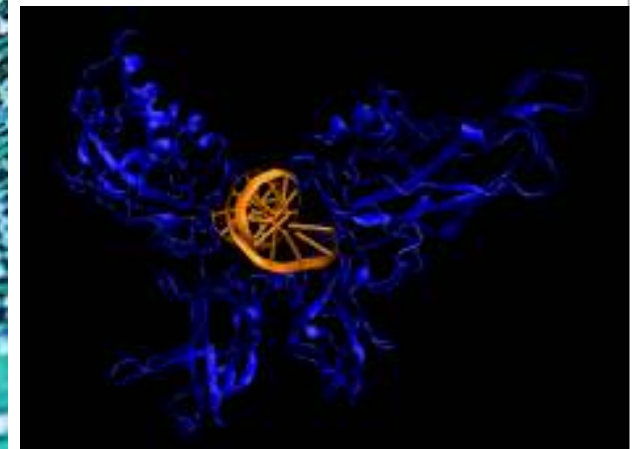
# Computational Sciences

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



Quick **evolution of innovation**:

- Instantaneous communication
- Geographically distributed work
- Increased productivity
- More data everywhere
- Increasing problem complexity
- Innovation happens worldwide





# Technology Evolution

## More data everywhere:

Radar, satellites, CAT scans, weather models, the human genome.

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.

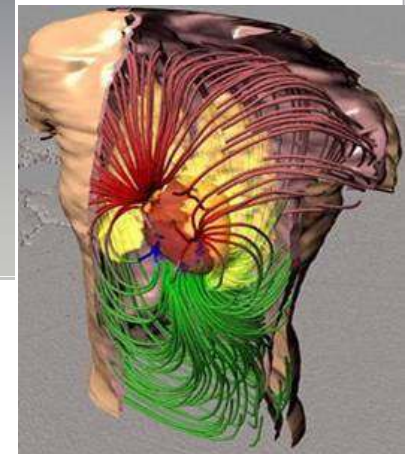
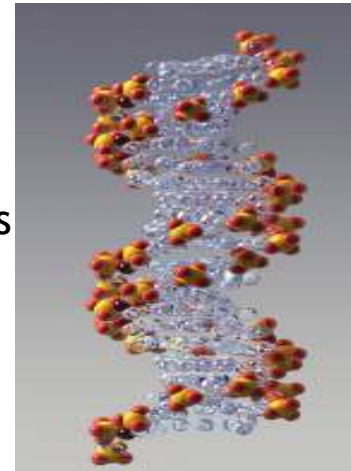
## Increasing problem complexity:

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 Sciences today

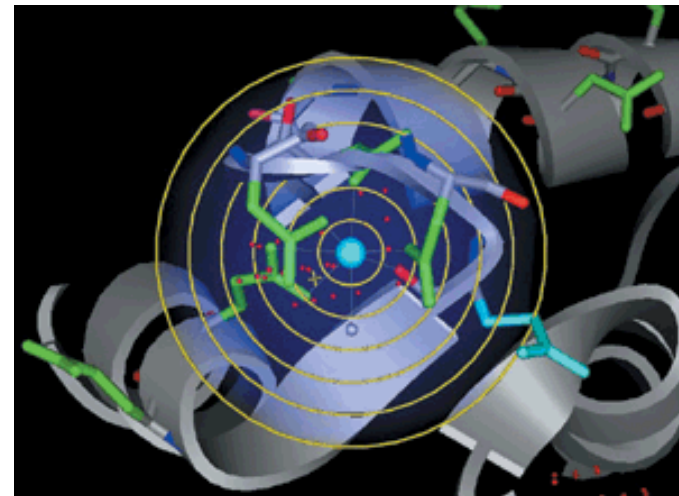
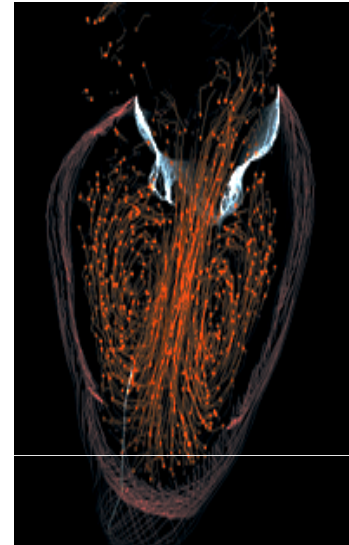
## Multidisciplinary problems

## Coupled applications

- 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

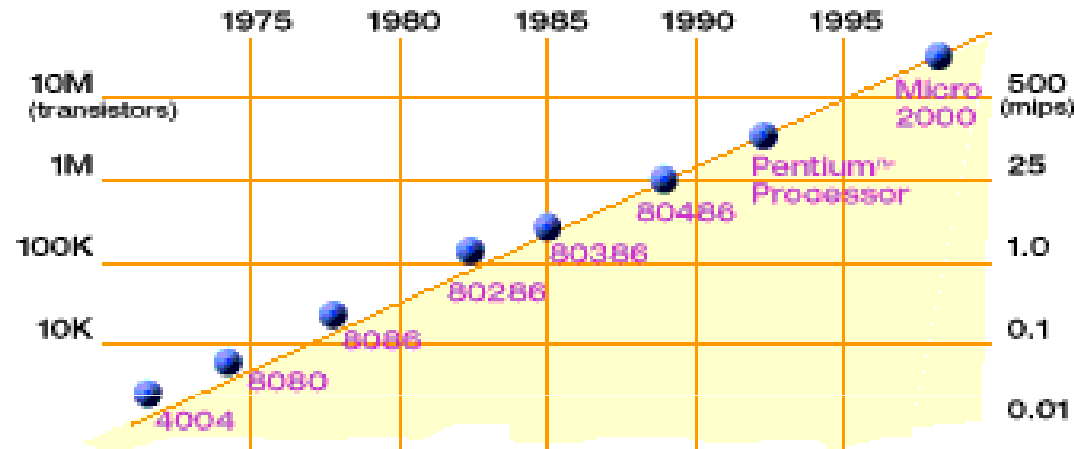
## Large amount of data

## Complex mathematical models





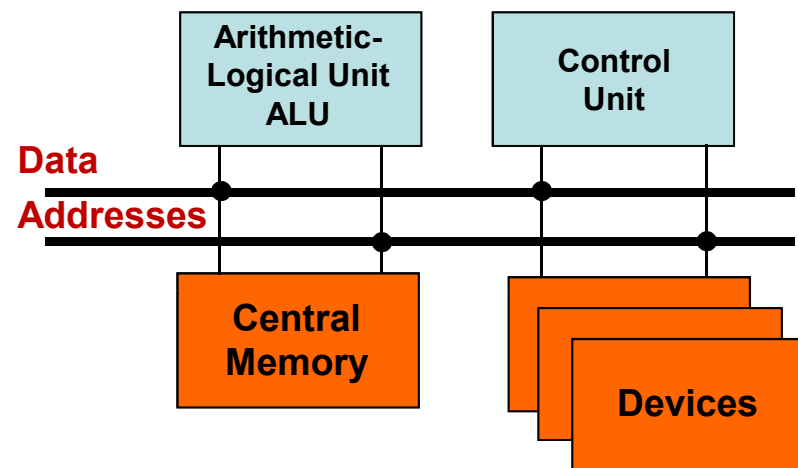
# Moore's Law



- Empirical law which states that the complexity of devices (number of transistors per square inch in microprocessors) doubles every 18 months..
- **Gordon Moore**, INTEL co-founder, **1965**
- It is estimated that Moore's Law still applies in the near future but applied to the
- number of cores per processor



# Other factors that affect Performance



In addition to processor power, other factors affect the performance of computers:

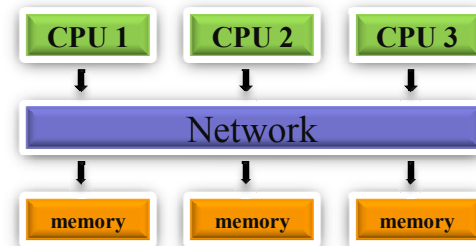
- Size of memory
- Bandwidth between processor and memory
- Bandwidth toward the I/O system
- Size and bandwidth of the cache
- Latency between processor, memory, and I/O system



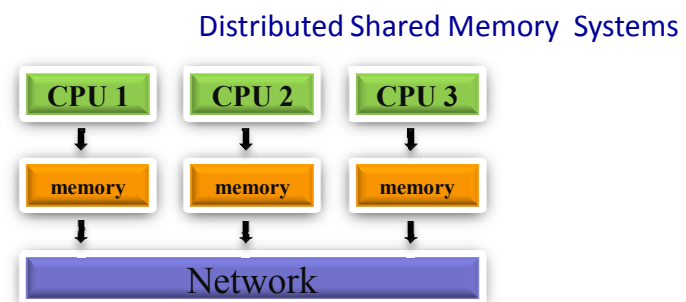
# HPC Architectures

Performance Comes from:

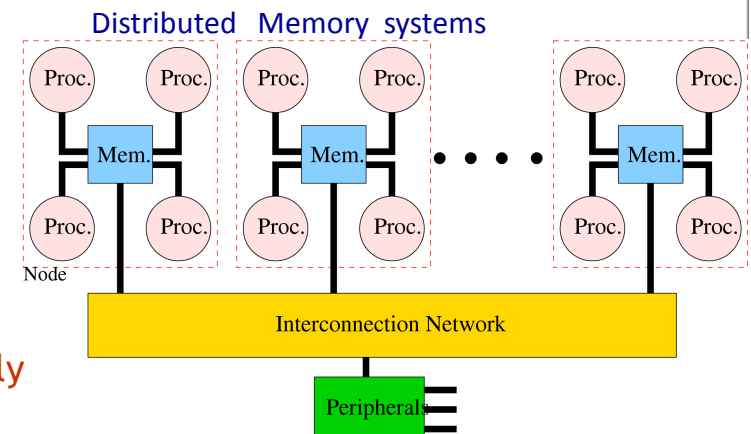
- **Device Technology**
  - Logic switching speed and device density
  - Memory capacity and access time
  - Communications bandwidth and latency
- **Computer Architecture**
  - Instruction issue rate
    - Execution pipelining
    - Reservation stations
    - Branch prediction
    - Cache management
  - Parallelism
    - Number of operations per cycle per processor
      - Instruction level parallelism (ILP)
      - Vector processing
    - Number of processors per node
    - Number of nodes in a system
  - Hybrids, combining
    - standard processors with accelerators.



Shared Memory Systems



Distributed Shared Memory Systems



The macro-architecture of HPC systems is presently uniform: but this aspect will change soon!





# HPC architectures / 1

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

- 1 Power consumption has become a primary headache.
- 2 Processor speed is never enough.
- 3 Network complexity/latency is a main hindrance.
- 4 There is still the memory wall.

Interestingly, solutions for point 1 and 2 can often be combined.

Since a few years computational accelerators of various kinds are offered that may address speed and power consumption.

Fitting accelerators into a general purpose system:

This is a general problem that is met by at least AMD and Intel.

Goal: Let accelerators communicate directly with the system's memory and General Purpose Computational Cores.



# HPC Architectures: Parameters affecting Performance

- Peak floating point performance
- Main memory capacity
- Bi-section bandwidth
- I/O bandwidth
- Secondary storage capacity
- Organization
  - Class of system
  - # nodes
  - # processors per node
  - Accelerators
  - Network topology
- Control strategy
  - MIMD
  - Vector, PVP
  - SIMD
  - SPMD



# HPC systems evolution

## Vector Processors

- Cray-1

## SIMD, Array Processors

- Goodyear MPP, MasPar 1 & 2, TMC CM-2

## Parallel Vector Processors (PVP)

- Cray XMP, YMP, C90 NEC Earth Simulator, SX-6

## Massively Parallel Processors (MPP)

- Cray T3D, T3E, TMC CM-5, Blue Gene/L

## Commodity Clusters

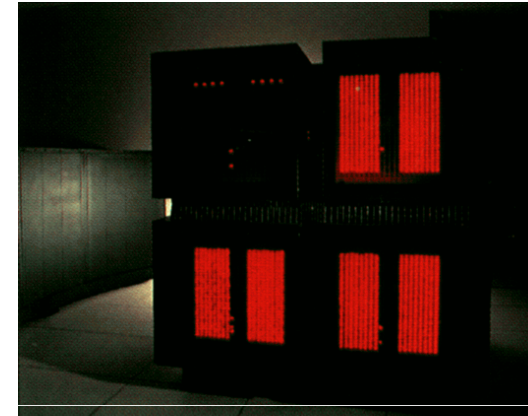
- Beowulf-class PC/Linux clusters
- Constellations

## Distributed Shared Memory (DSM)

- SGI Origin
- HP Superdome

## Hybrid HPC Systems

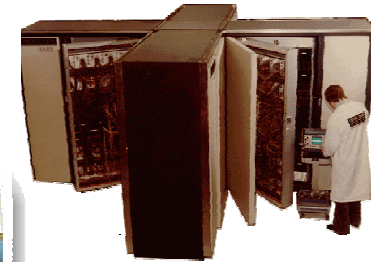
- Roadrunner
- Chinese Tianhe-1A system
- GPGPU systems





# HPC systems evolution in CINECA

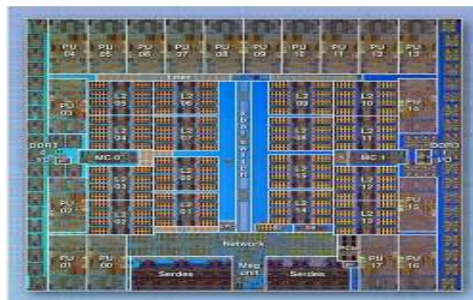
1969:	CDC 6600	1 <sup>st</sup> system for scientific computing
1975:	CDC 7600	1 <sup>st</sup> supercomputer
1985:	Cray X-MP/4 8	1 <sup>st</sup> vector supercomputer
1989:	Cray Y-MP/4 64	
1993:	Cray C-90/2 128	
1994:	Cray T3D 64	1 <sup>st</sup> parallel supercomputer
1995:	Cray T3D 128	
1998:	Cray T3E 256	1 <sup>st</sup> 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





## BG/Q in CINECA

- 10 BGQ Frame
- 10240 nodes
- 1 PowerA2 processor per node
- 16 core per processor
- 163840 cores
- 1GByte / core
- 2PByte of scratch space
- 2PFlop/s peak performance
- 1MWatt liquid cooled



-16 core chip @ 1.6 GHz

- a crossbar switch links the cores and L2 cache memory together.
- 5D torus interconnect

The Power A2 core has a **64-bit instruction set** (unlike the prior 32-bit PowerPC chips used in BG/L and BG/P

The A2 core have **four threads** and has **in-order dispatch**, execution, and completion instead of out-of-order execution common in many RISC processor designs.

The A2 core has **16KB of L1 data cache** and another **16KB of L1 instruction cache**.

Each core also includes a **quad-pumped double-precision floating point unit:**

Each FPU on each core has four pipelines, which can be used to execute scalar floating point instructions, four-wide SIMD instructions, or two-wide complex arithmetic SIMD instructions.



# PLX Cluster Linux

IBM Server dx360M3 - Compute node

2 x processori Intel Westmere 6c X5645

2.40GHz 12MB Cache, DDR3 1333MHz 80W

48GB RAM su 12 DIMM 4GB DDR3 1333MHz

1 x HDD 250GB SATA

1 x QDR Infiniband Card 40Gbs

2 x NVIDIA m2070 (m2070q su 10 nodi)

Peak performance 32 TFlops

(3288 cores a 2.40GHz)

Peak performance 565 TFlops Single Precision o

283 TFlops Double Precision (548 Nvidia M2070)





# EURORA Prototype

- Funded by PRACE 2IP Eurora aimed at evaluating new architectures for the next generation Tier 0 systems.
- Collaboration between [Eurotech](#) and Cineca
- 64 compute cards
- 128 Xeon SandyBridge
- 16GByte DDR3 1600MHz per node
- 160GByte SSD per node
- 1 FPGA (Altera Stratix V) per node
- IB QDR interconnect
- 3D Torus interconnect
- 128 Accelerator cards (NVIDIA K20 and INTEL PHI)
- based on a hot water cooling system, delivers 3,150 megaflops per watt of sustained performance
- 26 percent higher than the top system on the most recent Green500 list of the world's most efficient supercomputers.





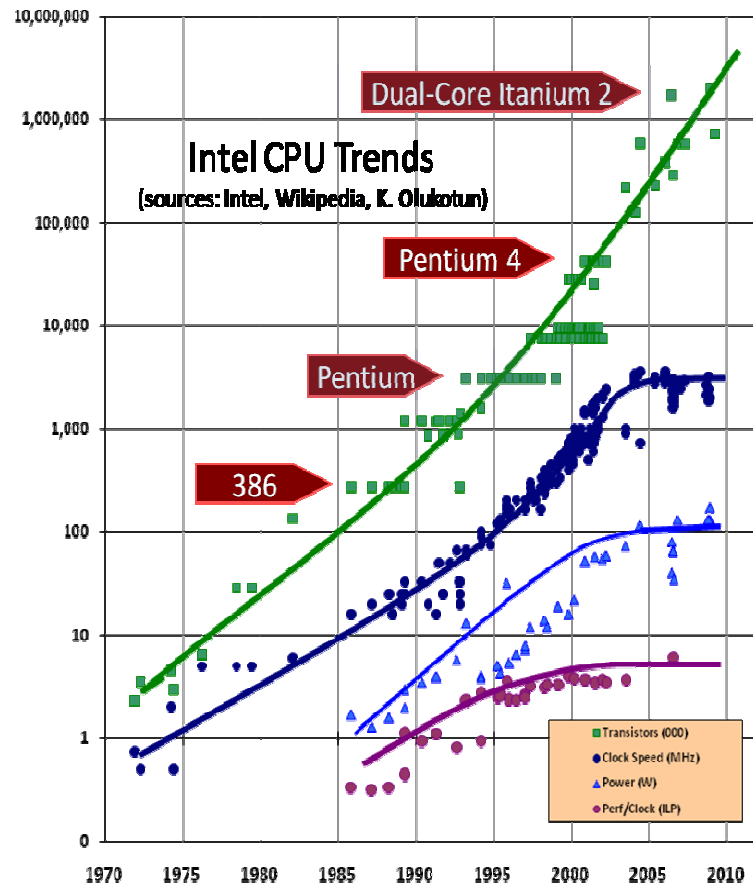
# Top 500: some facts

- 1976** Cray 1 installed at Los Alamos: peak performance 160 MegaFlop/s ( $10^6$  flop/s)
- 1993** (1° Edition Top 500) N. 1 59.7 GFlop/s ( $10^{12}$  flop/s)
- 1997** Teraflop/s barrier ( $10^{12}$  flop/s)
- 2008** Petaflop/s ( $10^{15}$  flop/s): **Roadrunner** (LANL) Rmax 1026 Gflop/s, Rpeak 1375 Gflop/s hybrid system: 6562 processors dual-core AMD Opteron accelerated with 12240 IBM Cell processors (98 TByte di RAM)
- 2011** 11.2 Petaflop/s : K computer (SPARC64 VIIIfx 2.0GHz, Tofu interconnect) RIKEN Japan
  - 62% of the systems on the top500 use processors with six or more cores
  - 39 systems use GPUs as accelerators (35 NVIDIA , 2 Cell , 2 ATI Radeon)
- 2012** 17.5 Petaflop/s(Rmax), 27.2 Petaflop/s (Rpeak) Titan DOE Oak Ridge Nat. Lab Cray XK7, Opteron 6267 2.2 GHz 560,640 cores (including 261,632 NVIDIA K20x accelerator cores.) Cray Gemini interconnect
  - 84.6% of the systems on the top500 use processors with six or more cores
  - 62 systems use GPUs as accelerators (56 GPU, 6 MIC Phi)





# HPC Evolution



## Moore's law is holding, in the number of transistors

- Transistors on an ASIC still doubling every 18 months at constant cost
- 15 years of *exponential* clock rate growth has ended

## Moore's Law reinterpreted

- Performance improvements are now coming from the increase in the number of cores on a processor (ASIC)
- #cores per chip doubles every 18 months *instead of clock*
- 64-512 threads per node will become visible soon
- Million-way parallelism

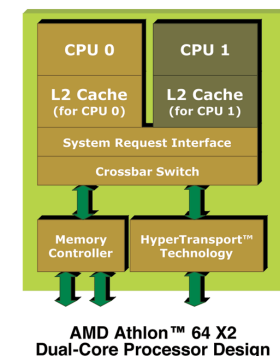
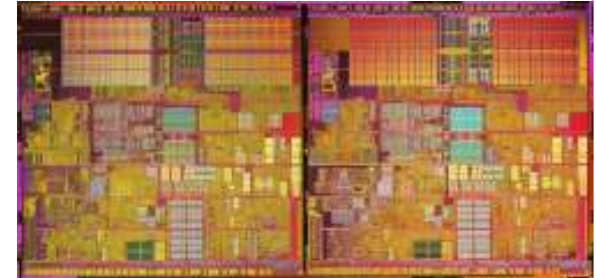
## Heterogeneity: Accelerators

- GPGPU
- MIC



# Multi-core

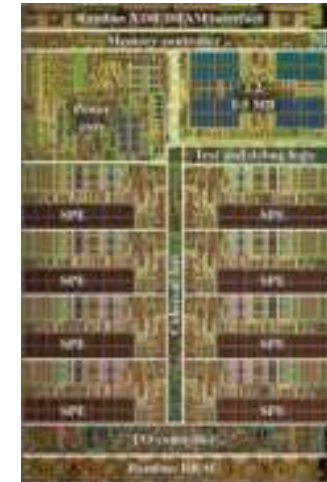
- Motivation for Multi-Core
  - Exploits improved feature-size and density
  - Increases functional units per chip (spatial efficiency)
  - Limits energy consumption per operation
  - Constrains growth in processor complexity
- Challenges resulting from multi-core
  - Relies on effective exploitation of multiple-thread parallelism
    - Need for parallel computing model and parallel programming model
  - Aggravates memory wall
    - Memory bandwidth
      - Way to get data out of memory banks
      - Way to get data into multi-core processor array
    - Memory latency
    - Fragments (shared) L3 cache
  - Pins become strangle point
    - Rate of pin growth projected to slow and flatten
    - Rate of bandwidth per pin (pair) projected to grow slowly
  - Requires mechanisms for efficient inter-processor coordination
    - Synchronization
    - Mutual exclusion
    - Context switching





# Heterogeneous Multicore Architecture

- Combines different types of processors
  - Each optimized for a different operational modality
    - Performance >  $Nx$  better than other  $N$  processor types
  - Synthesis favors superior performance
    - For complex computation exhibiting distinct modalities
- Purpose-designed accelerators
  - Integrated to significantly speedup some critical aspect of one or more important classes of computation
  - IBM Cell architecture (past)
  - FPGA
  - ClearSpeed SIMD attached array processor
- Conventional co-processors
  - Graphical processing units (GPU)
  - MIC
  - Network controllers (NIC)
  - Efforts underway to apply existing special purpose components to general applications
- **Drawback of all accelerators: No standard (yet).**
  - Software Development Kits far from uniform (but improving rapidly, OpenFPGA initiative, OpenCL ...).



**Programming is hard work.**



# 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



# 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

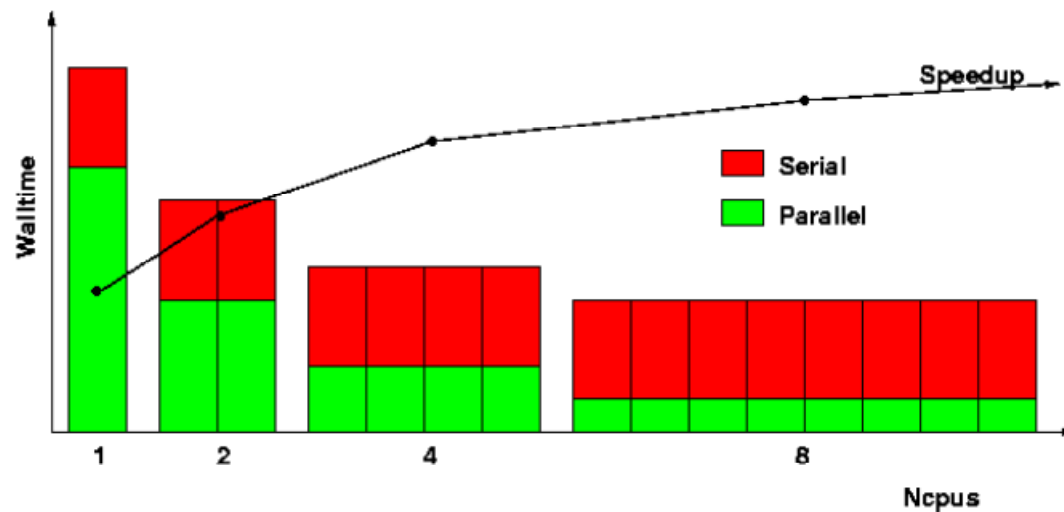
- Complexity is rising dramatically
- Challenges for the applications on Petaflop systems
- Improvement of existing codes will become complex and partly impossible
- The use of  $O(100K)$  cores implies dramatic optimization effort
- New paradigm as the support of a hundred threads in one node implies new parallelization strategies
- Implementation of new parallel programming methods in existing large applications has not always a promising perspective

There is the need for new community codes



# What about parallel App?

- 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).



maximum speedup tends to

$$1 / (1 - P)$$

$P$  = parallel fraction

1000000 core

$$P = 0.999999$$

*serial fraction* = 0.000001



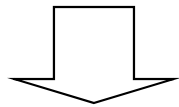
# Programming Models

- **Message Passing (MPI)**
- **Shared Memory (OpenMP)**
- **Partitioned Global Address Space Programming (PGAS) Languages**
  - UPC, Co-array Fortran
- **Next Generation Programming Languages and Models**
  - Chapel, X10
- **Languages and Paradigm for Hardware Accelerators**
  - CUDA, OpenCL
- **Hybrid: MPI + OpenMP + CUDA/OpenCL**

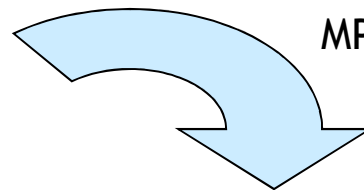


# Trends

Scalar Application

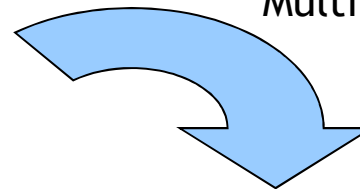


*Vector*



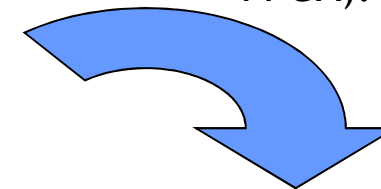
MPP System, Message Passing: MPI

*Distributed  
memory*



Multi core nodes: OpenMP

*Shared  
Memory*



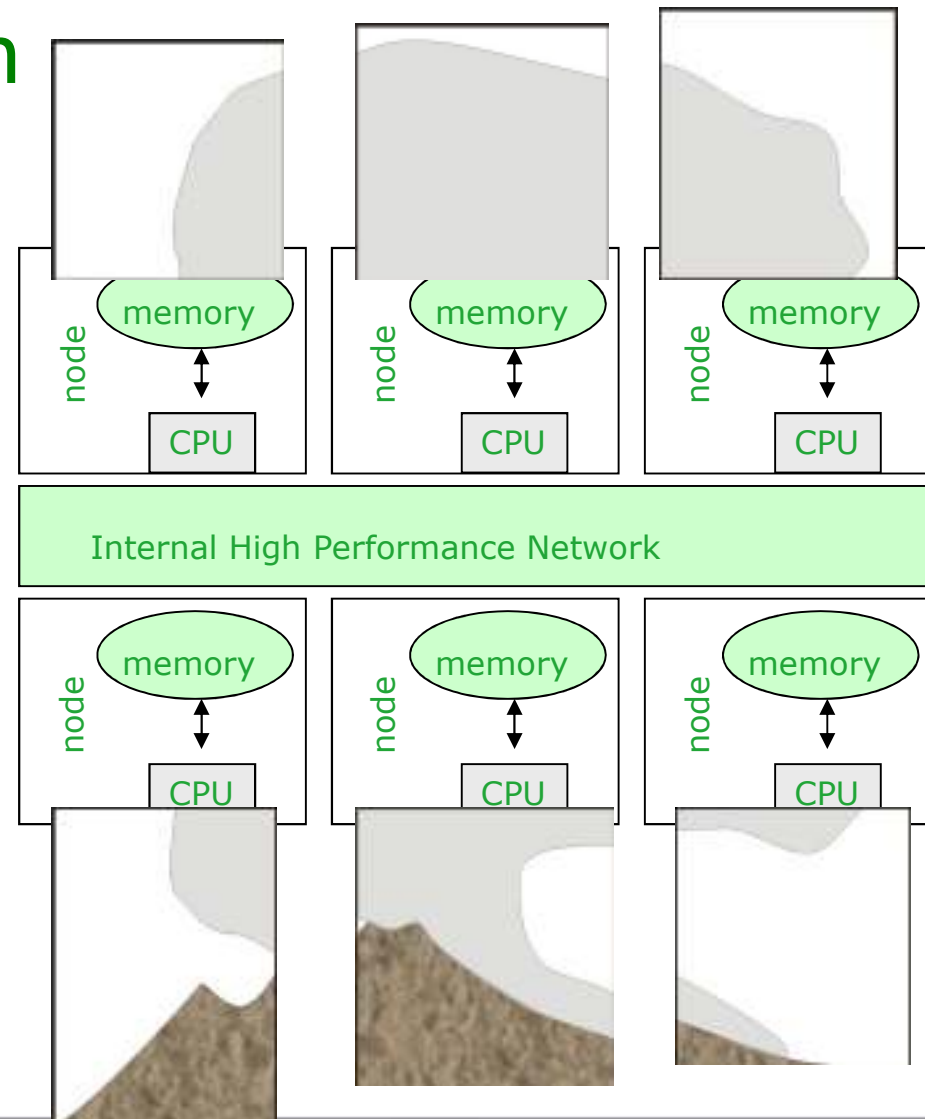
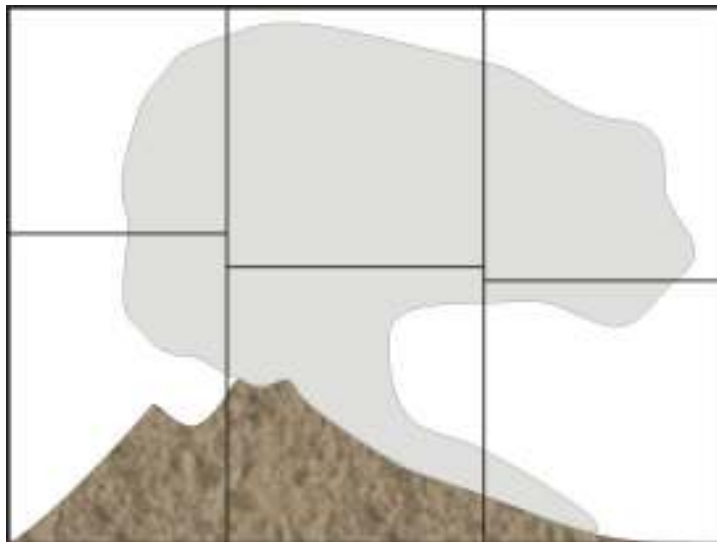
Accelerator (GPGPU,  
FPGA): Cuda, OpenCL

*Hybrid codes*





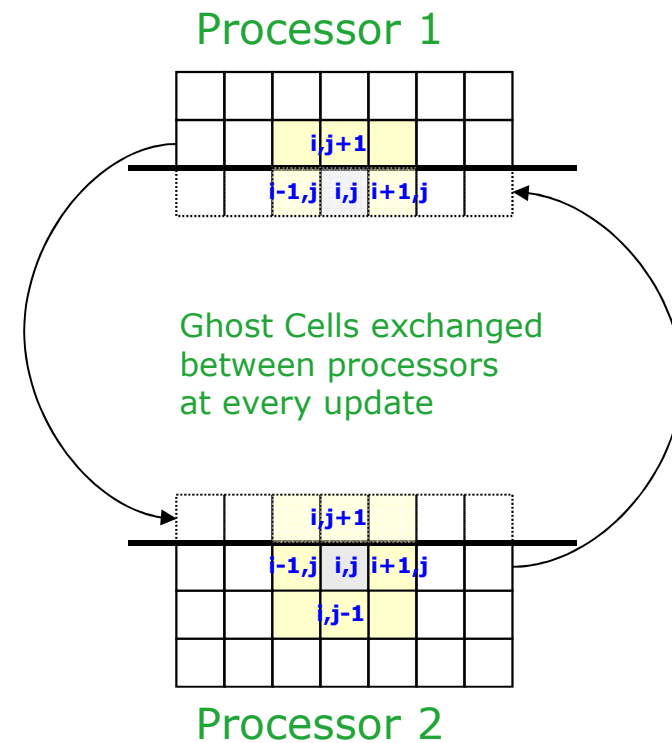
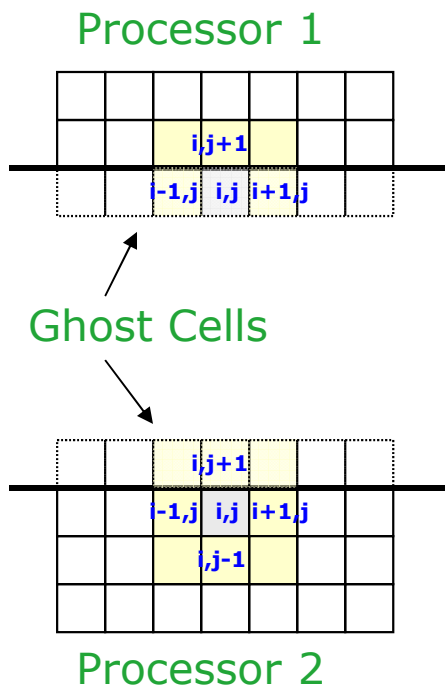
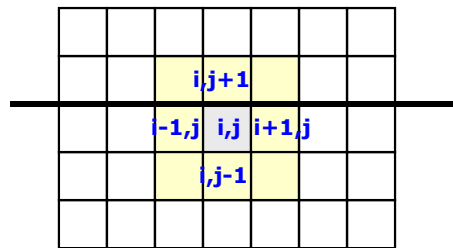
# Message Passing domain decomposition





# Ghost Cells - Data exchange

sub-domain boundaries





# Message Passing: MPI

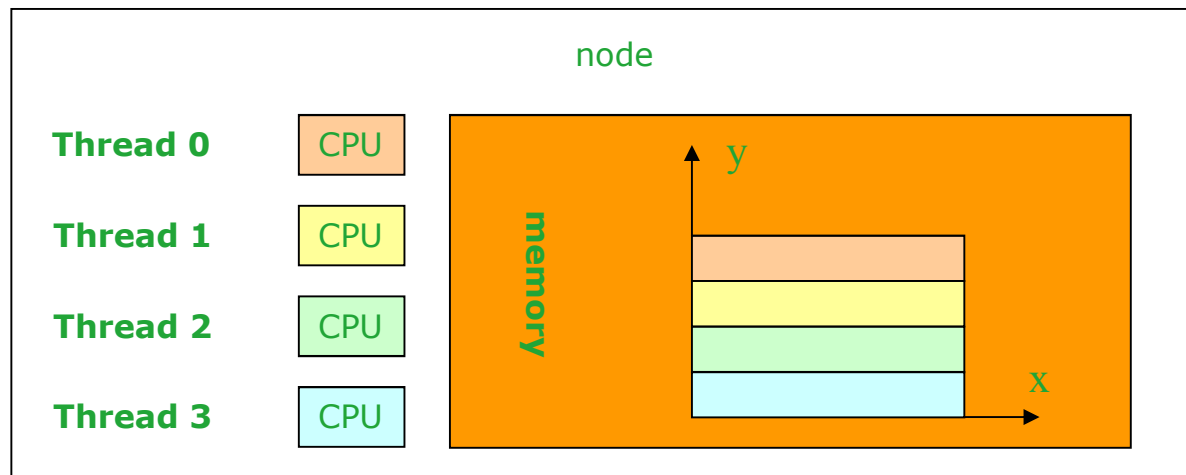
- **Main Characteristic**
- Library
- Coarse grain
- Inter node parallelization (few real alternative)
- Domain partition
- Distributed Memory
- Almost all HPC parallel App

## Open Issue

- Latency
- OS jitter
- Scalability



# Shared memory





# Shared Memory: OpenMP

- **Main Characteristic**
- Compiler directives
- Medium grain
- Intra node parallelization (p-threads)
- Loop or iteration partition
- Shared memory
- Many HPC App

## Open Issue

- Thread creation overhead
- Memory/core affinity
- Interface with MPI

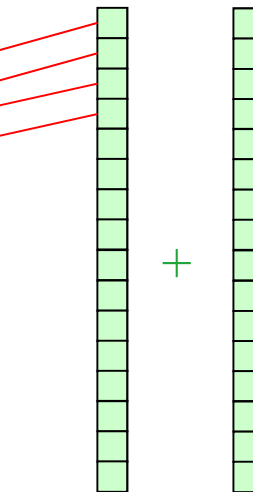
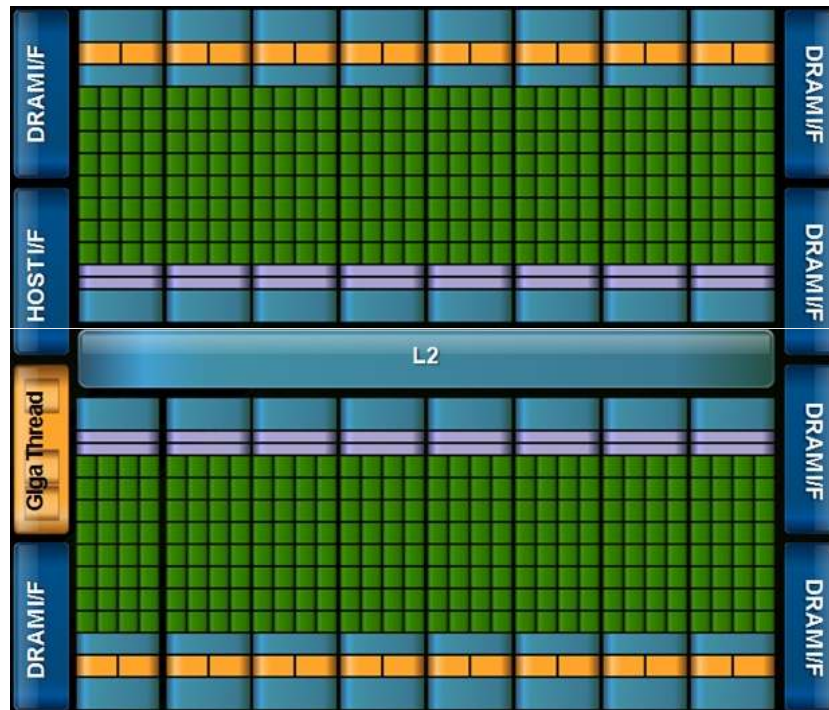


# OpenMP

- `!$omp parallel do`
- `do i = 1 , nsl`
- `call 1DFFT along z ( f [ offset( threadid ) ] )`
- `end do`
- `!$omp end parallel do`
- `call fw_scatter ( . . . )`
- `!$omp parallel`
- `do i = 1 , nzl`
- `!$omp parallel do`
- `do j = 1 , Nx`
- `call 1DFFT along y ( f [ offset( threadid ) ] )`
- `end do`
- `!$omp parallel do`
- `do j = 1 , Ny`
- `call 1DFFT along x ( f [ offset( threadid ) ] )`
- `end do`
- `end do`
- `!$omp end parallel`



# Accelerator/GPGPU



Sum of 1D array



# CUDA sample

```
void CPUCode( int* input1, int* input2, int* output, int length) {  
    for ( int i = 0; i < length; ++i ) {  
        output[ i ] = input1[ i ] + input2[ i ];  
    }  
}
```

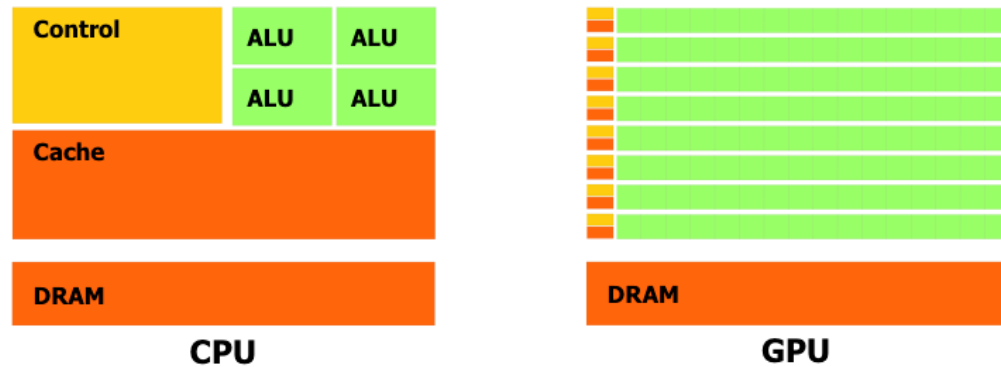
```
__global__ void GPUCode( int* input1, int*input2, int* output, int length) {  
    int idx = blockDim.x * blockIdx.x + threadIdx.x;  
    if ( idx < length ) {  
        output[ idx ] = input1[ idx ] + input2[ idx ];  
    }  
}
```

Each thread execute one loop iteration





# CUDA - OpenCL



## Main Characteristic

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

## Open Issue

- Memory copy
- Standard
- Tools
- Integration with other languages



## Hybrid (MPI+OpenMP+CUDA+...

- Take the positive of all models
- Exploit memory hierarchy
- Many HPC applications are adopting this model
- Mainly due to developer inertia
- Hard to rewrite million of source lines

**...+python)**



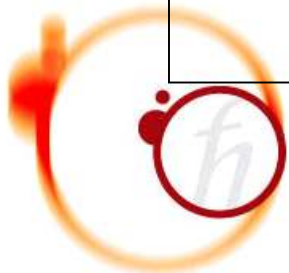
# Hybrid parallel programming (example)

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/>



# PRACE

- Partnership for Advanced Computing in Europe
- <http://www.prace-ri.eu/>
- The mission of PRACE is to enable high impact scientific discovery and engineering research and development across all disciplines to enhance European competitiveness for the benefit of society.
- PRACE seeks to realize this mission through world class computing and data management resources and services through a peer review process.
- PRACE is established as an international non-profit association with its seat in Brussels. It has 25 member countries .
- Four Hosting Members (France, Germany, Italy and Spain) provide Multi-PFlop/s Tier-0 Systems





# PRACE Tier-0 Systems

- JUQUEENE BM BG/Q, 5.87 Pflop/s



- CURIE BULL x86 system , 2 PFlop/s



- HERMIT Cray XE6, 1 PFlop/s



- SuperMUC IBM x iDataPlex , 3 PFlop/s



- FERMII BM BG/Q, 2.1 PFlop/s



- Marenostrum IBM X iDataplex, 1 PFlop/s



# PRACE: Tier-0 calls



PRACE offers 3 different forms of access to its resources:

- **Project Regular Access calls**

It is open to researchers and their collaborators from recognized academic institutions and industry for projects deemed to have significant European and international impact.

Calls for Proposals are issued twice a year and are evaluated by leading scientists and engineers in a peer-review process

- **Multi-Year Project Access**

It is available to major projects or infrastructures that can benefit from PRACE resources and for which more than a single year of access is needed.

- **Preparatory Access**

It is a simplified form of access for limited resources for the preparation of resource requests in response to Project Access Calls for Proposals.

Type **A** (scalability tests)

Type **B** (Enabling + Scalability tests)

Type **C** (Enabling + Scalability tests with PRACE involvement)



# CINECA ISCRA calls

ISCRA - Italian SuperComputing Resource Allocation

Web suite: [iscra.cineca.it](http://iscra.cineca.it)

Project classes on the new BG/Q machine are:

- **Class A**: greater than 5 Million core hours
- **Class B**: between 2 and 5 Million core hours
- **Class C**: up to 2 Million core hours.

Class A and B calls are issued twice a year

The next call for projects of Class A and B will open on 13th February 2013 and will close on 26th March 2013.

Class C projects are received through continuous submission and reviewed two times per year.



# Conclusions

- Parallel programming trends in extremely scalable architectures
  - Exploit millions of ALU
  - Hybrid Hardware
  - Hybrid codes
  - Memory Hierarchy
  - Flops/Watt (more than Flops/sec)
  - I/O subsystem
  - Non volatile memory
  - Fault Tolerance!
- 
- Advanced HPC infrastructures to enhance Science through computational methods