UNIVERSITÉ CÔTE D'AZUR

Large-scale distributed computing systems

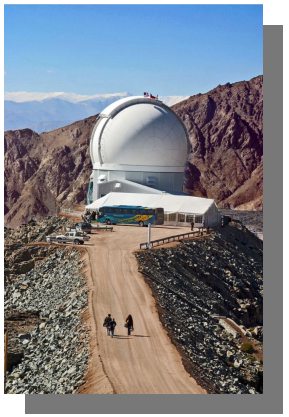
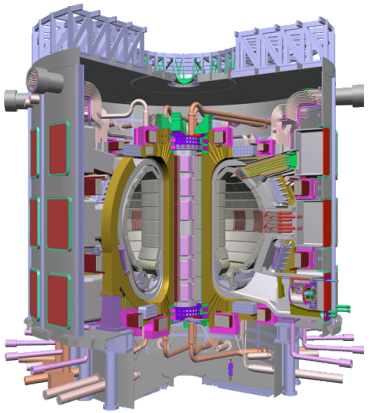
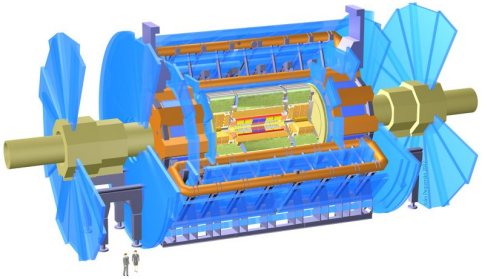
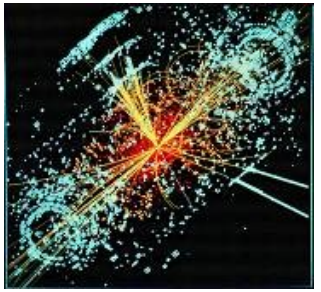
Weeks 1 and 2

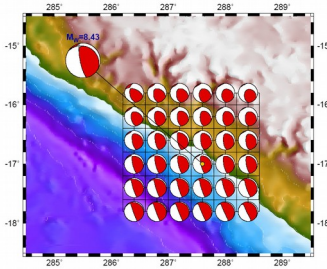
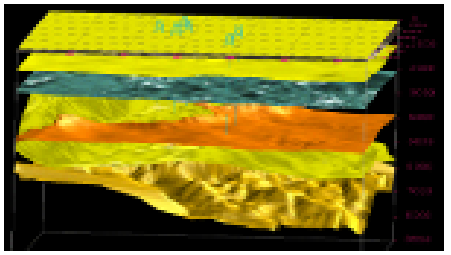
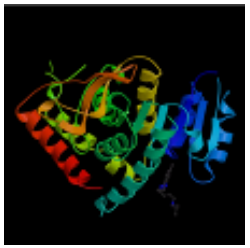
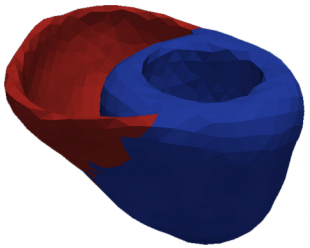
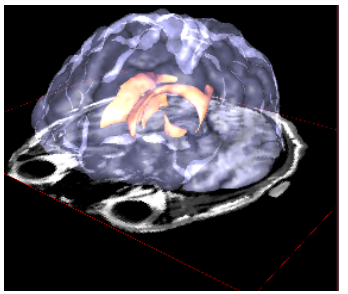
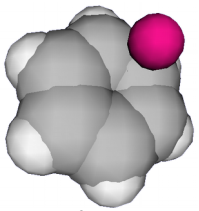
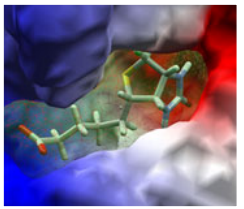
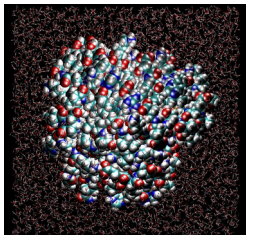
December 2016

Johan Montagnat
CNRS, I3S, SPARKS
http://www.i3s.unice.fr/~johan/

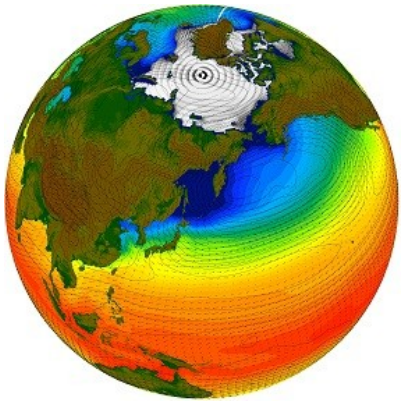
Science

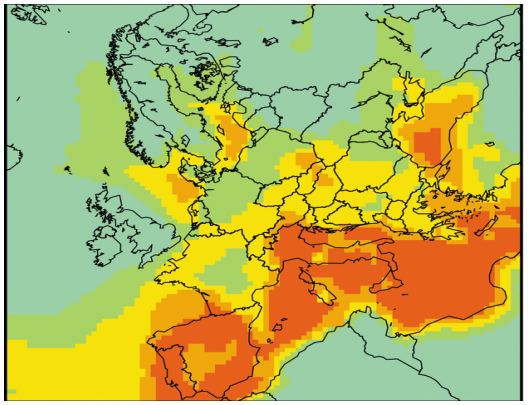
- Astronomy
- High energy physics
  - Fundamental particles
  - Nuclear fusion
- Earth observation
  - Satellite acquisition
  - Environmental sciences
- Biomedical sciences
- Physics
- Chemistry
- Human sciences
- Complex systems

Society





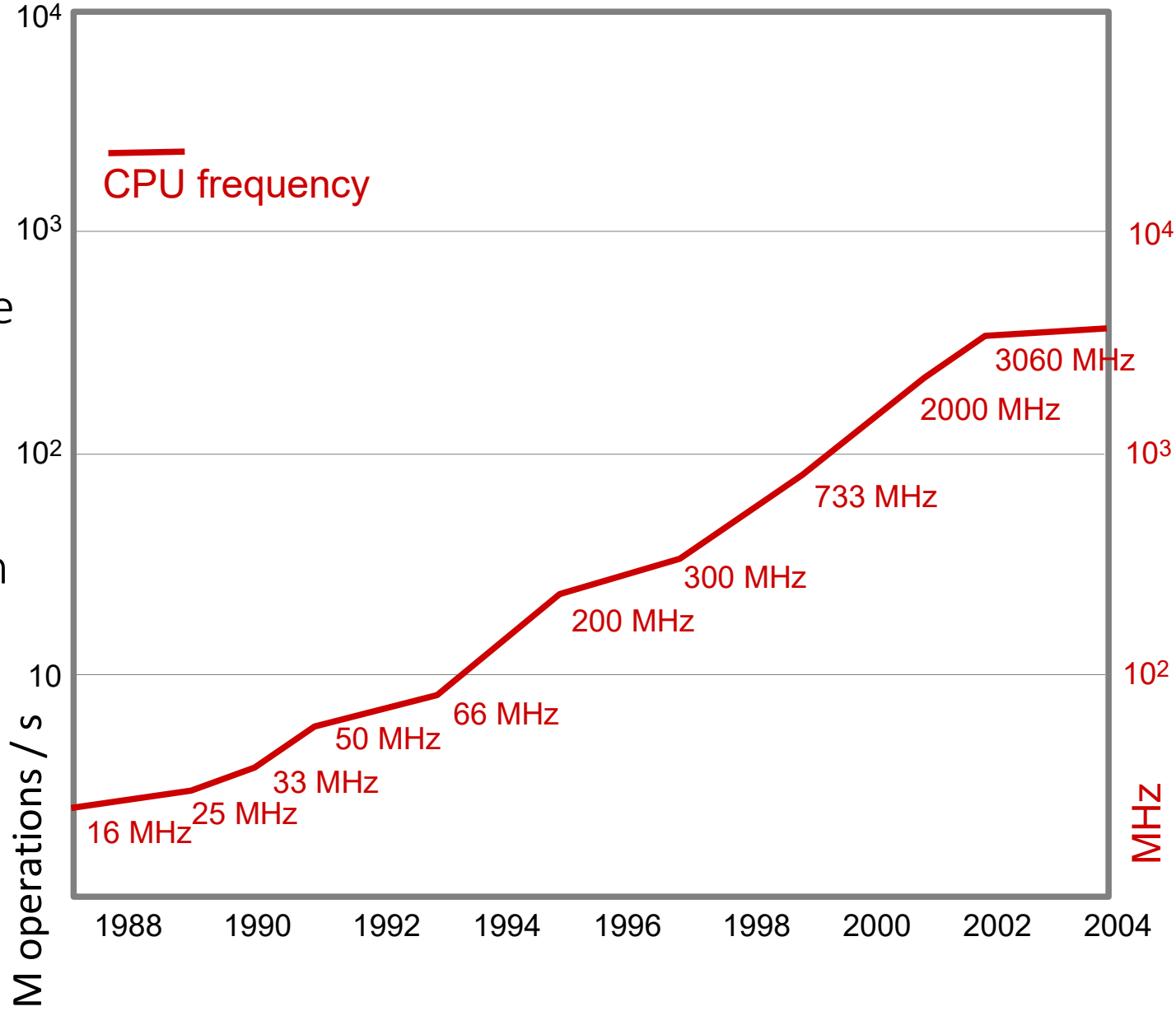
- Forecasting
  - Meteorology
  - Climate
  - Pollution…
- Disasters prevention
  - Seismology
  - Flooding…
- Renewable energy
- Smart cities
- Information society
  - Network infrastructures and data centres
  - Social networks

Digital age

- Digital data production and archiving
  - Improved sensors (spatial, temporal) and connectivity
  - Online availability
  - Production of (Meta)data from data
- Many opportunities
  - Ever growing digital applications portfolio
  - Data secondary reuse
  - Modeling, simulation and prediction
  - No (more) science without data
- Many threats
  - Limited data analysis capability
  - Overall power consumption
  - Long term data archiving limitation

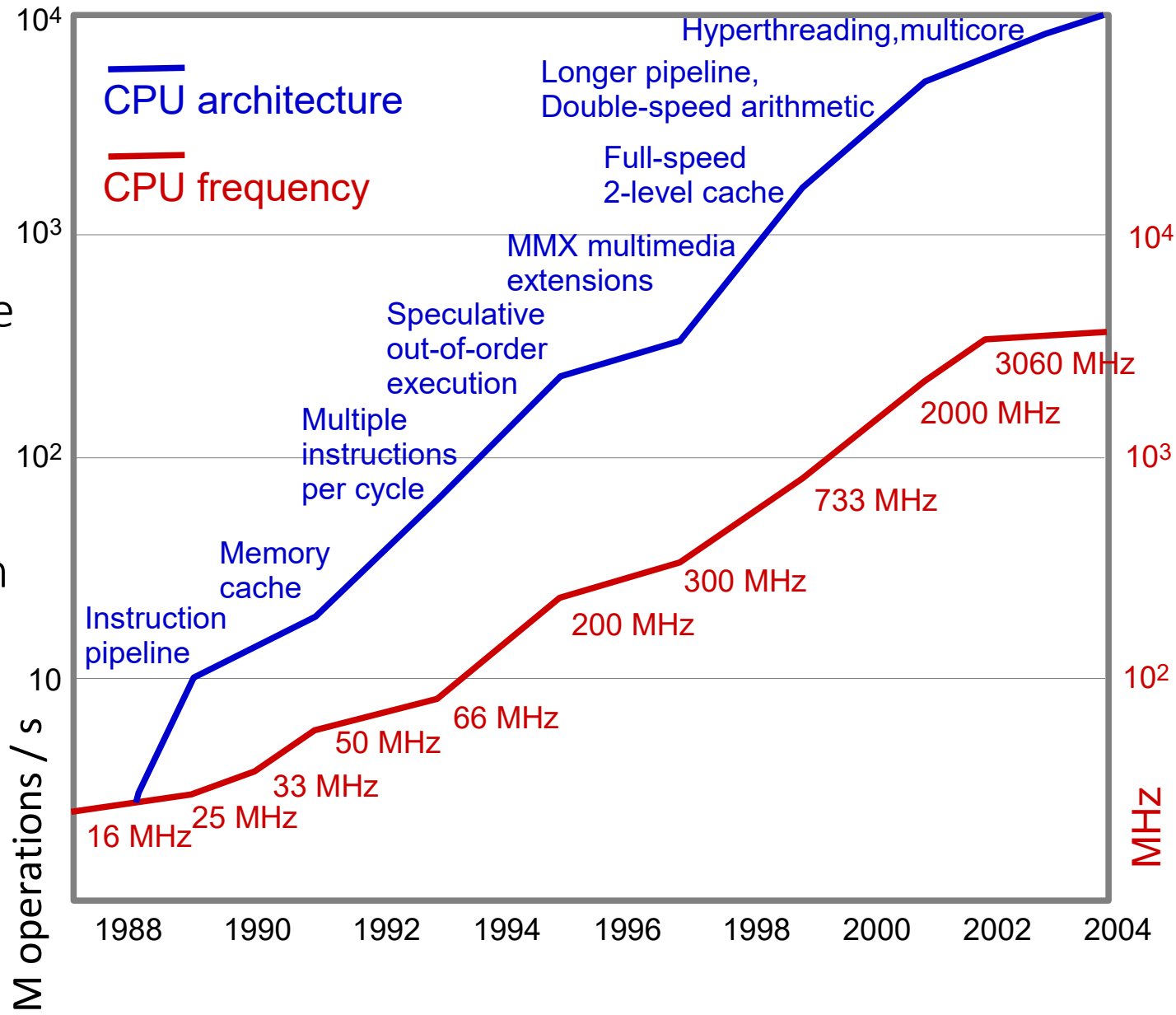
How to improve computer performance?

- Increasing CPU frequency
  - In the 80's-90's, frequency was driving performance improvement
  - CPU frequency is directly related to power consumption
  - Acceptable limit reached around 2002
    - F < 4 GHz



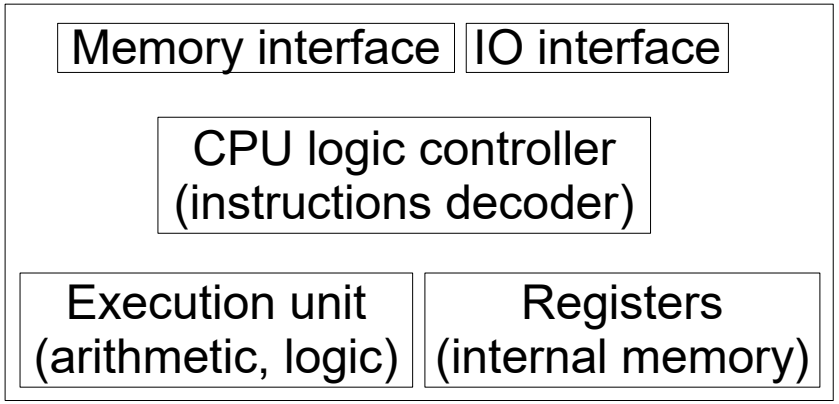
How to improve computer performance?

- Increasing CPU frequency
  - In the 80's-90's, frequency was driving performance improvement
  - CPU frequency is directly related to power consumption
  - Acceptable limit reached around 2002
    - F < 4 GHz
- Architectural improvements



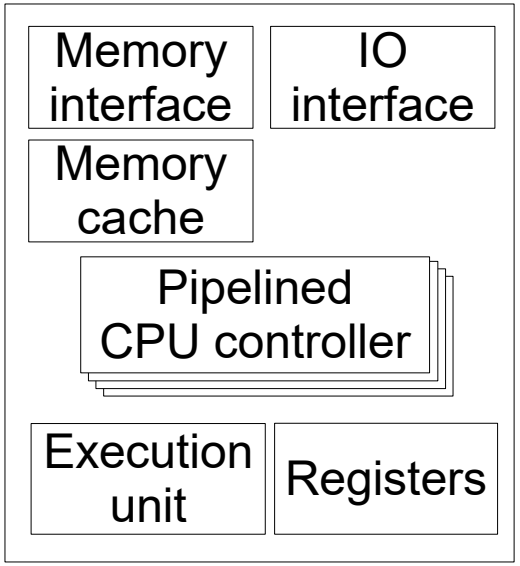
CPU basics

- Coarse CPU architecture overview
  - CPU logic controller
    - Fetch and decode instructions
  - Execution unit
    - Compute (arithmetic and logic)
  - Registers
    - Extremely fast, limited internal memory
  - Interfaces
    - Fast memory bus, slower devices bus



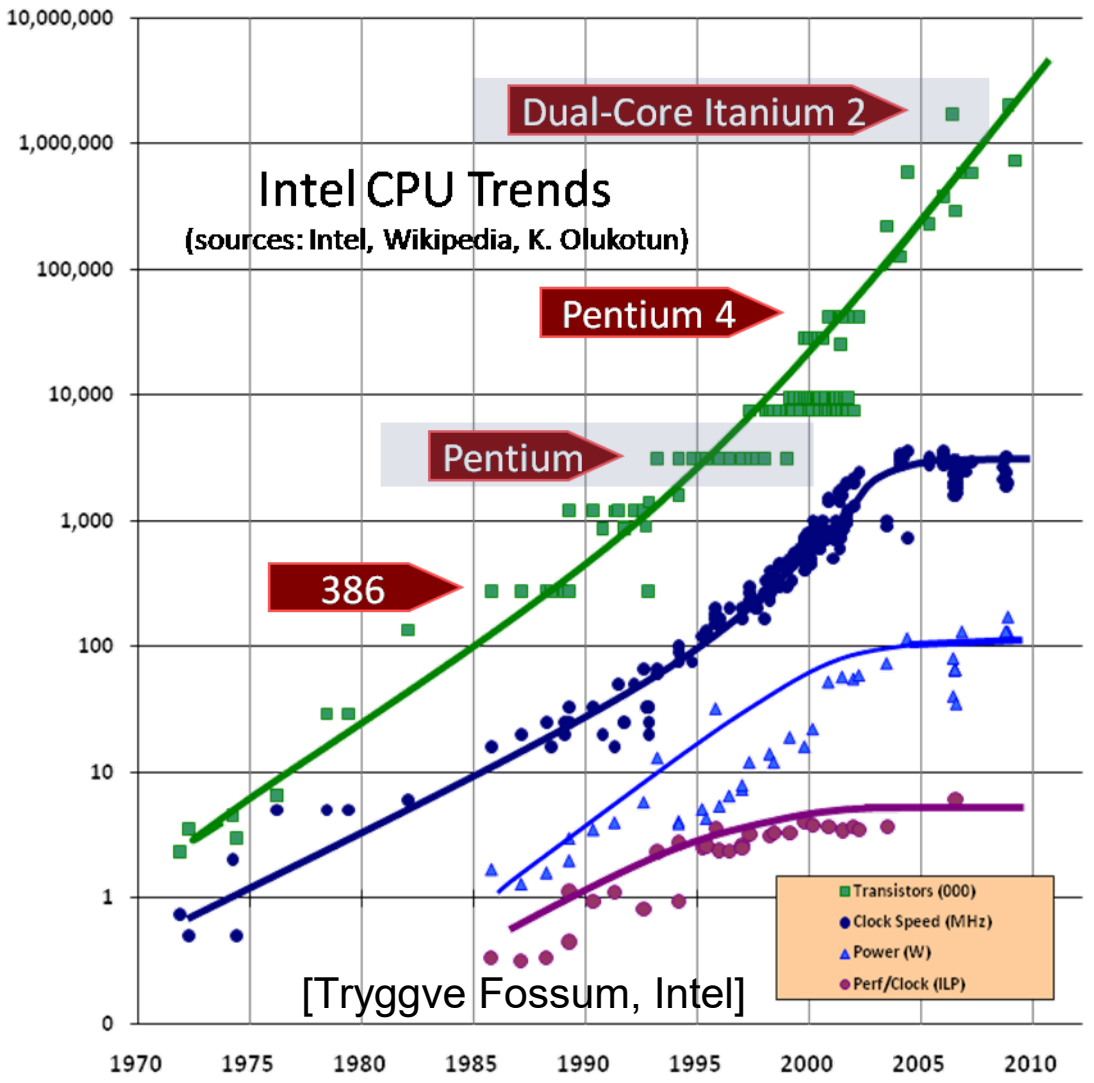
Some architectural optimizations

- Some architectural optimizations
  - Caches to speed-up memory accesses
    - Avoids instructions/data starvation
    - Up to 3 layers, separate data and instruction caches
  - Pipelining (multiple instructions processing)
    - Improve throughput
  - Branch prediction
    - Anticipate possible future processing



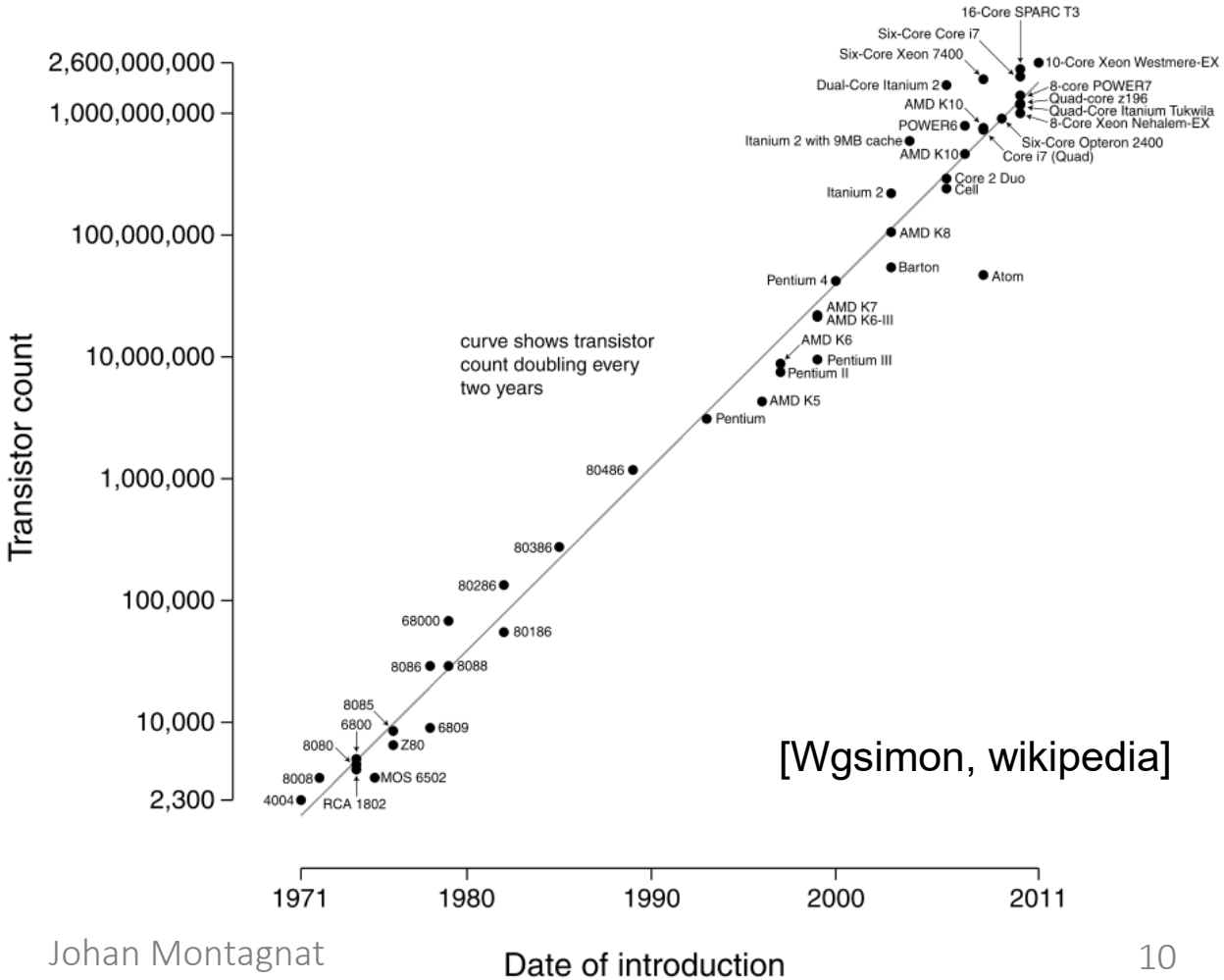
Limiting factors

- CPU performance increase slows down
  - Only number of transistors still exponentially increases
  - Power consumption reached acceptable limits around 2002
  - Frequency suddenly capped as a consequence
  - ILP (Instruction Level Parallelism = capability to process multiple instructions per clock beat) faces chip design complexity limitations
- Current trend is to multiply computing units



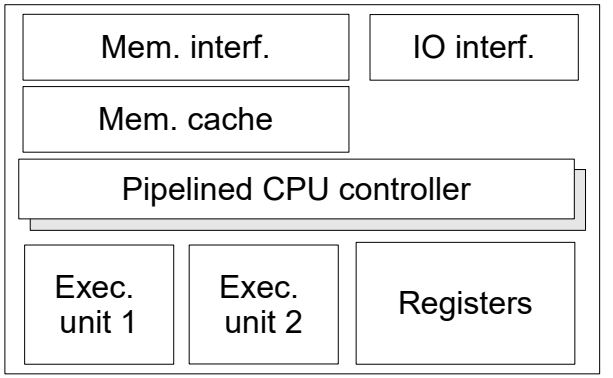
Transistors miniaturization

- Moore's Law: “number of transistors in microchip doubles every 18 months” (later on revised: every 2 years growth)
- Hold true for 40 years!
- But...
  - Number of transistors is not the same as computing power
  - We are getting close to the known limit in miniaturization (quantum tunneling effect and atoms size)
  - Technological breakthroughs happen though

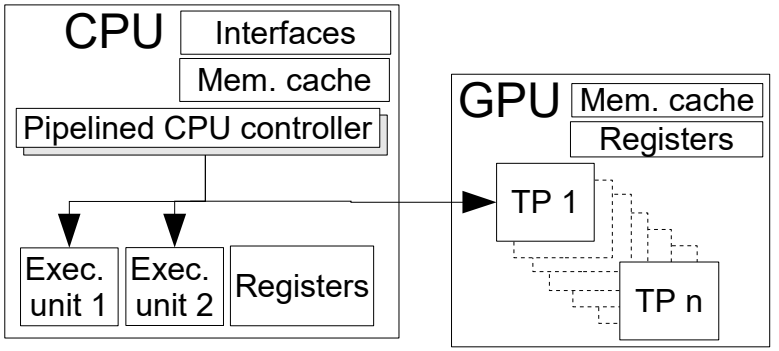


More transistors, no more power

- Multiply execution units
  - Doubling CPU execution units (hyperthreading)
    - 15-30% performance increase

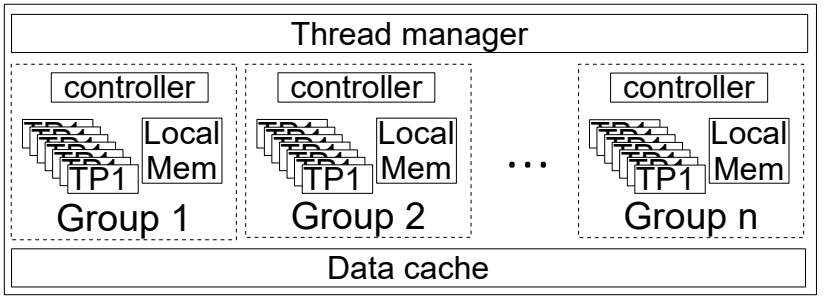


  - GPU execution units
    - Hundreds of units (Thread Processors)
    - Performance depends on application capability to exploit all execution units



General-Purpose computing on Graphical Processing Units (GPGPU / accelerators)

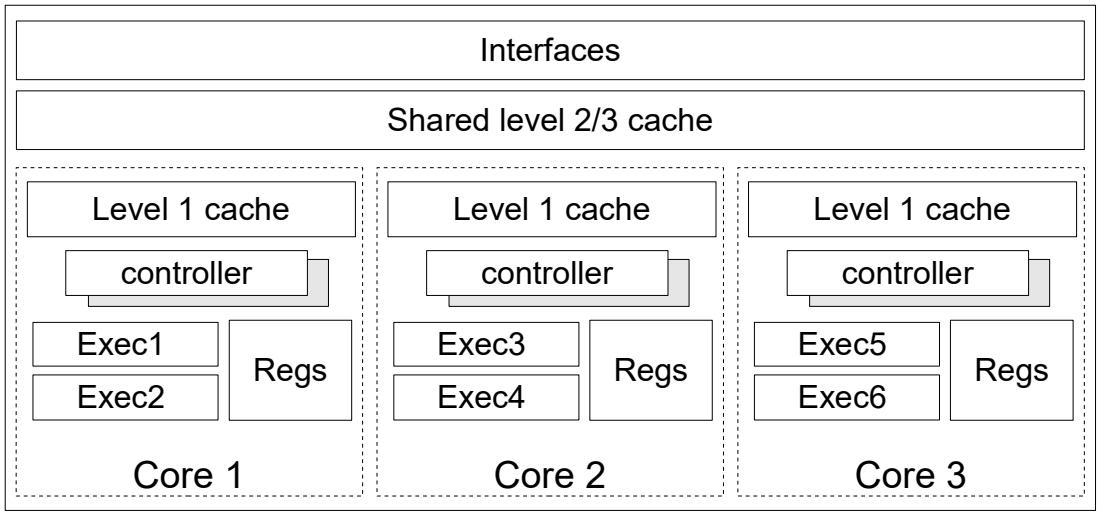
- Many-cores architecture
  - 100's (limited capability) “Thread Processors” (TP)
  - Simple Instruction Multiple Data (SIMD) per group
  - Small memory buffer per core
  - Fewer specialized units
  - Many cache coherence problems
  - Main memory access bottleneck



- SIMD parallel computation framework
  - Vectorial computation-like model
  - Very efficient for graphics but also other kind of data-independent computation (hence General Purpose)
- Very high end theoretical computing power
  - Limited computing capability per core, but the power of many
  - Specialized for some kinds of computation
  - Requires specific languages and algorithms

More transistors, no more power

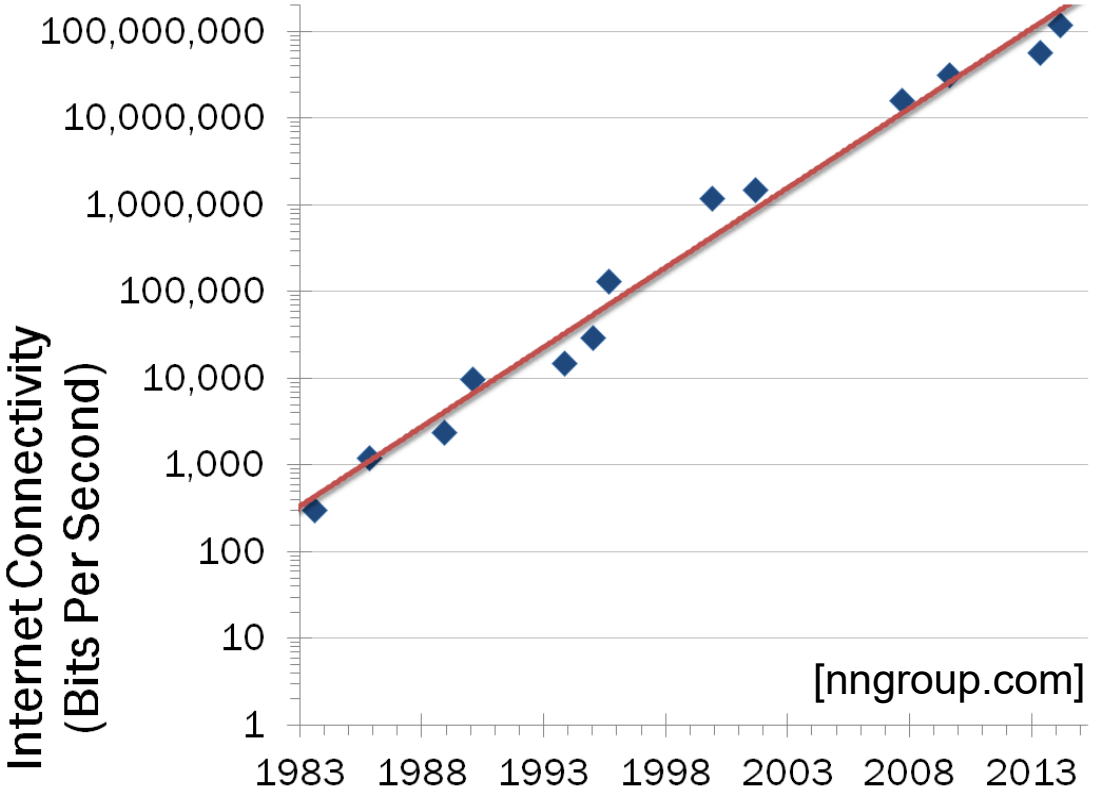
- Multiple cores on chip
  - One CPU, several cores
  - Some sharing
    - Memory caches (with coherency)
    - Shared-memory (through cache) or message passing inter-cores communication



Network performance

- Network bandwidth growth remains exponential
  - Internet back-bone: doubles every 6 months (Gilder's Law)
  - End user's connection: doubles every 21 months (Nielsen's Law)
  - WiFi connection (from 802.11 at 2 Mbit/s in 1997 to 802.11ac at 1.3 Gbits/s in 2013)

Nielsen's Law

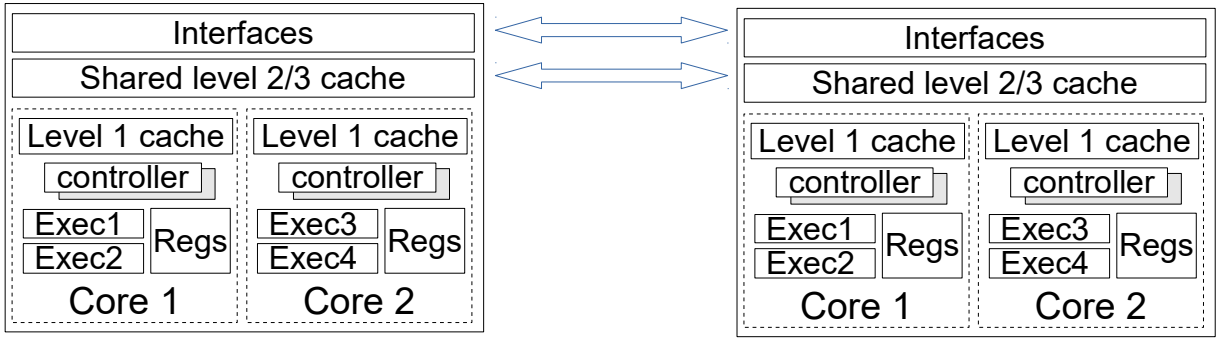


Early conclusions

- Computing power of individual units reaches limitations
  - But need for computing power still increases
- Communication technologies growth is faster than CPU computing power growth
  - Data exchanges become “cheaper”
- Duplicating computing units is a key to increased performance

CPU performance increase in a power consumption limited world

- Multiple CPUs on mother-board
  - Several CPUs
  - Some specific CPU architectures
    - Optimize IO access bus
    - Optimized inter-CPU communication



- Many CPUs with dedicated communication network
  - Also known as super-computer
- Many CPUs inter-connected through standard networking technologies (ethernet, infiniband)
  - Also known as computing cluster

Parallel and distributed computing hierarchy

1. Multi-instructions per clock beat CPUs (architectural optimization)

2. Multiple computing units per CPU (hyperthreading)

3. Multiple cores per CPU / GPU

4. Multiple CPUs connected over a private network (super-computers)

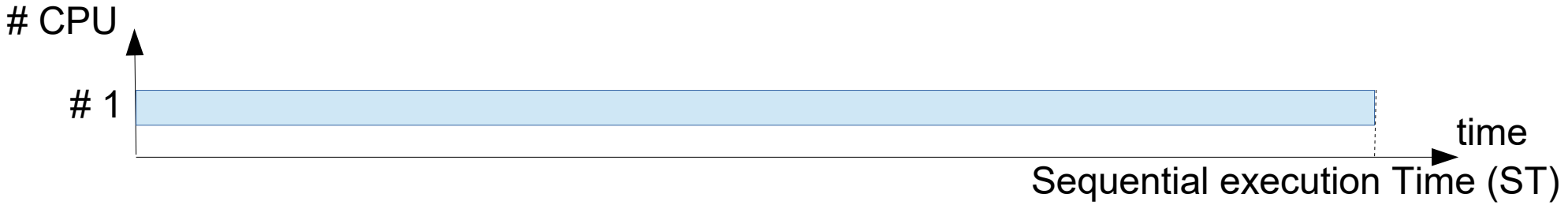
5. Multiple CPUs connected over a LAN (cluster computing)

6. Multiple CPUs connected over a WAN (grid / cloud computing)

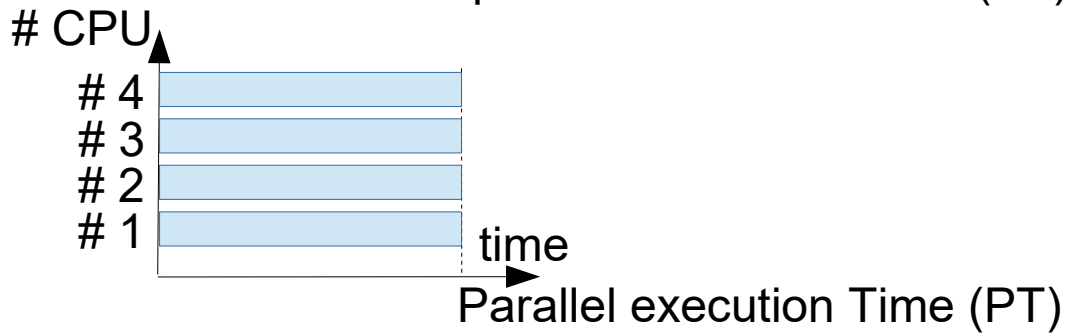
Concurrency and programming

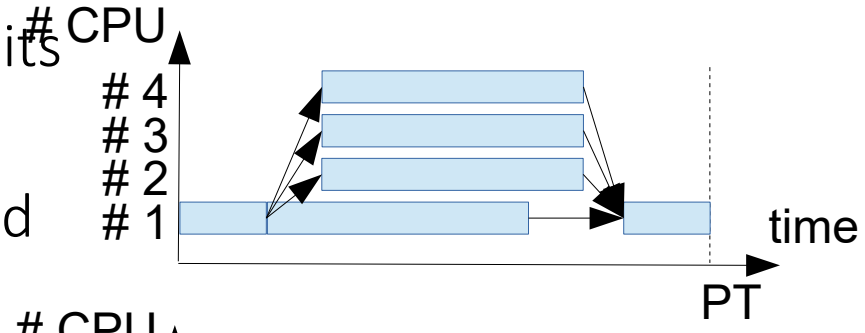
Sequential vs parallel execution

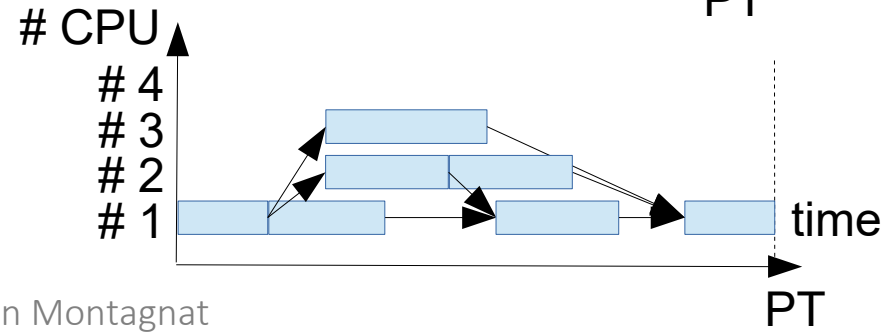
- Sequential execution



- Parallel executions
  - Ideal case
    - Sliceable problem
    - No communications
    - N times faster with N units
  - Freely sliceable problem
    - Number of slices mapped to available resources
    - Communications delay to consider
  - More general case







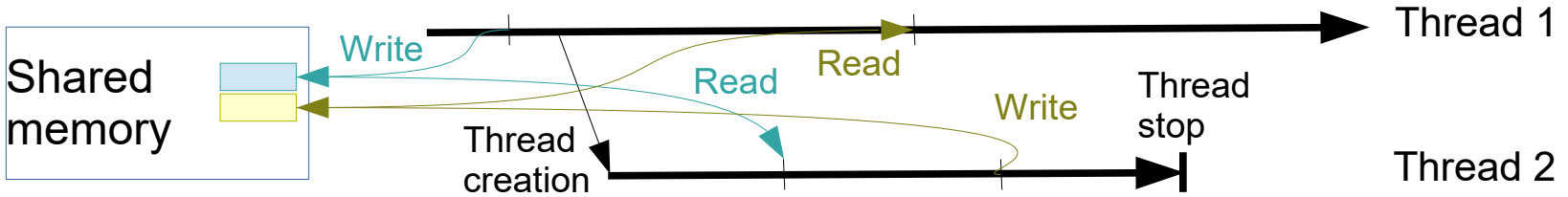
Parallel & distributed computing

- Multiple simultaneous execution threads whit communications
- A continuum from tightly connected computing resources to Internet-wide connection
  - Within a single OS, communication through shared memory
    ⇒ **Parallel computing (shared memory)**
  - Across LAN and WAN, network communication
    ⇒ **Parallel comp. (distributed memory) / Distributed comp.**
  - With minimal communication needs
    - File sharing in a common file system
    - File transfer and copying over the Internet

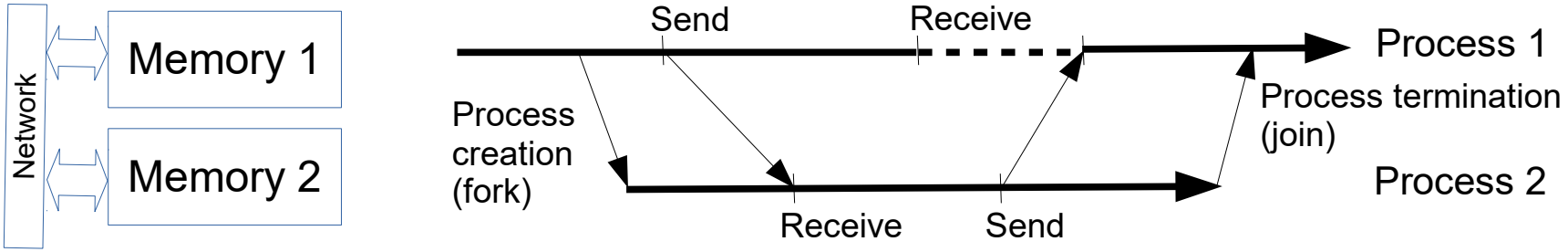
    ⇒ **Distributed computing**
- Trade-off to be found between parallel computation grain and communication cost
  - May apply to regular interconnected desktops as well as Supercomputers

Parallel programming

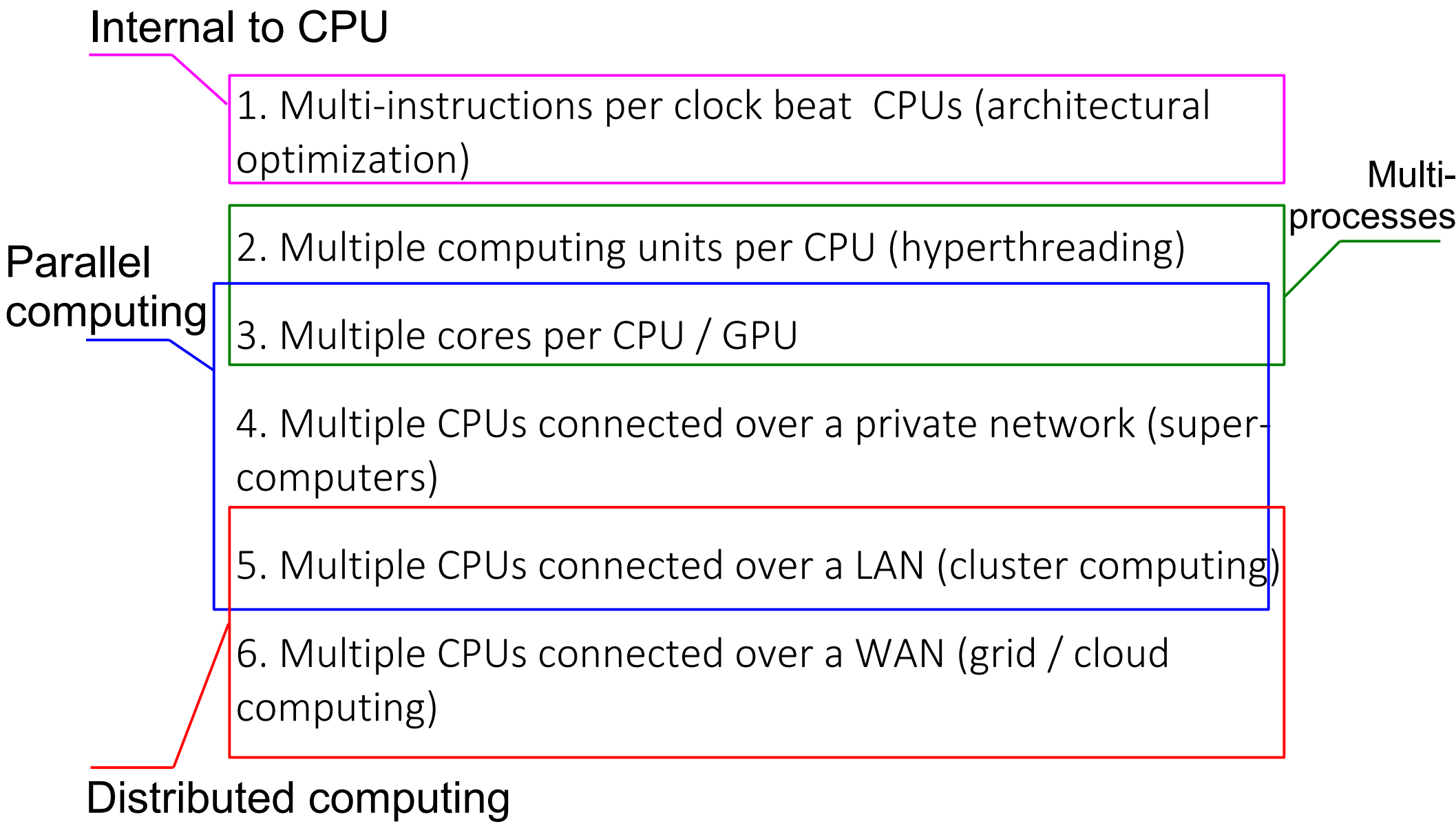
- Multi-thread programming
  - Data exchanges through shared memory
  - Use of system threading / concurrency libraries
  - OpenMP (openmp.org) to ease parallelization of regular code



- Multi-processes programming
  - Shared memory (when possible: e.g. multi-cores)
  - Exchange of messages over the network otherwise (distributed computing requires message passing)
  - MPI (Message Passing Interface) widely used



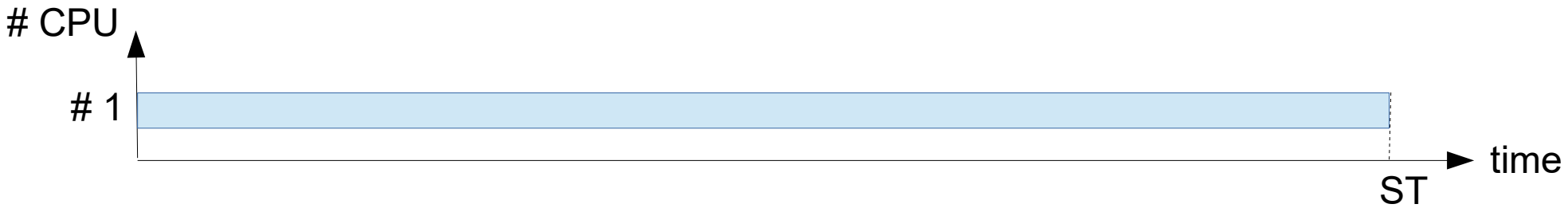
Parallel and distributed computing



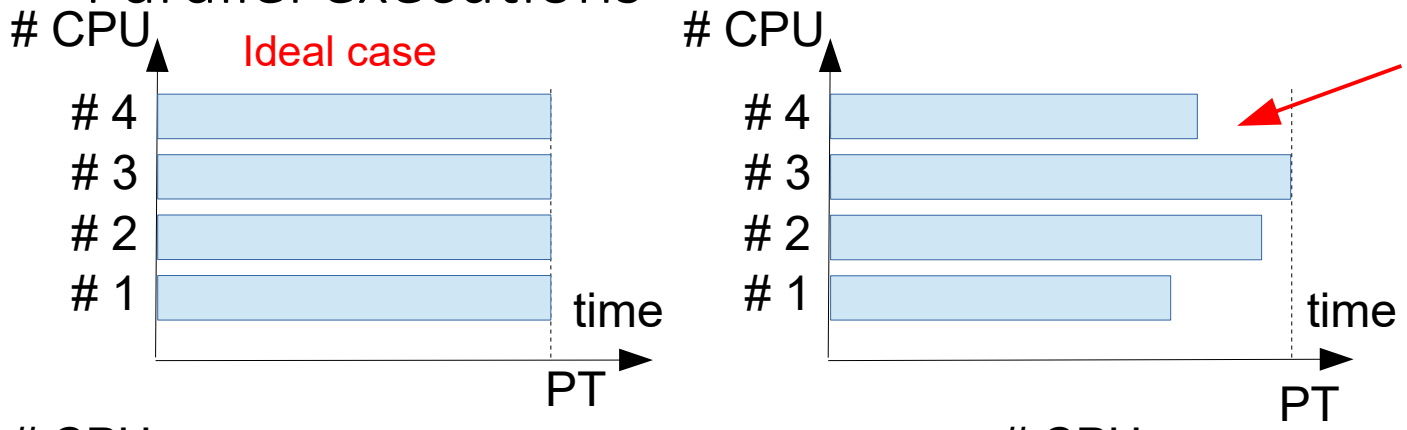
Parallel Computing

Parallel execution performance

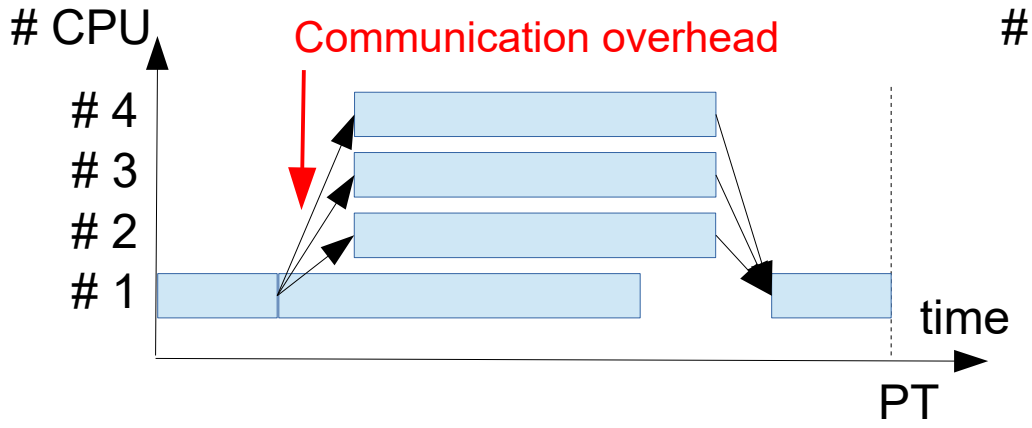
- Sequential execution

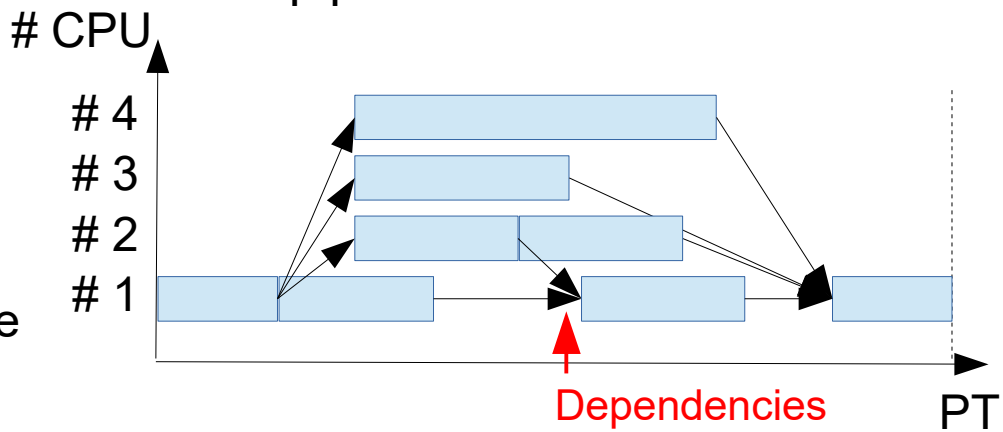


- Parallel executions



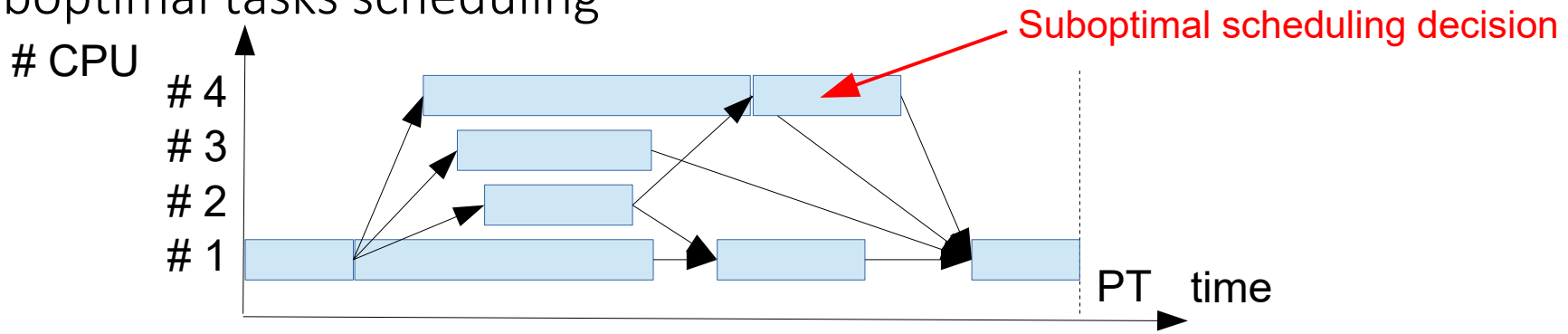
Speed-up:
S(n) = ST/PT(n)





Parallel execution performance

- Other causes of parallel performance loss
  - Heterogeneous computing units or network links
  - Suboptimal tasks scheduling



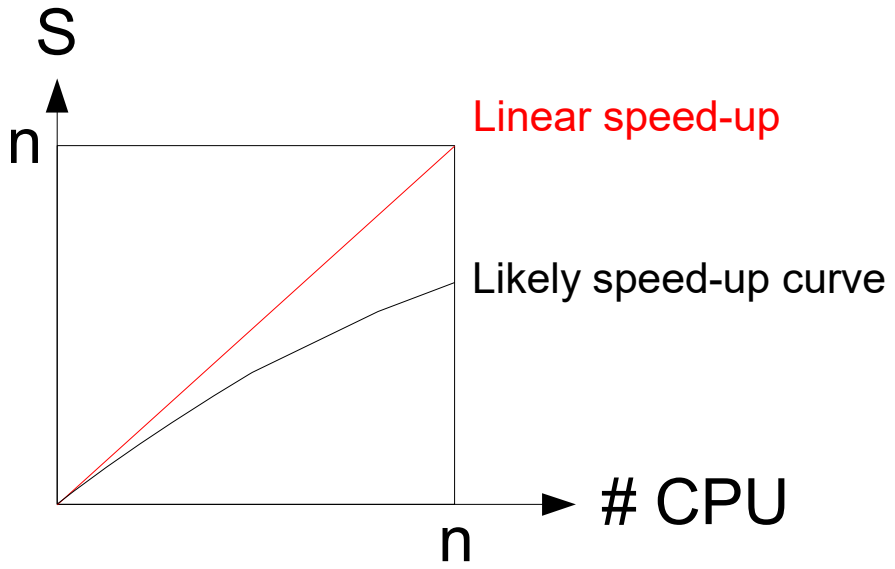
  - Computation unit inter-dependencies (bottlenecks)
  - Security overheads
  - Communication faults...
- Many other concerns related to parallel computation
  - Enabling communications (authorization, firewalls)
  - Scalability issues
  - Deadlocks and concurrency issues
  - Non-dedicated resources sharing impact...

Parallel performance metric

- Speed-up
  - Function of the number of processors (n)
  - Ratio of sequential execution time, ST, over parallel processing time, PT(n):

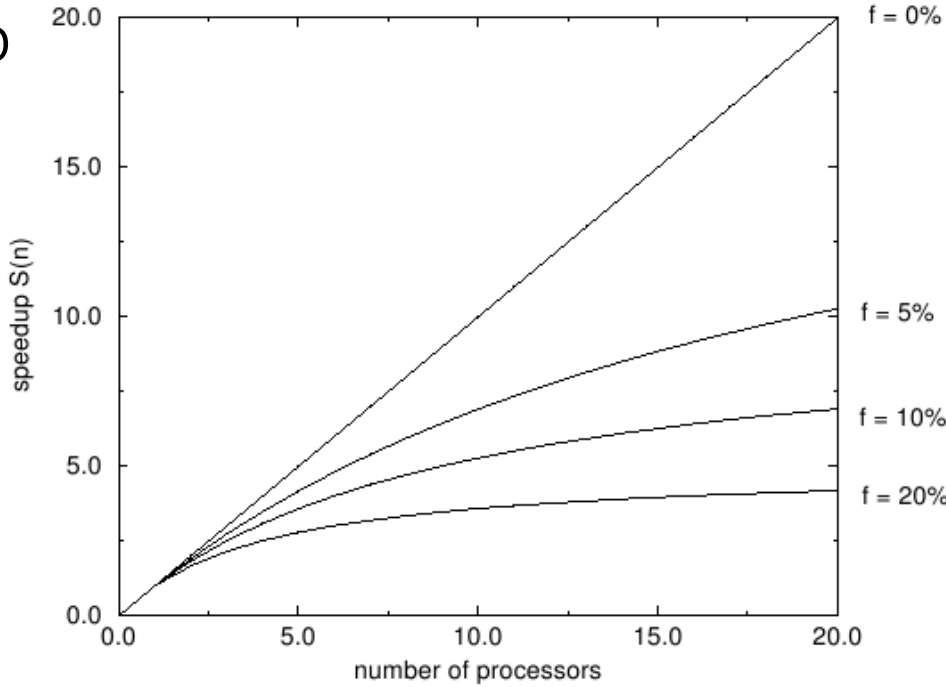
    $S(n) = ST/PT(n)$
  - Efficiency:

    $E(n) = S(n) / n = ST / (n \times PT(n))$
  - Linear (ideal) speed-up: S = n when using n computing units
  - Super-linear speedup : S > n



Amdahl's law for parallel computers

- A code is always made of a sequential part (fraction f) and parallelizable part (1-f)
  - ST is the sequential execution time
  - n is the number of processor
  - Parallel time supposing linear speed-up of parallelizable sections: $PT(n) = f\,ST + (1-f)\frac{ST}{n}$
- Maximum speed-up, assuming no communication overhead
  - $S(n) = \frac{ST}{PT(n)} = \frac{n}{1 + f\,(n-1)} \xrightarrow{n \to \infty} \frac{1}{f}$
- Conclusions
  - Asymptotic speed-up: $\frac{1}{f}$
  - Small reductions in sequential overhead and communication overheads can make a huge difference in throughput

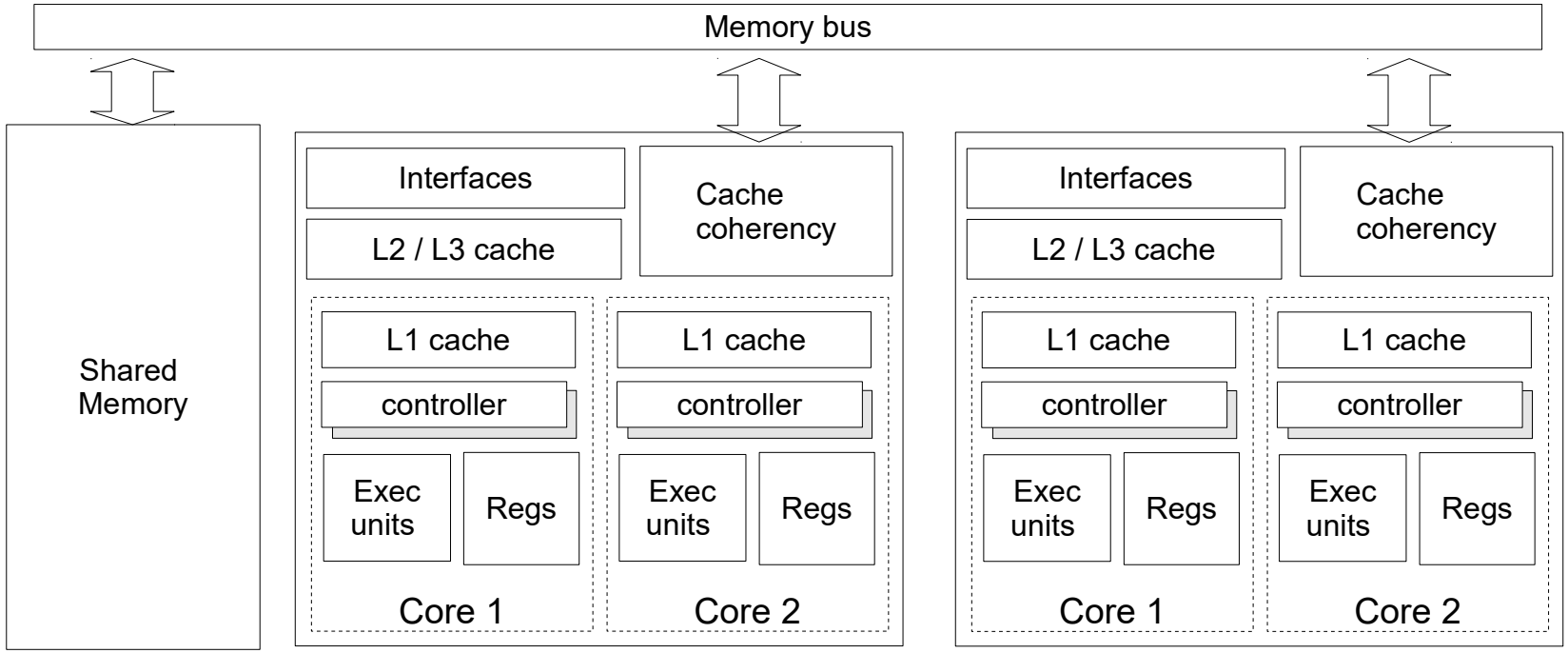


Parallel computing: Supercomputers

- Supercomputers are intensive parallel computing machines
  - Number of cores up to 3 000 000 (as of 2015)
- Supercomputers architectures
  - “regular” CPUs
  - GPUs and accelerators
  - coupled with very efficient interconnecting network
- A super computer is a computing cluster with highly efficient internal communication network

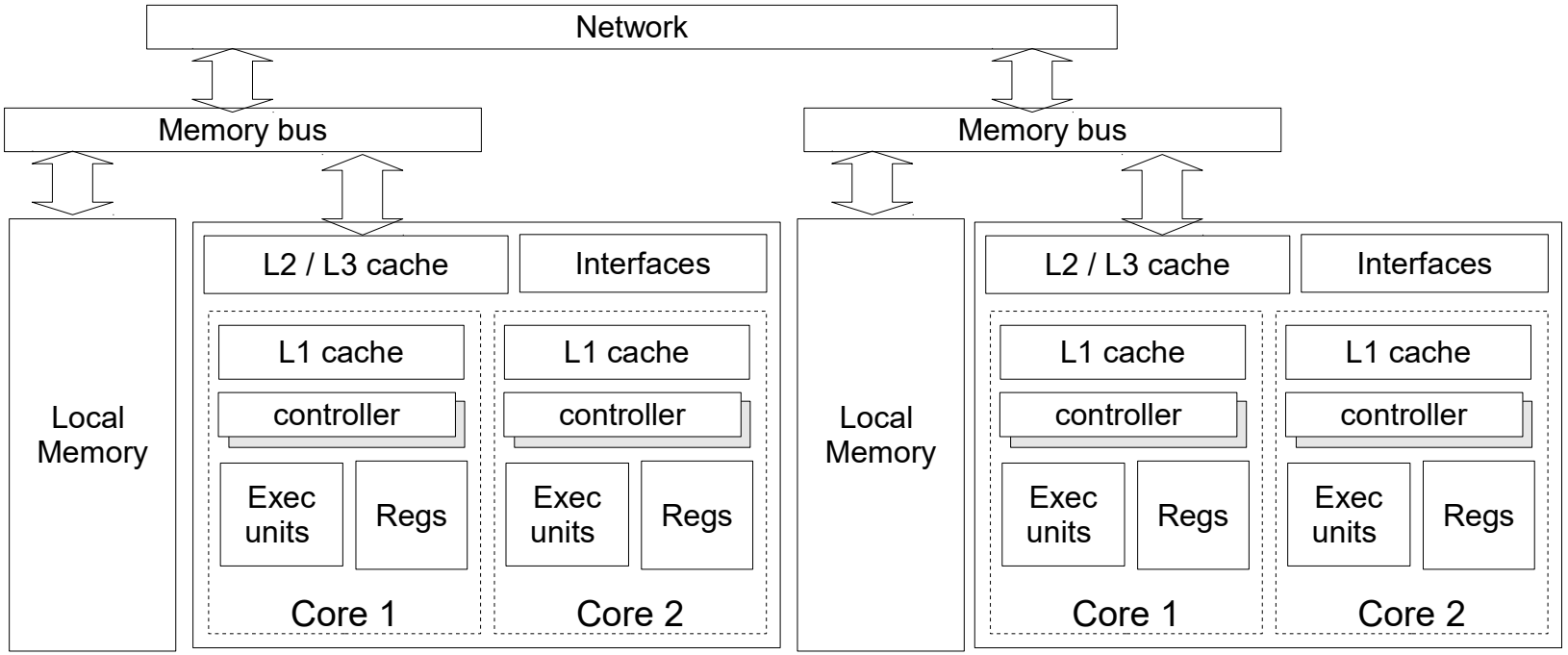
Data exchange strategies

- Shared (global) memory
  - Communications through memory access
  - Concurrent R/W operations possible
  - High cache coherency cost
  - Memory bus bottleneck



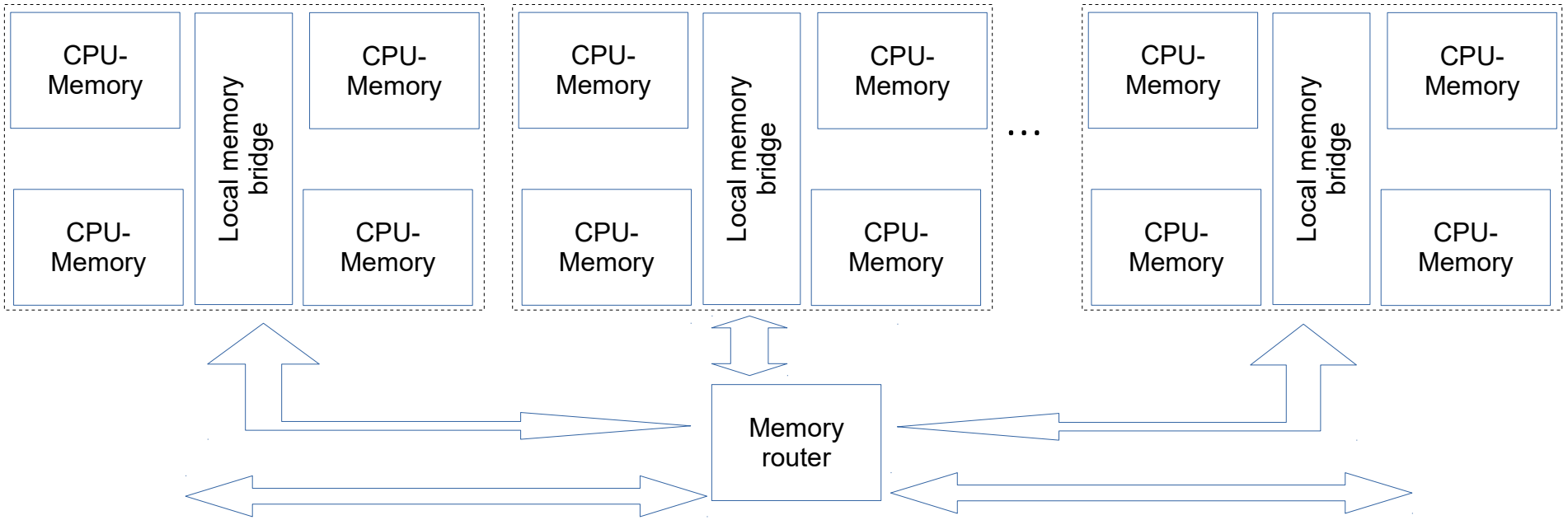
Data exchange strategies

- Distributed (local) memory
  - Explicit communications (message passing)
  - Highest communication cost
  - Concurrency and coherency managed by message library



Data exchange strategies

- Hybrid local / global memory
  - Hierarchical memory: NUMA (Non-Uniform Mem Access)
  - Cost vs. ease of use trade-off

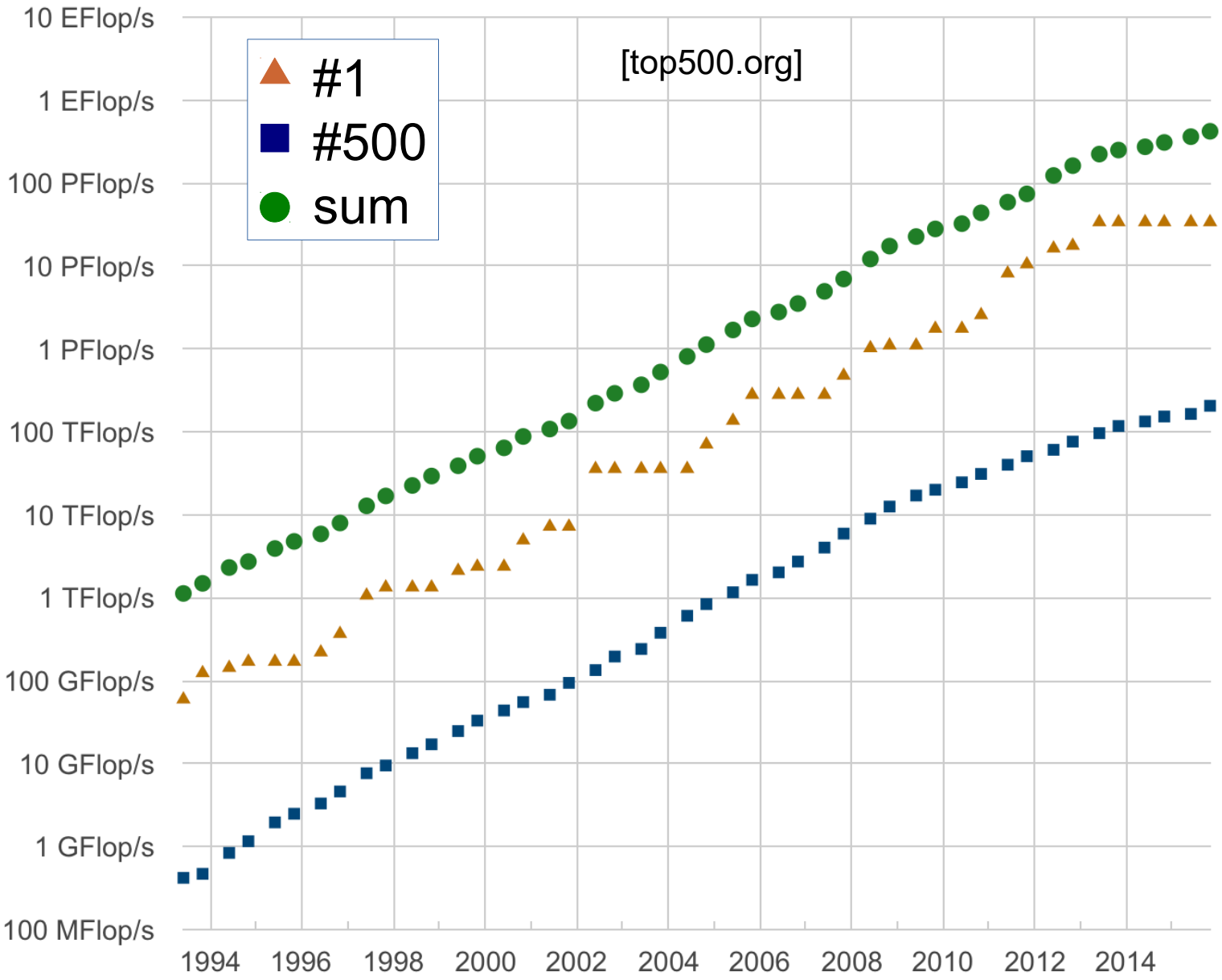


Supercomputers performance

- Number of operations per second
  - Expressed in flops (number of floating operations per second), and benchmarked (e.g. SPECint2006)
  - Peak performance (n x individual unit performance) is a theoretical upper limit
  - Real performance is very dependent on algorithms (see Amdahl's law)
- Top500 benchmark
  - SuperComputer benchmark based on Linpack benchmark (linear algebra)
  - Renewed every 6 months. See top500.org .

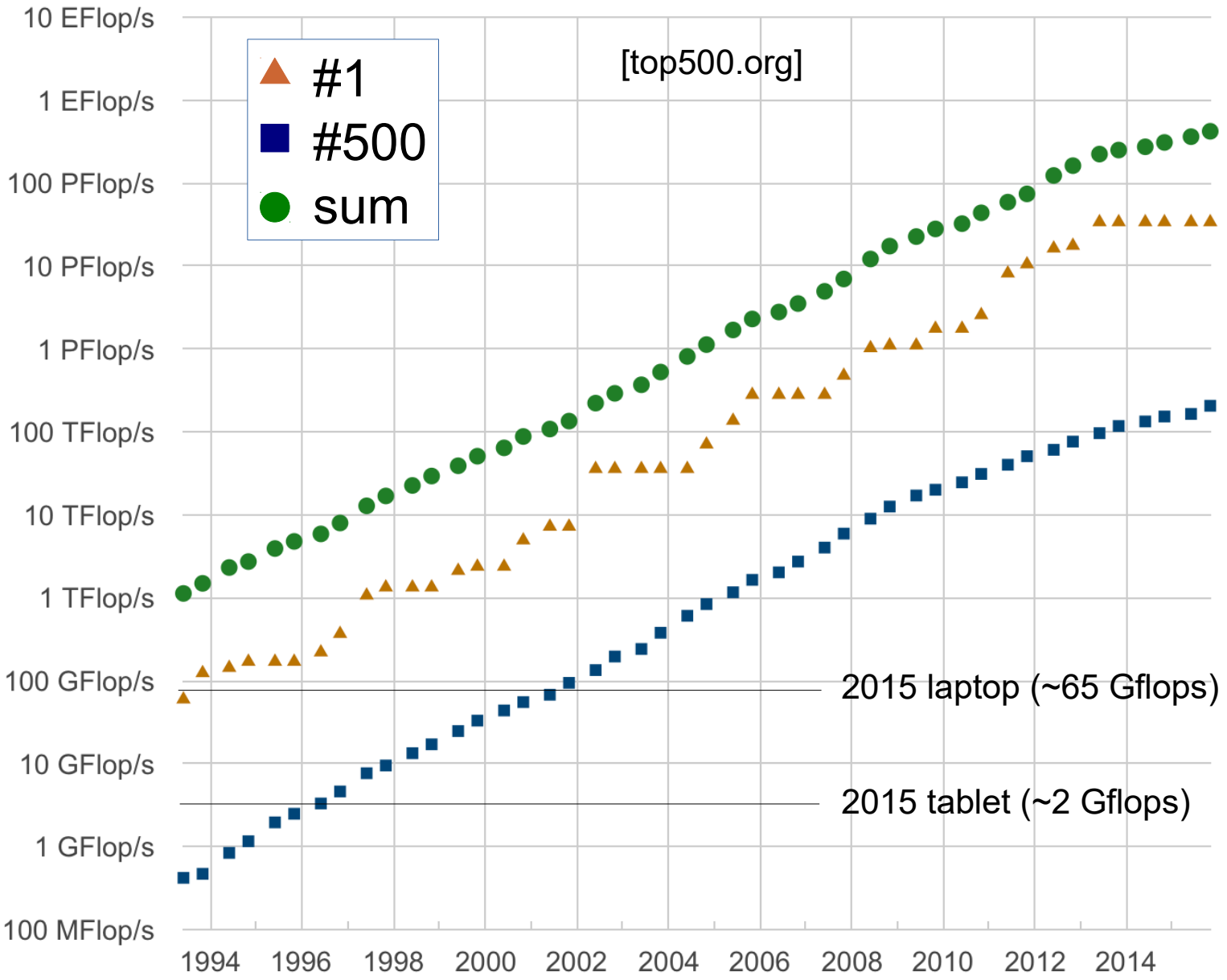
Top500 supercomputers (nov. 2015)

- Linpack benchmark performance
- Scale
  - Giga = 10^9
  - Tera = 10^{12}
  - Peta = 10^{15}
  - Exa = 10^{18}
- Since 1993, performance x10 every 3 years

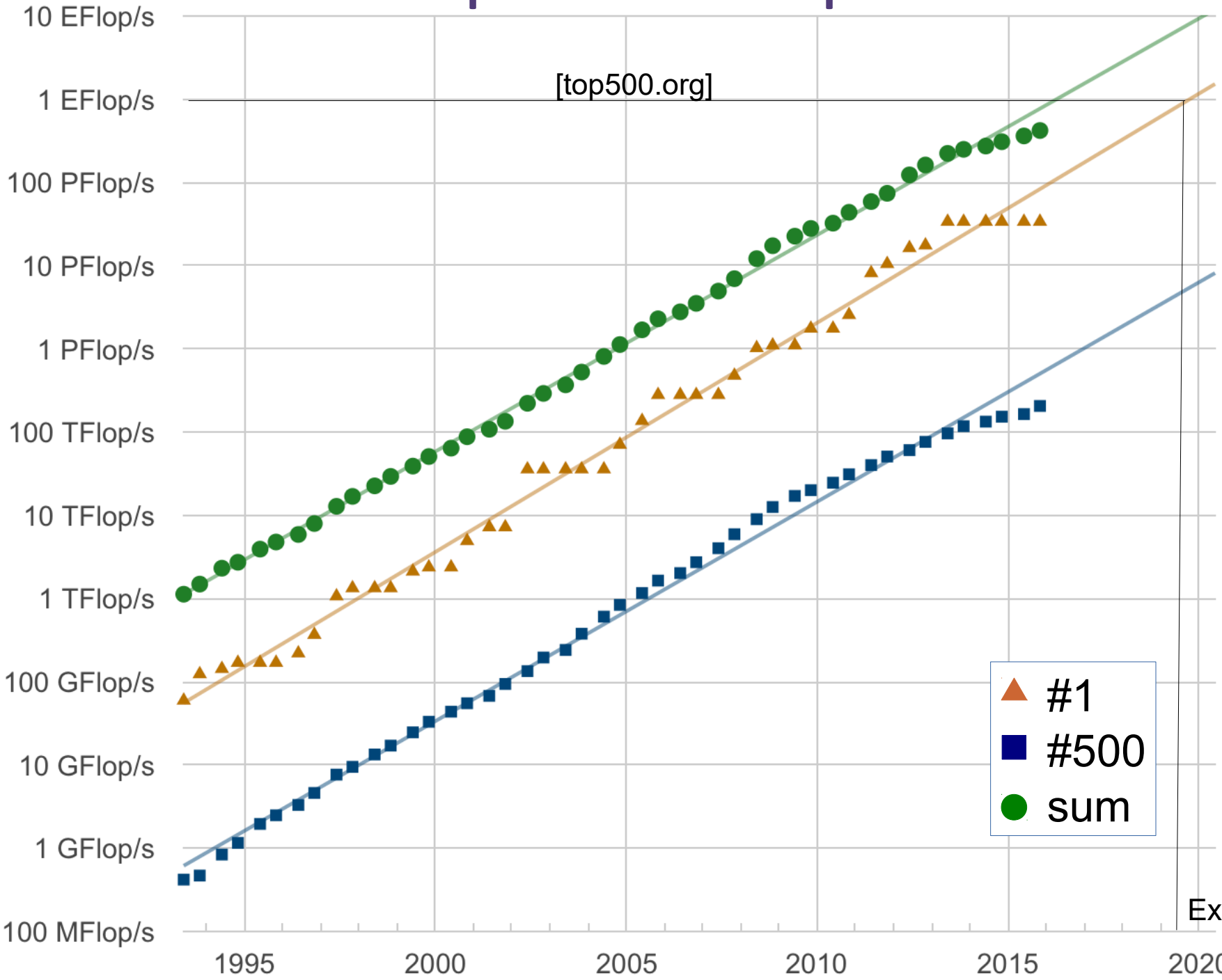


Top500 supercomputers (nov. 2015)

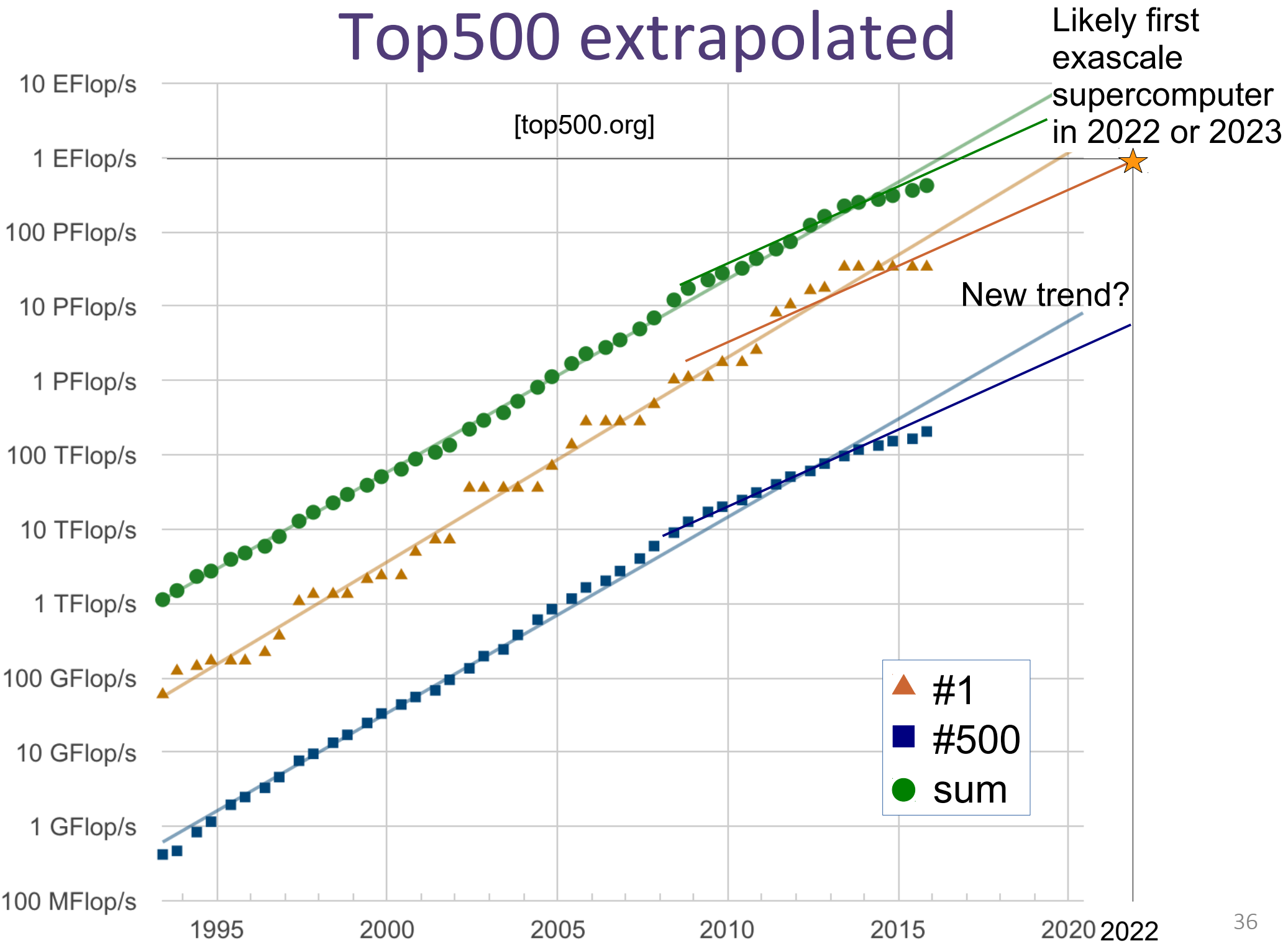
- Linpack benchmark performance
- Scale
  - Giga = 10^9
  - Tera = 10^{12}
  - Peta = 10^{15}
  - Exa = 10^{18}
- Since 1993, performance x10 every 3 years



Top500 extrapolated

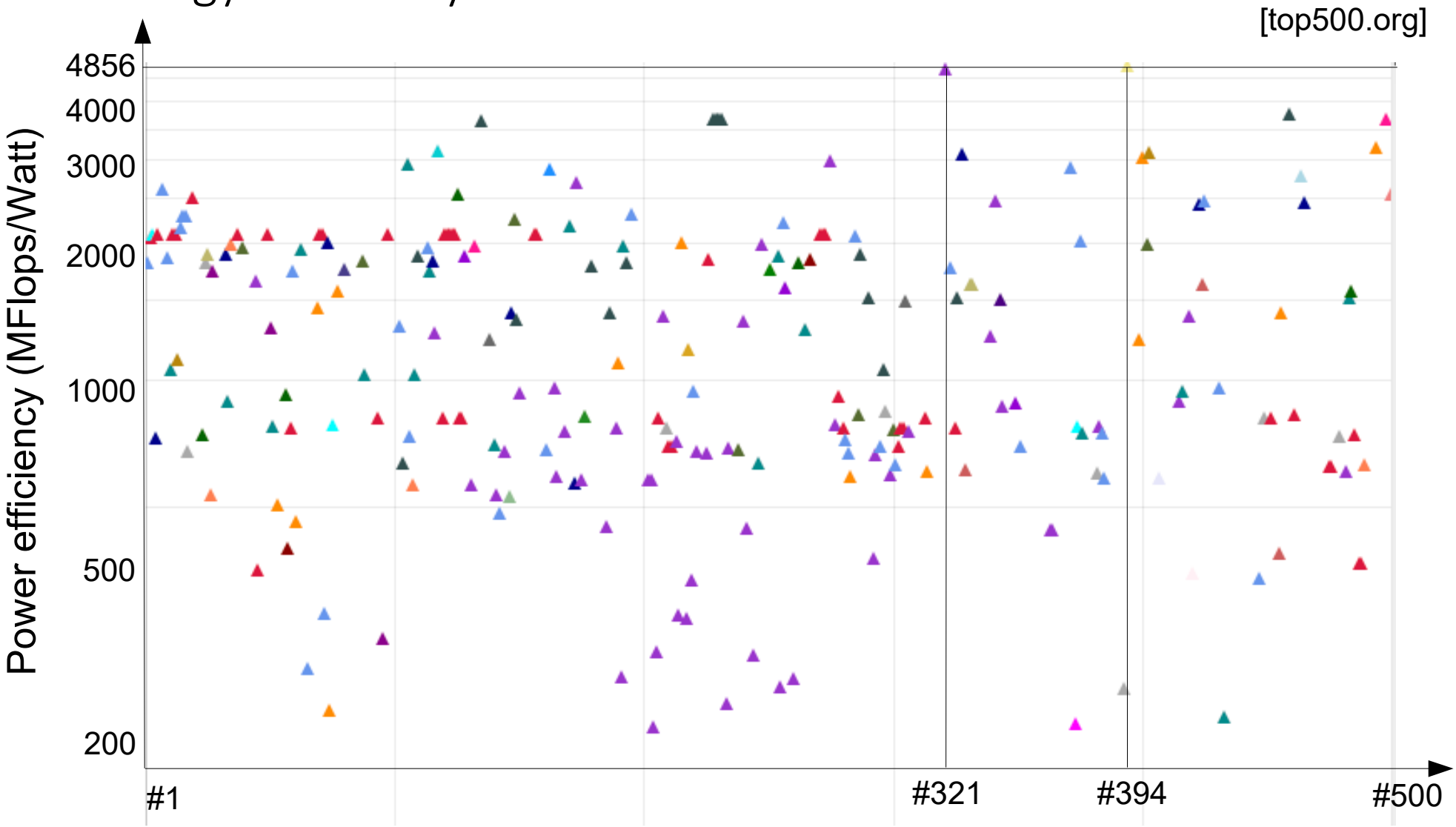


Top500 extrapolated



Efficiency and reliability factors

- Energy efficiency matters



- Mean time between failure in #1 supercomputer ~8 hours

Distributed Computing

Distributed computing

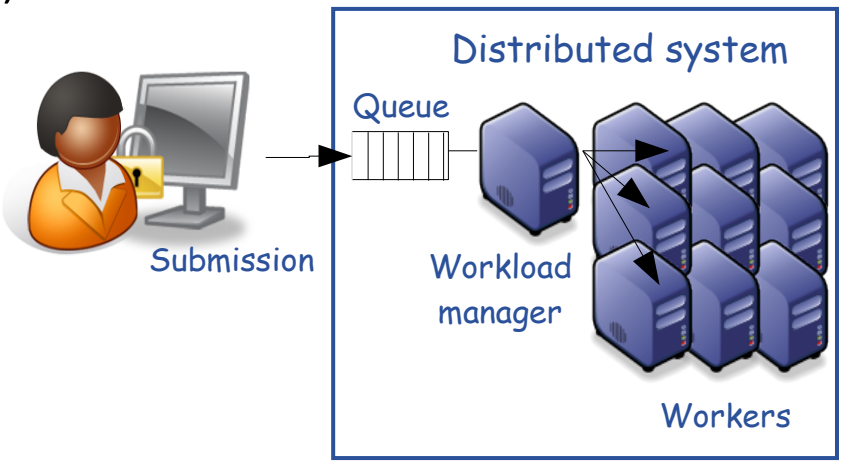
- Distributed computing refers to independent processes executed concurrently on remote resources
  - With no or hardly any exchange between processes
  - Coarse-grained kind of parallelism
- Motivations
  - Cheaper than supercomputers
  - Resources sharing among multiple users
    - Computing and data resources, data sets...
  - Many scales for distributed computing systems
- Different distributed computing architectures
  - Computing center (aka Data Center)
  - Multiple resources distributed over the local network
  - Many resources distributed over the Internet

Distributed computing requirements

- Computation distribution strategy (programming model)
  - Reduce inter-dependencies between processes
- Resources allocation strategy
  - Scheduling algorithms to optimize resources usage
    - NP-completeness to achieve optimality
    - Many heuristics, including on-the-fly best effort allocation applying to batch systems
  - Advance reservation, for critical tasks
    - Possibly with resources pre-emption
- Remote invocation method
  - Through remote command lines or invocation methods
  - For homogeneous or heterogeneous sets of computers
- At the larger scale, more difficulties arise
  - working across many institutions, WAN dependencies, heterogeneity...

Batch systems

- Batch computing
  - Shared resources
  - Single entry point (batch manager)
  - Tasks queue with priorities
  - Ever-loaded, multi-user system
- Batch speed-up
  - Workload queuing time: QT
  - User waiting time: PT(n) + QT
  - Real speed-up: ST / (PT(n) + QT)



- Focus on Throughput (number of tasks per unit of time)
  - **H**igh **P**erformance **C**omputing: fast response time
  - **H**igh **T**hroughput **C**omputing: many results produced (potentially with large latency)

Distributed systems strength

- Widely available
  - Low cost
  - In some case use of idle resources over the Internet
- Coarse-grained parallelism
  - Data-intensive scientific applications
  - Long experiment campaigns (HTC)
- Distributed data management
  - Large data sets federation
  - Rare/sparse data
  - Remote data acquisition devices
- Cost reduction through scale effect
  - High peak computing power
  - But all users cannot always use all resources → **sharing**

Distributed computers limitations

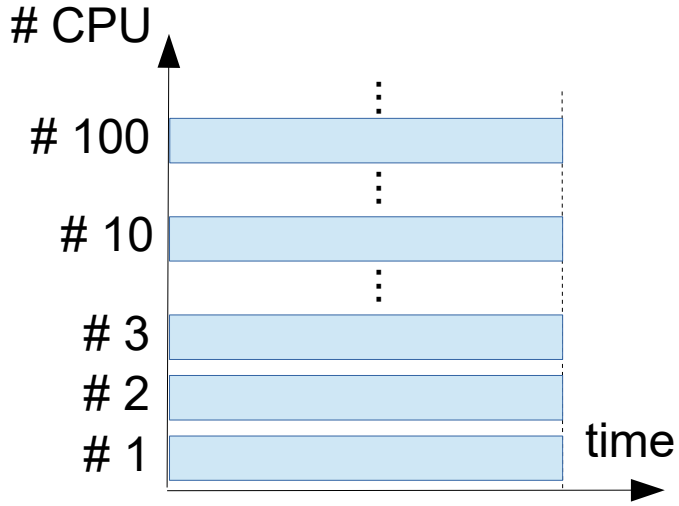
- Single computing unit capabilities
  - Powerful, yet limited capabilities of each host
  - Using general purpose network connection
- No magic in application parallelization
  - Automatic exploitation of “explicit” parallelism only
  - Require manual parallelization in complex cases
- Supercomputers are costly but
  - lowest communications overhead, smaller grain parallelism
  - some problems are not efficiently executed on distributed computing infrastructures and do require supercomputers
- Parallelism is always a trade-off between granularity and communication overhead

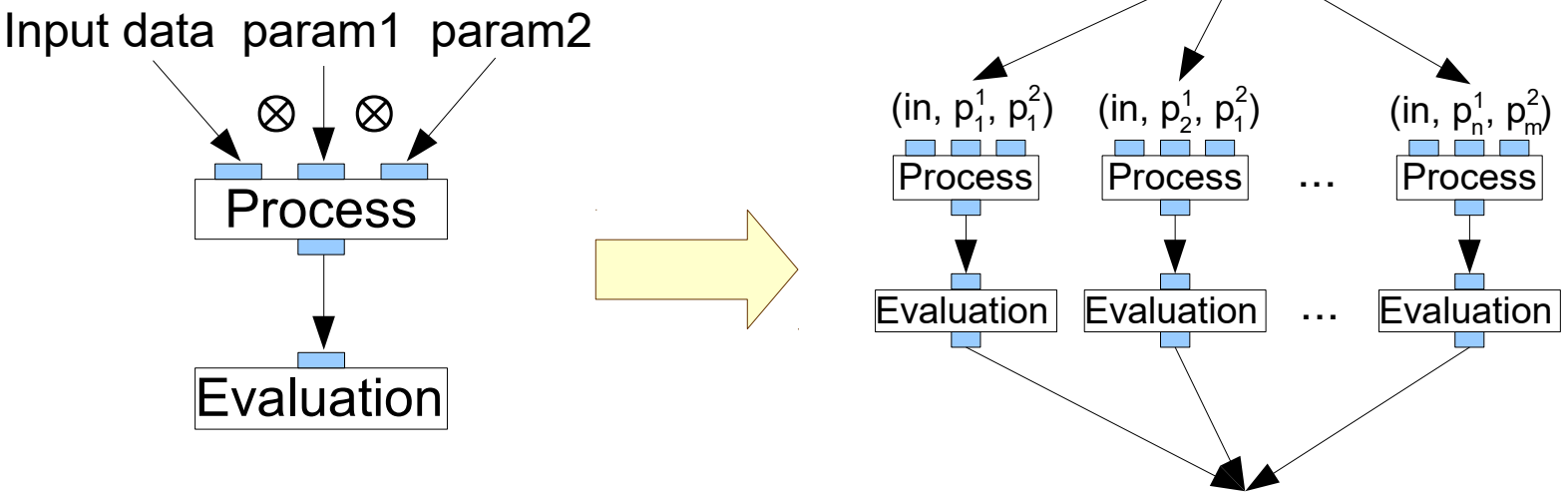
Distributed computers limitations

- Clusters of standard computers are cheap…
  - but require system administration
- Several trends to address complexity
  - Autonomic computing
    - Self-resilient, Self-adaptive, Self-corrective systems
  - Virtualization
    - More homogeneous system pools management
    - Ease code deployment and migration
  - Externalization
    - Services hosting by service providers
    - Use of cloud infrastructures

Distributed application models

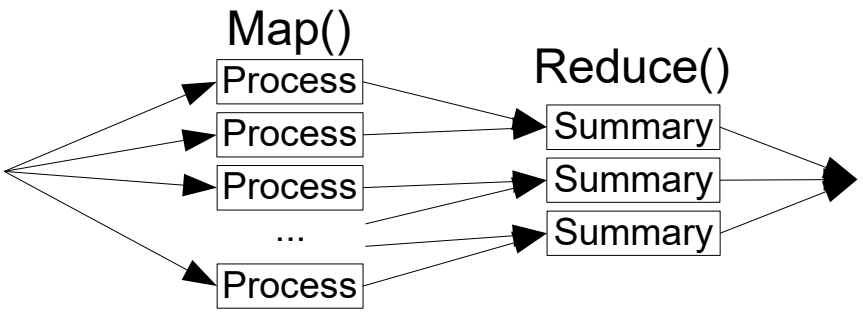
- Embarrassingly parallel applications
  - Embarrassing size
    - Scalability is the challenge
  - Trivial parallelisation
- Parameters Sweep
  - Explore process parameters space
  - Many optimization problems
  - Combinatorial



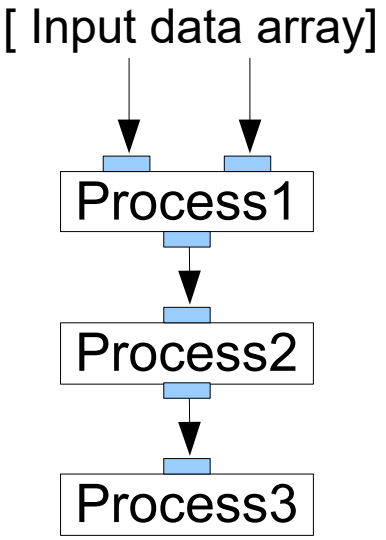


Distributed application models

- Map-Reduce framework
  - Two steps simple parallelization framework
    - Map() function decomposes computation
    - Reduce() function combines results
  - Backed-up by fault-tolerant and scalable support tools
    - e.g. Hadoop

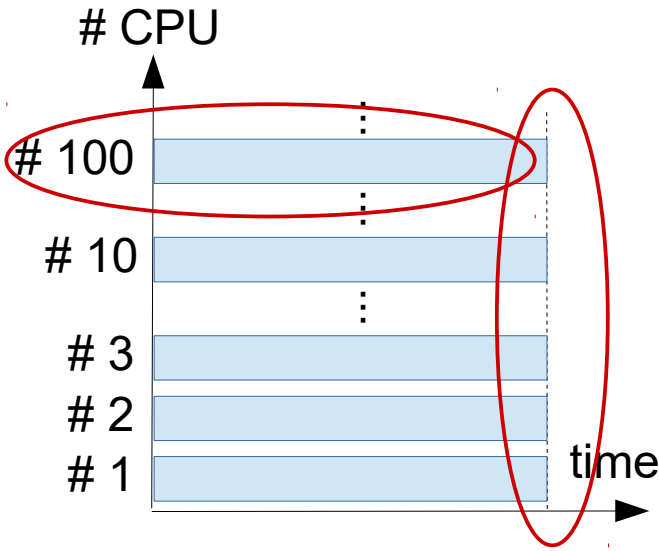


- SPMD (Single Processing, Multiple Data)
  - Exploit large data sets parallelism
  - Independent computations on different data items
  - Simple process: embarassingly parallel
  - Complex process: pipelines / workflows



Massive parallelism

- Challenges
  - Scale, address heterogeneity, synchronize tasks
- Scale
  - Gently handle increasing number of tasks
  - Protect system from overload
  - Manage workload through time
  - Cope with components reliability issues…
- Heterogeneity
  - Computing resources, networks…
- Synchronize tasks
  - Allocate resources
  - Share workload
  - Deal with balancing issues and runtime evolution



Large-scale computing systems

Large-scale

- Never enough computing power
  - Reality is complex, computerized models are complex
  - Increasingly larger scientific projects (international scale)
  - Brute force is used when analytical resolution fails
  - Brute force is tempting (easy to set up)
- Computation / communication ratio
  - Network performance has increased faster than computing units performance
  - Wide-area network reliability is improving
- Increasing amount of computing units
  - Standard computing units are affordable
  - > 1 billion units connected to the Internet today

Other motivations for distributed computing

- Exploiting unused resources
  - Resources are used part-time only
  - Sharing hardware resources
  - Exploiting lost cycles

- From workstations to clusters to grids to clouds
  - Multi-tasks workstation: sharing CPU time
  - Clusters: sharing local computers among users
  - Grids: sharing global resources among users
  - Clouds: renting unused resources to improve efficiency

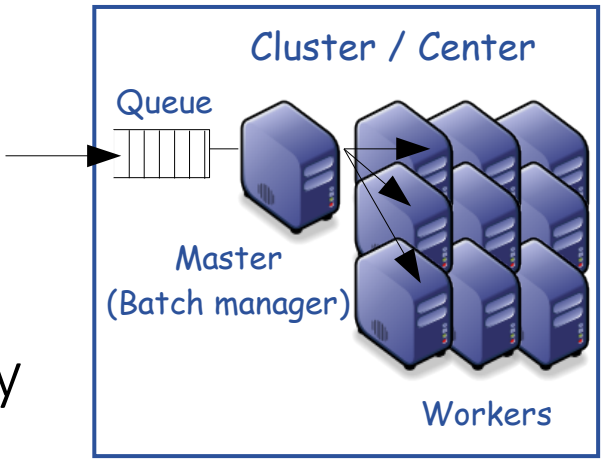
Many kinds of distributed computing infrastructures

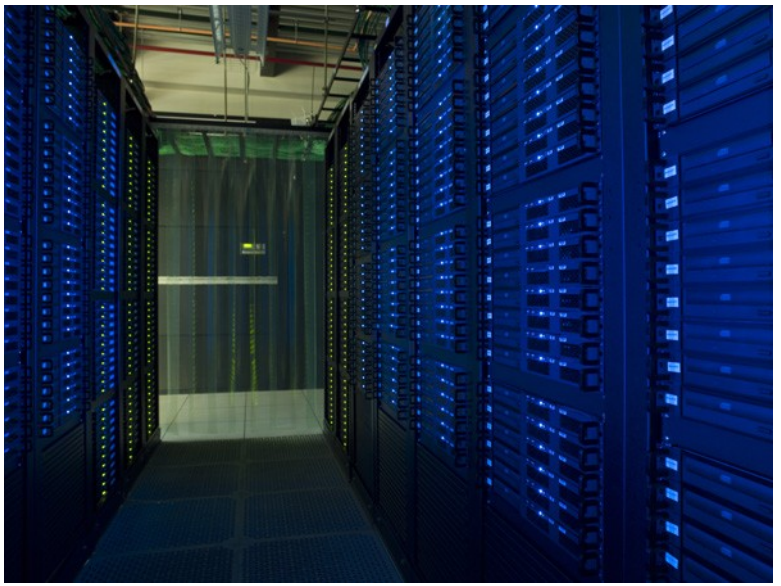
- Clusters
  - Usually internal to an organization
  - Sometimes large computing centers shared
- Internet grids
  - Desktop grids: use of individual's computing units
  - Multi-computing center grids
- Clouds
  - Private clouds: internal to an organization
    - Extension of cluster computing, making use of virtualization technologies
  - Public clouds
    - Delivering public service over the Internet
  - Specialized clouds
    - Mail servicing, data storage, etc.

Computing clusters

Clusters / Data Centres

- Mutualize and share computing resources
  - From a dozen to 100,000s CPU cores
  - Batch-oriented
    - Independent computing tasks
    - Submission through a gateway (batch manager)
    - Queue request and process according priority policies
    - Deal with regular executables (command line code)
  - With shared storage facilities
  - Some parallel codes (e.g. MPI)





Production systems

- Infrastructure is tightly integrated in all scientific and social applications
  - Becomes a foundation layer, just like networks
  - Data acquisition and archiving, data warehouses
  - Computation services
  - Deliver high quality specific services targeted to a wide audience
  - Results archiving and dissemination

- Production system
  - Production quality needed to build large-scale scientific or industrial applications
  - A 24/7 system, highly available, reliable
  - Requires fair share (accounting and billing)

Production systems

- Maintenance requirements
  - Hardware and software updates
    - Keep the infrastructure in a coherent status
  - Hardware renewal
    - Defunct hardware replacement and upgrades
- Continuous operations
  - Administration of large resource sets
  - Local and Global administration policies
  - (Distributed) problems identification and fixing
  - Cross-areas authentication and authorization schema
  - Local and Global accounts management
  - Operations are very costly – reducing operation costs is a challenge in large scale systems

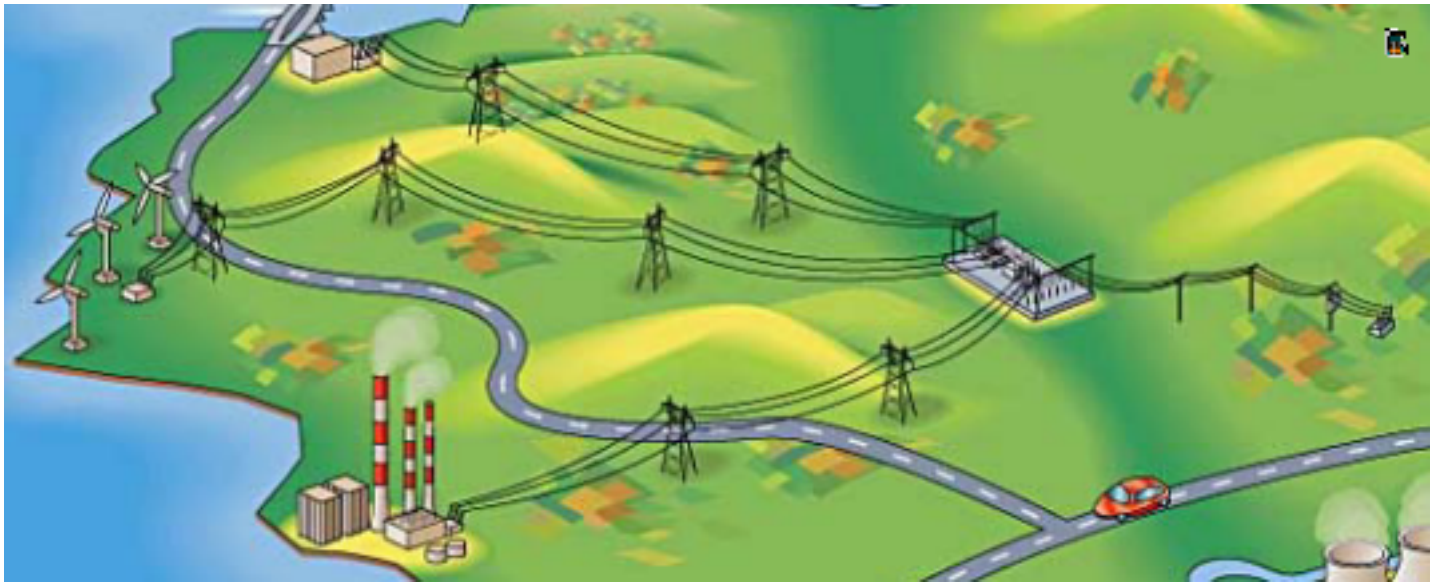
Enabling distributed groups of users

- Infrastructure is shared by multiple users
  - Distributed users, across many institutions
  - Group of users federated by a common application area, across administrative boundaries, aka **Virtual Organizations** (VO)
- Fostering scientific communities
  - International, multi-disciplinary research infrastructures
- A VO defines the authorization unit
  - Access control to resources (computers, data...)
- VO-specific functions
  - Shared resources allocation
  - VO users administration
  - Community-specific software installation
  - Application areas identification and support unit

Computing Grids

Grids

- The myth: providing **unlimited** computing power by letting the user **transparently** access to **infinite** Internet resources.

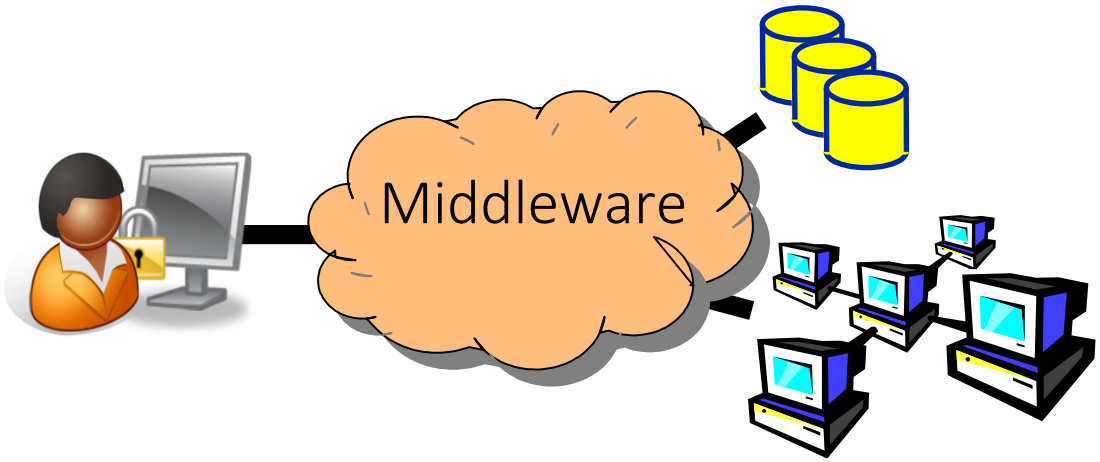


The infamous electrical network analogy

- Early grid adopters
  - Data storage: napster
  - Computation: SETI@home
  - Information: web

Incentives behind grids

- Assembling resources
  - Storage
  - Computing
  - Network
  - Software
  - Data source...

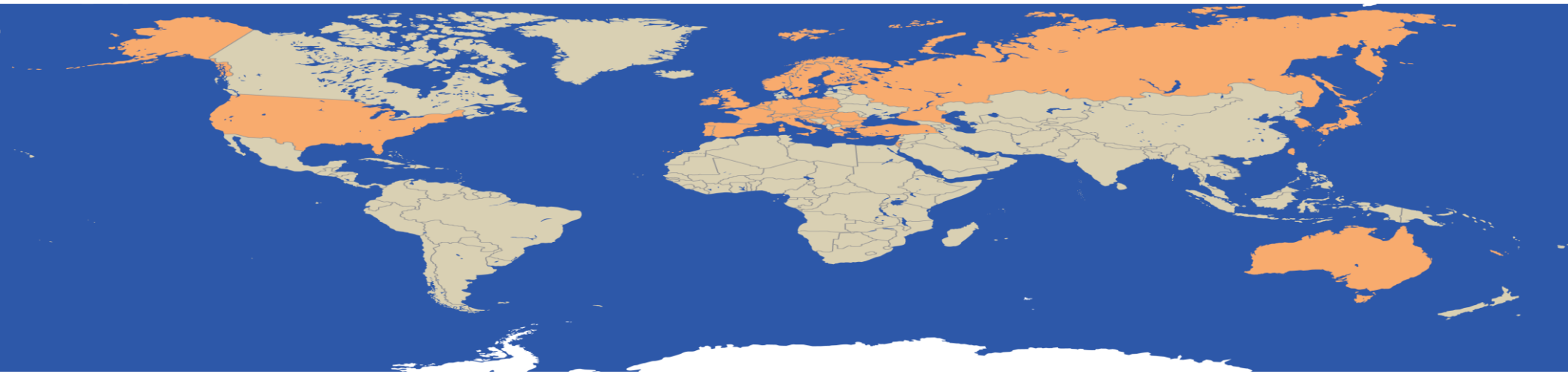


- Federating users
  - Large scale user communities (Virtual Organizations)
  - Ease exchanges
- Pushing standards
  - Communication protocols
  - Data representation and formats
  - Computation control languages
- The real potential is in sharing resources, data, and knowledge.

Grid/Cloud infrastructures

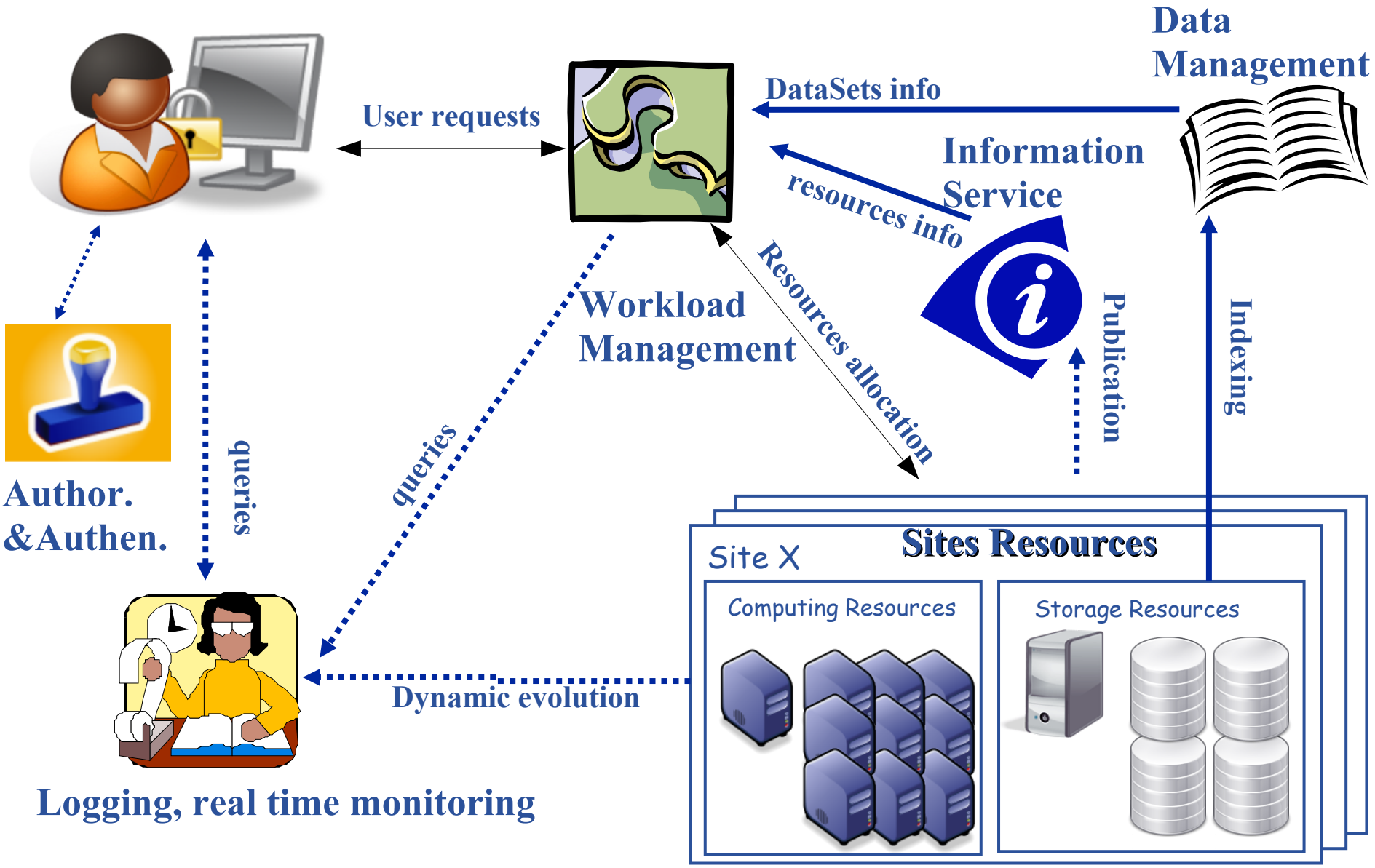
- A few large production grids
  - EGI (EU, http://www.egi.eu)
  - OSG (USA, http://www.opensciencegrid.org/)
  - TeraGrid (USA, http://www.teragrid.org)
  - NAREGI (Japan, http://www.naregi.org)
- A couple of smaller experimental grids
  - Grid5000 (France, http://www.grid5000.org)
  - US Future Grid (https://portal.futuregrid.org)

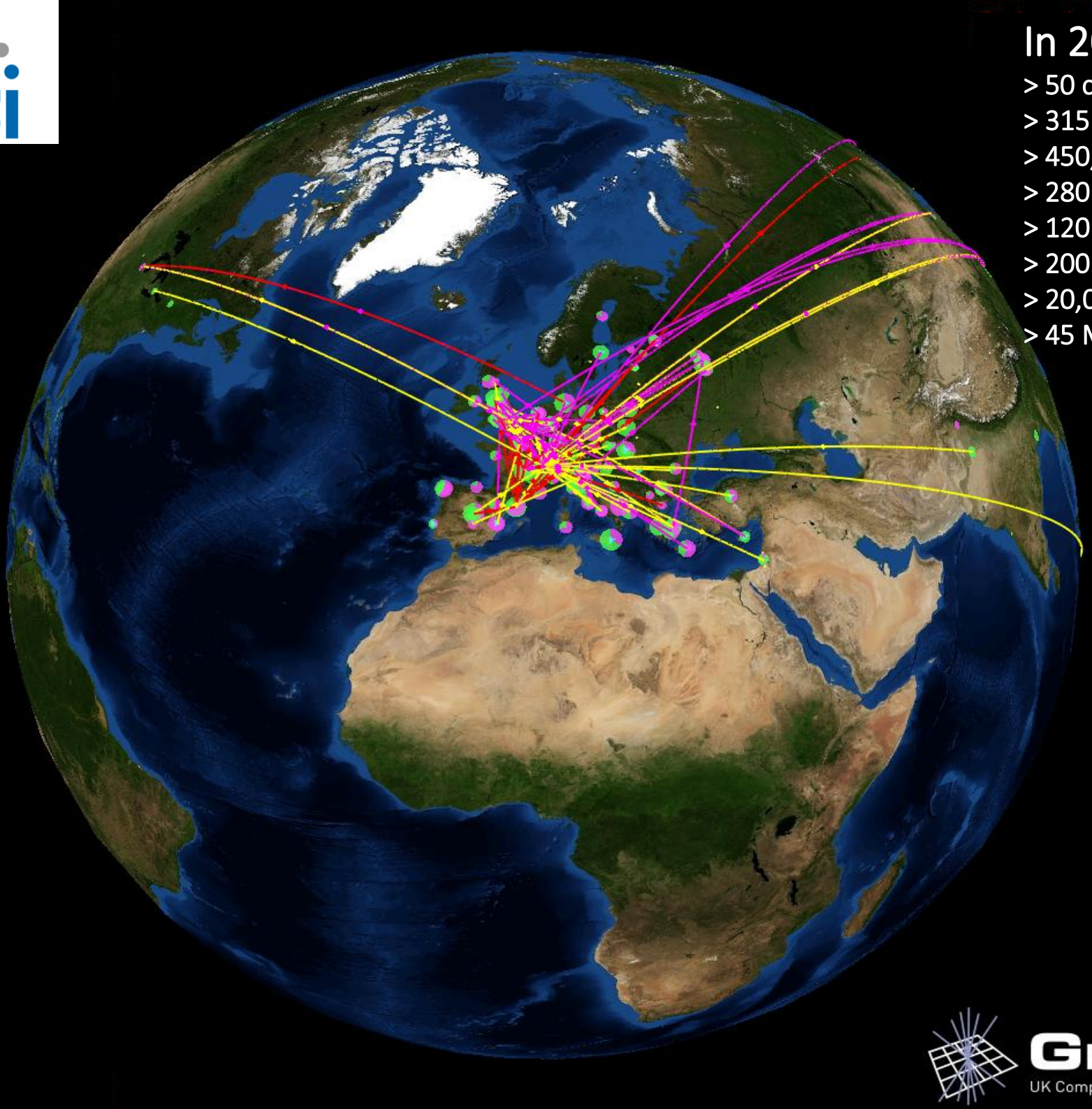
European Grid Infrastructure



- Production infrastructure
  - Operate a large-scale, **production quality** grid infrastructure for e-Science
  - This is **24-7-365** activity
- EMI High Throughput Computing middleware
  - Mutualize computing centres resources

EMI middleware at a glance





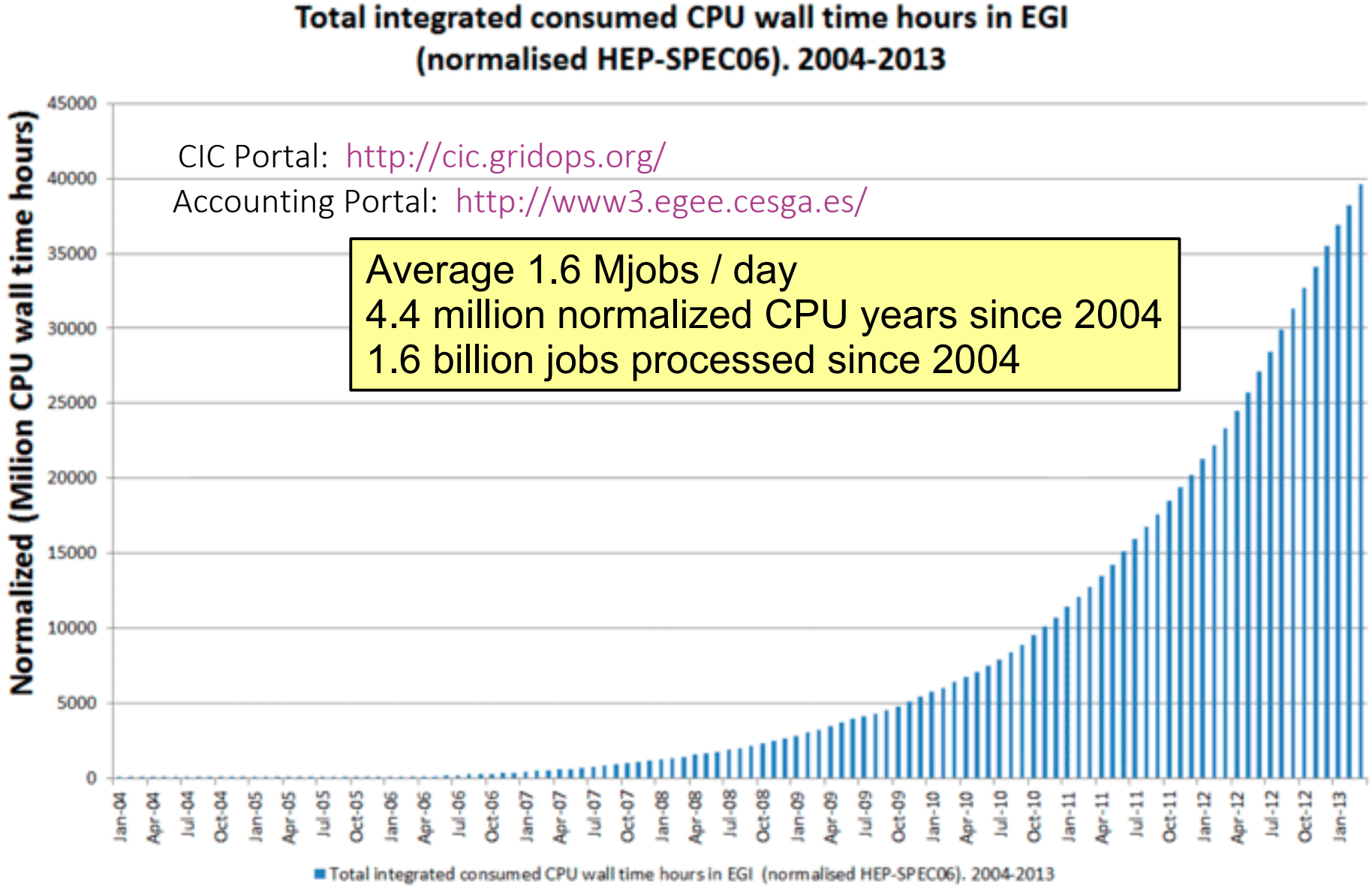
In 2015:

- > 50 countries
- > 315 sites
- > 450,000 CPU cores
- > 280 PB disks
- > 120 PB tapes
- > 200 VOs
- > 20,000 users
- > 45 Mjobs / month

GridPP
UK Computing for Particle Physics

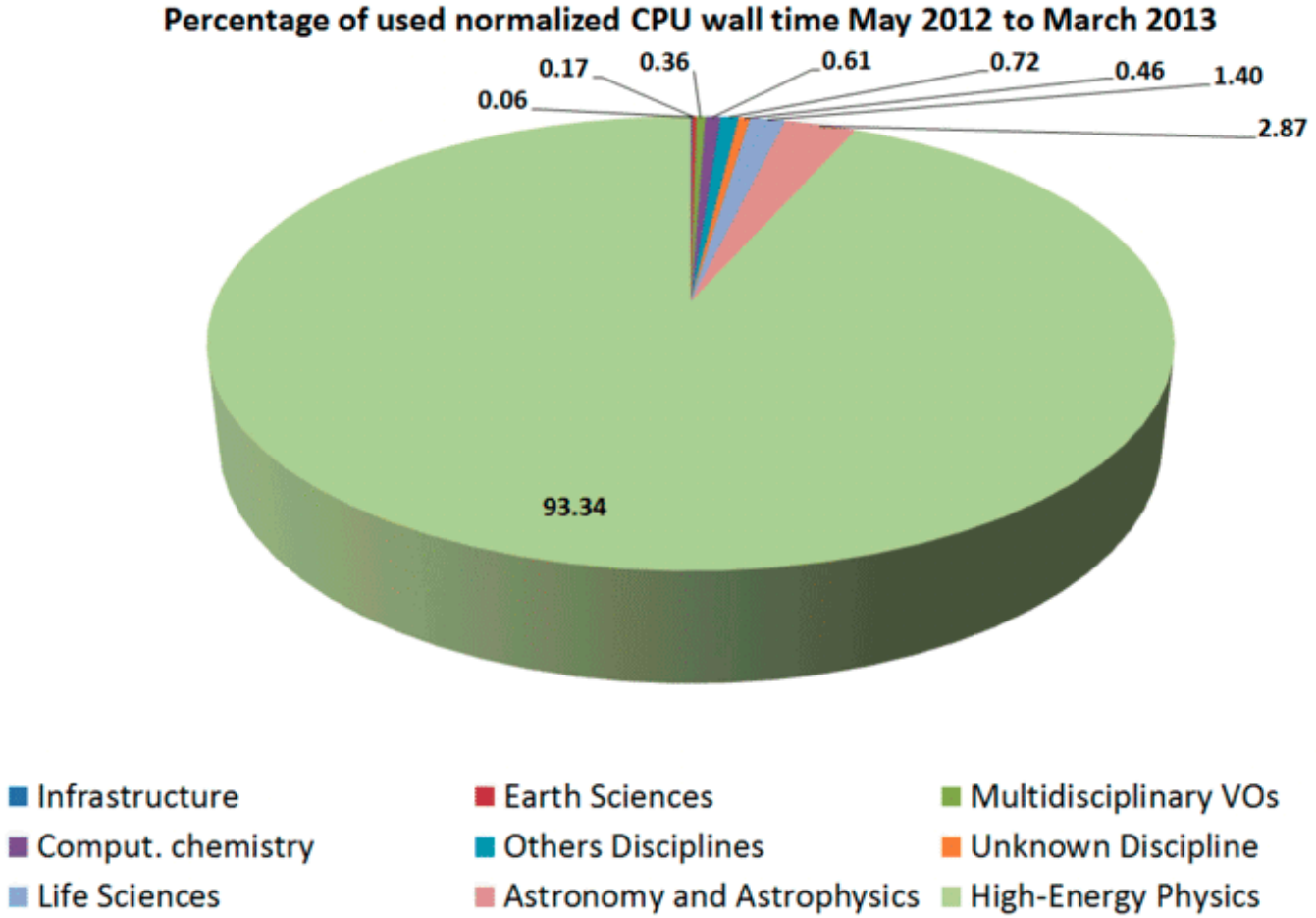
Capacity

- Regular increase of infrastructure growth since 2004

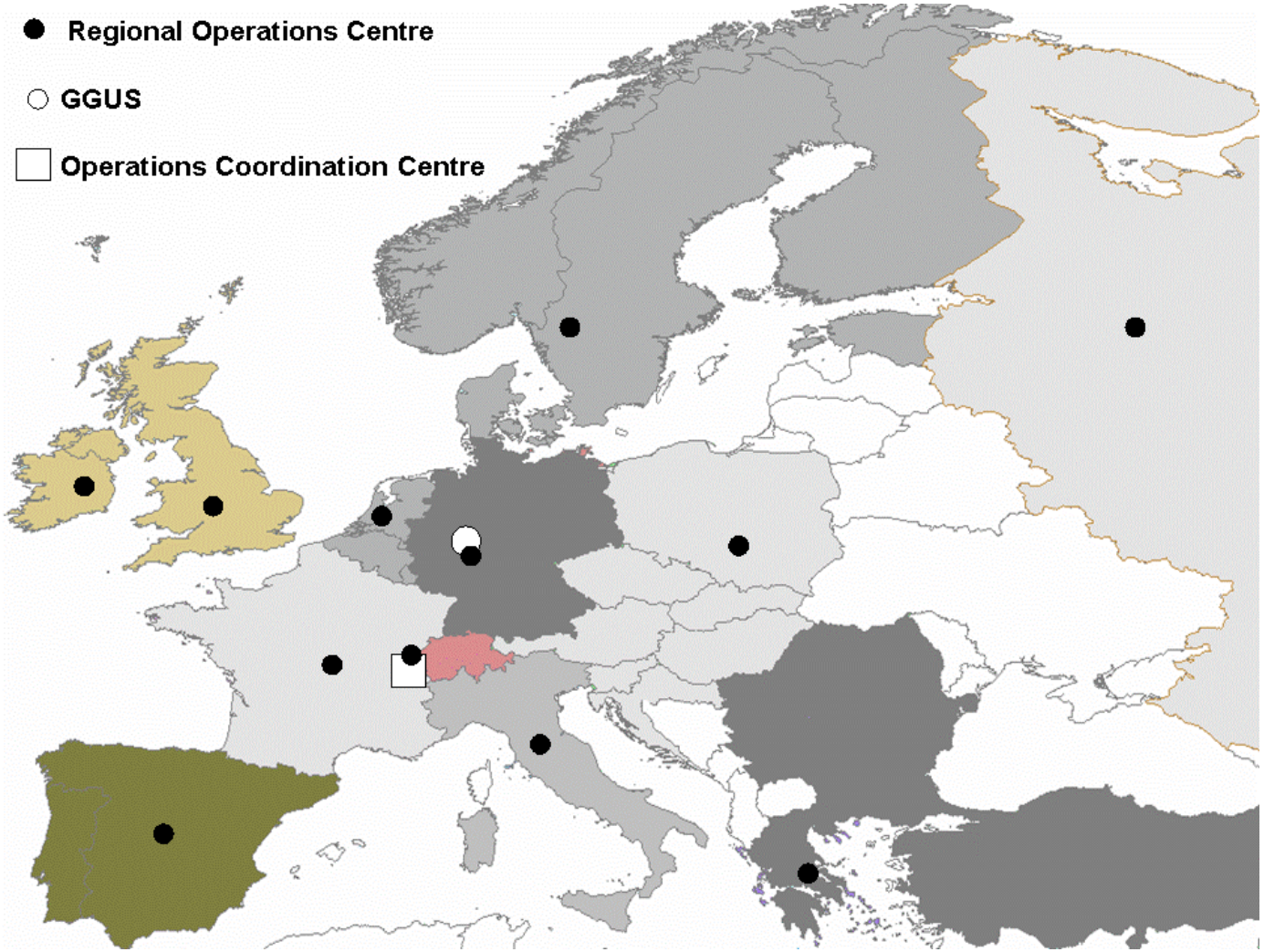


User communities

- Regular increase of infrastructure growth since 2004
  - > 20,000 registered users



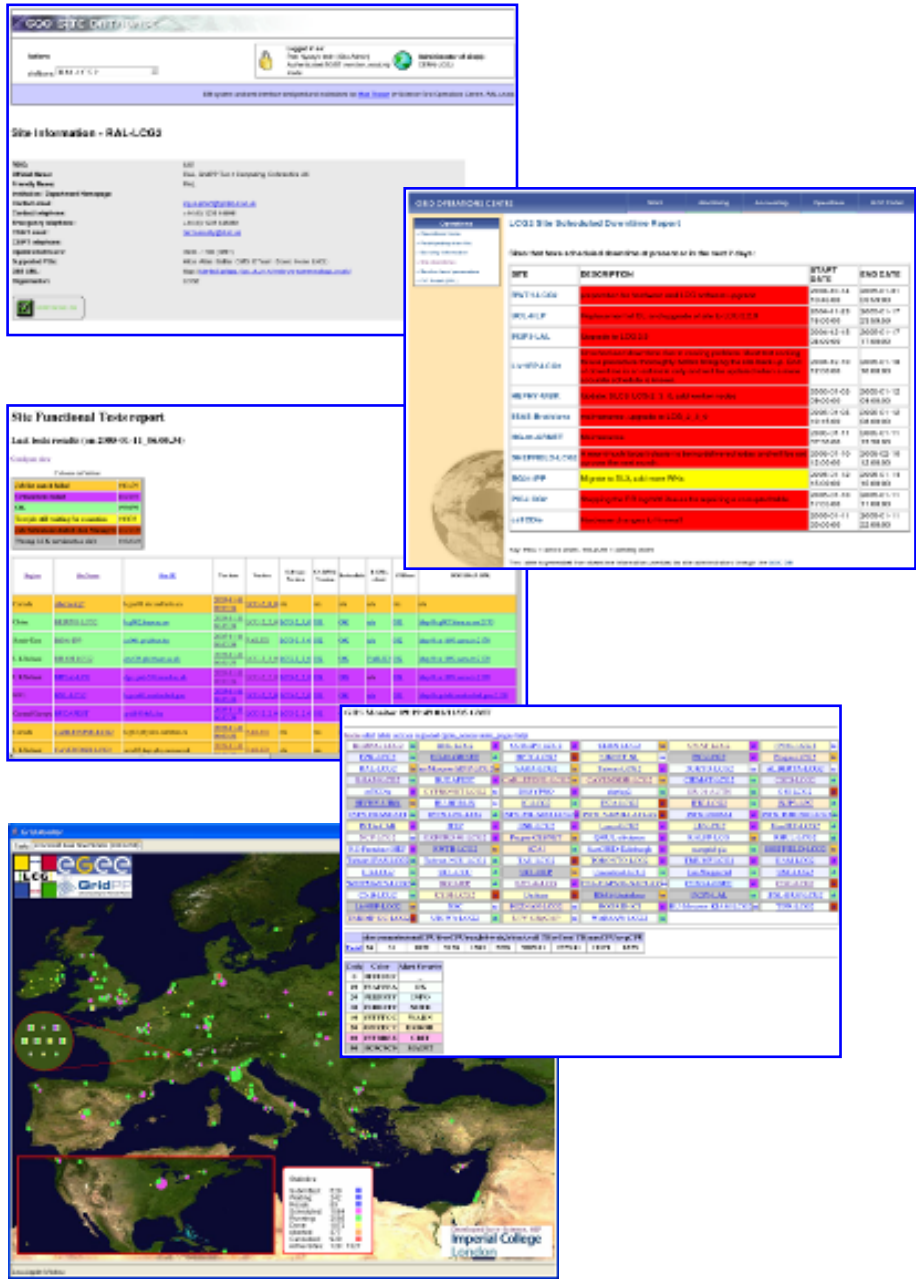
Grid management: structure



- Operations Coordination Centre (OCC)
  - management, oversight of all operational and support activities
- Regional Operations Centres (ROC)
  - providing the core of the support infrastructure, each supporting a number of resource centres within its region
  - Grid Operator on Duty
- Resource centres
  - providing resources (computing, storage, network, etc.);
- Grid User Support (GGUS)
  - At FZK, coordination and management of user support, single point of contact for users

Grid monitoring tools

- Tools used by the Grid Operator on Duty team to detect problems
- CIC portal http://cic.gridops.org/
  - single entry point
  - Integrated view of monitoring tools
- Site Functional Tests (SFT)
- Service Availability Monitoring (SAM)
- Grid Operations Centre Core Database (GOCDB)
- GIIS monitor (Gstat)
- …

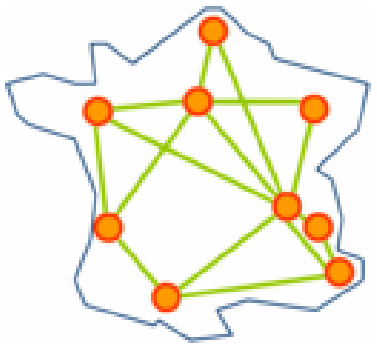


The Grid5000 infrastructure

- Research-oriented infrastructure
  - Understand behavior of grid systems
  - Between production-scale and simulation
  - 8000+ CPU cores deployed in 9 sites over France
- In 2014

Cores \ Sites	Grenoble	Lille	Luxembourg	Lyon	Nancy	Reims	Rennes	Sophia	Toulouse	Cores total
AMD Opteron 2218								200	560	**760**
AMD Opteron 250				158						**158**
AMD Opteron 275								224		**224**
AMD Opteron 6164						1056	960			**2016**
Intel Xeon E5-2620				48						**48**
Intel Xeon E5-2630				240						**240**
Intel Xeon E5-2630L			192							**192**
Intel Xeon E5-2650					64					**64**
Intel Xeon E5420	272									**272**
Intel Xeon E5440		368								**368**
Intel Xeon E5520	656							360		**1016**
Intel Xeon E5620		224								**224**
Intel Xeon L5335			176							**176**
Intel Xeon L5420					736		512			**1248**
Intel Xeon X3440					576					**576**
Intel Xeon X5570							200			**200**
Sites total	928	592	368	446	1376	1056	1672	784	560	**7782**

Cards \ Sites	Grenoble	Lille	Luxembourg	Lyon	Nancy	Rennes	Sophia	Cards total
			38					**38**
Intel				20				**20**
Intel 82599EB 10-Gigabit SFI/SFP+ Network Connection				4				**4**
Mellanox MT25418					1	65		**66**
Mellanox MT26418	34				234			**268**
Mellanox MT26428	82							**82**
Myricom 10G-PCIE-8A-C		46						**46**
Myricom M3F-PCIXF-2							80	**80**
intel 82599EB					4			**4**
Sites total	116	46	38	24	239	65	80	**608**

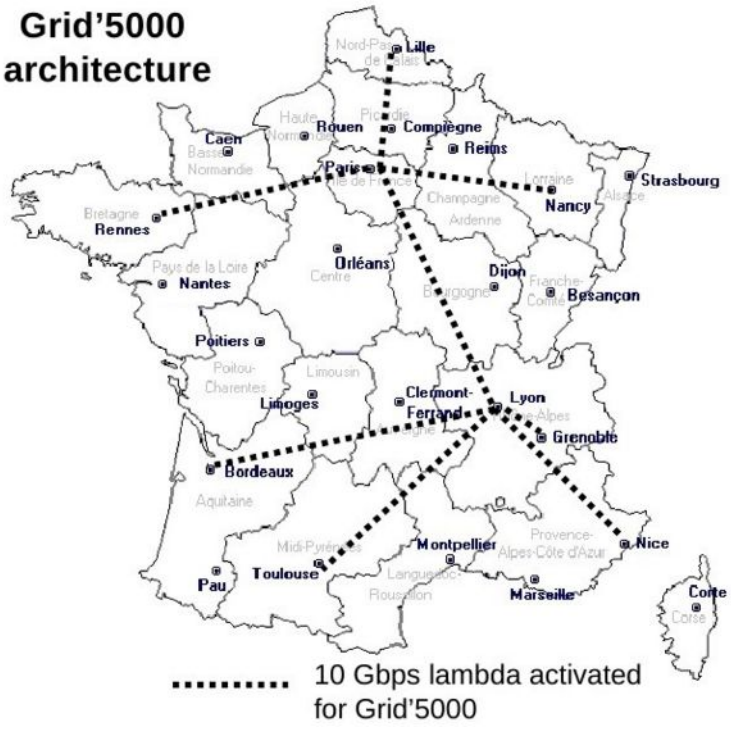


Grid5000 architecture

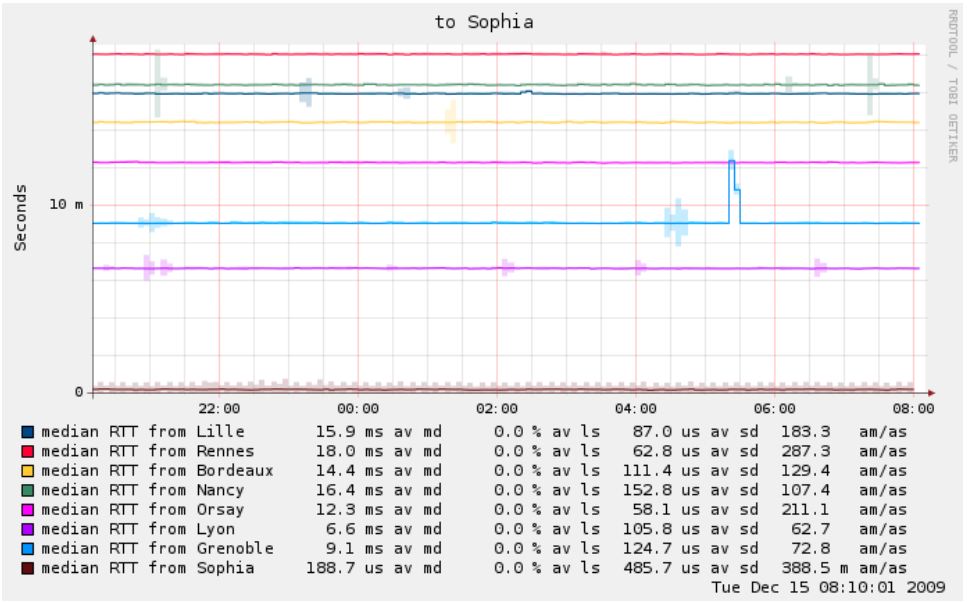
- Multi-clusters environment
  - One gateway per site
  - Multiple clusters per gateway
- Multi-sites user account management
  - User account replicated on all sites
  - NFS file sharing in each site
- Reserve resource, deploy anything
  - Advanced reservation of resource pools
  - System image deployment on reserved nodes
  - Nodes hot rebooting (with specified system image)
  - Get root access to all nodes
- Security
  - Confined environment (only ssh traffic from outside to gateways, no outbound connection from worker nodes)

Grid5000 network

- Dedicated 10Gbits links
- “Dark fiber” dedicated lambdas



- Monitoring tools



Tooling

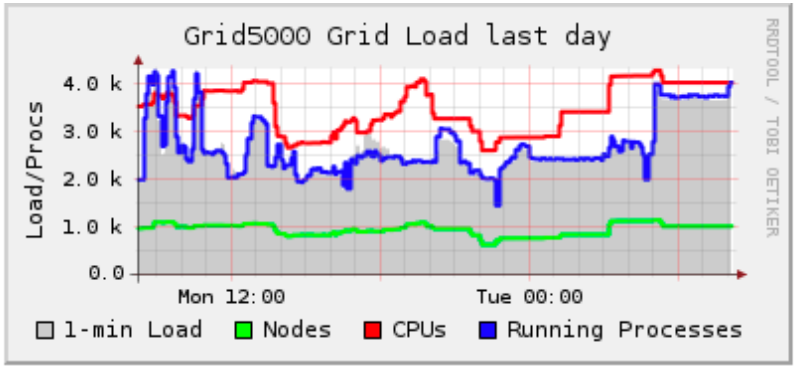
- OAR2
  - First-Fit Scheduler with matching resource (job/node properties)
  - Advance Reservation
  - Batch and Interactive jobs
  - Walltime
  - Hold and resume jobs
  - Multi-queues with priority
  - Best-effort queues (for exploiting idle resources)
  - Epilogue/Prologue scripts
  - No Daemon on compute nodes; rsh and ssh as remote execution protocols
  - Dynamic insertion/deletion of compute node
  - Logging
- kdeploy
  - System images deployment on reserved nodes

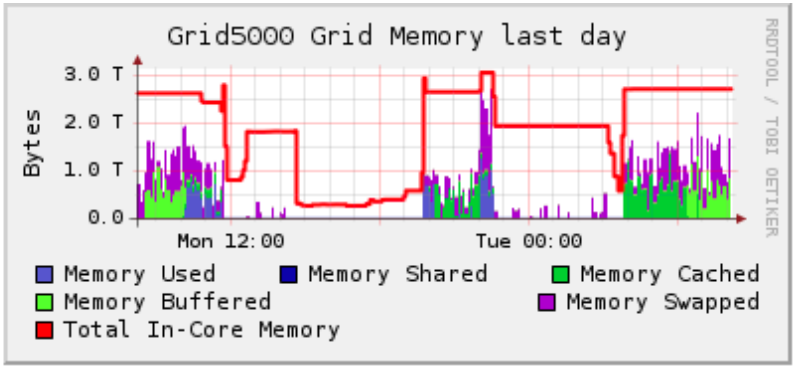
Tooling

- Monitoring tools

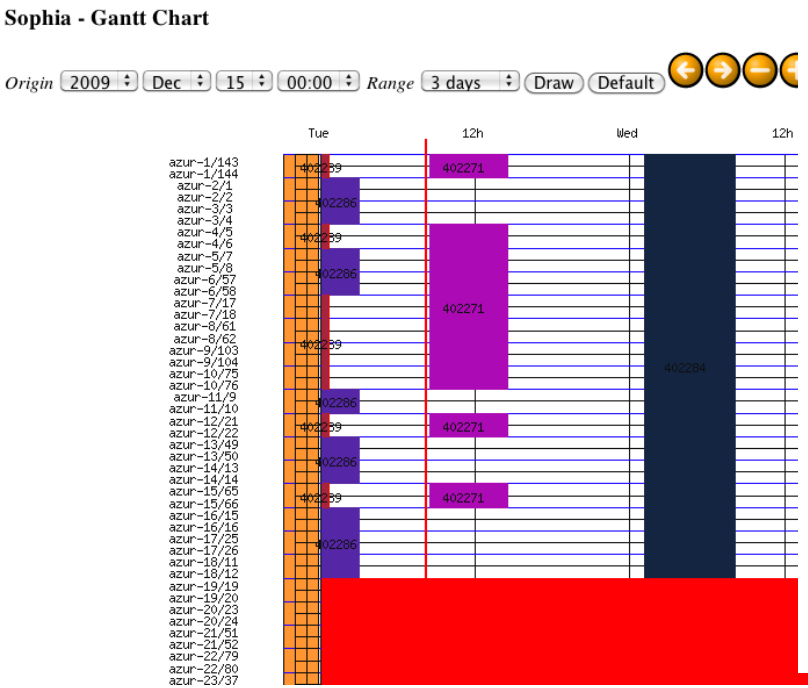
CPUs Total: **4018**
Hosts up: **1011**
Hosts down: **317**

Avg Load (15, 5, 1m):
93%, 97%, 98%
Localtime:
2009-12-15 08:13





- Schedules



From clusters to large-scale systems

- Distribution scale
  - LAN = highly reliable, high throughput network
  - WAN = not so reliable, high throughput but shared network
  - Security: WANs are more sensitive to attacks
- Performance
  - The computing power over data transmission performance ratio is lower with large-scale systems
  - Different resources, variable performances
- Heterogeneity
  - Causes hardware and system incompatibilities
  - Different sites with different administration policies
  - System virtualization and cloud technologies restore homogeneity to some extent
  - Other compatibility techniques used in particular cases: on-the-fly recompilation...

From clusters to large-scale systems

- Reliability
  - Probability of hardware failure increases
  - Administrative procedures frequency increases
  - Network scale impacts reliability
- Multi-disciplinary
  - Some multi-disciplinary clusters
  - Most grids are multi-disciplinary (scale effect)
  - Cross-administrative user communities (VOs)
  - Complex authorization and accounting schemes
- Standardization matters
  - W3C http://www.w3c.org/ (HTTP(S), *ML, SOAP, WSDL...)
  - OASIS http://www.oasis-open.org/ (WS, security...)
  - Open Grid Forum http://www.ogf.org/ (OGSA, WS-RF...)
  - ...

Clouds

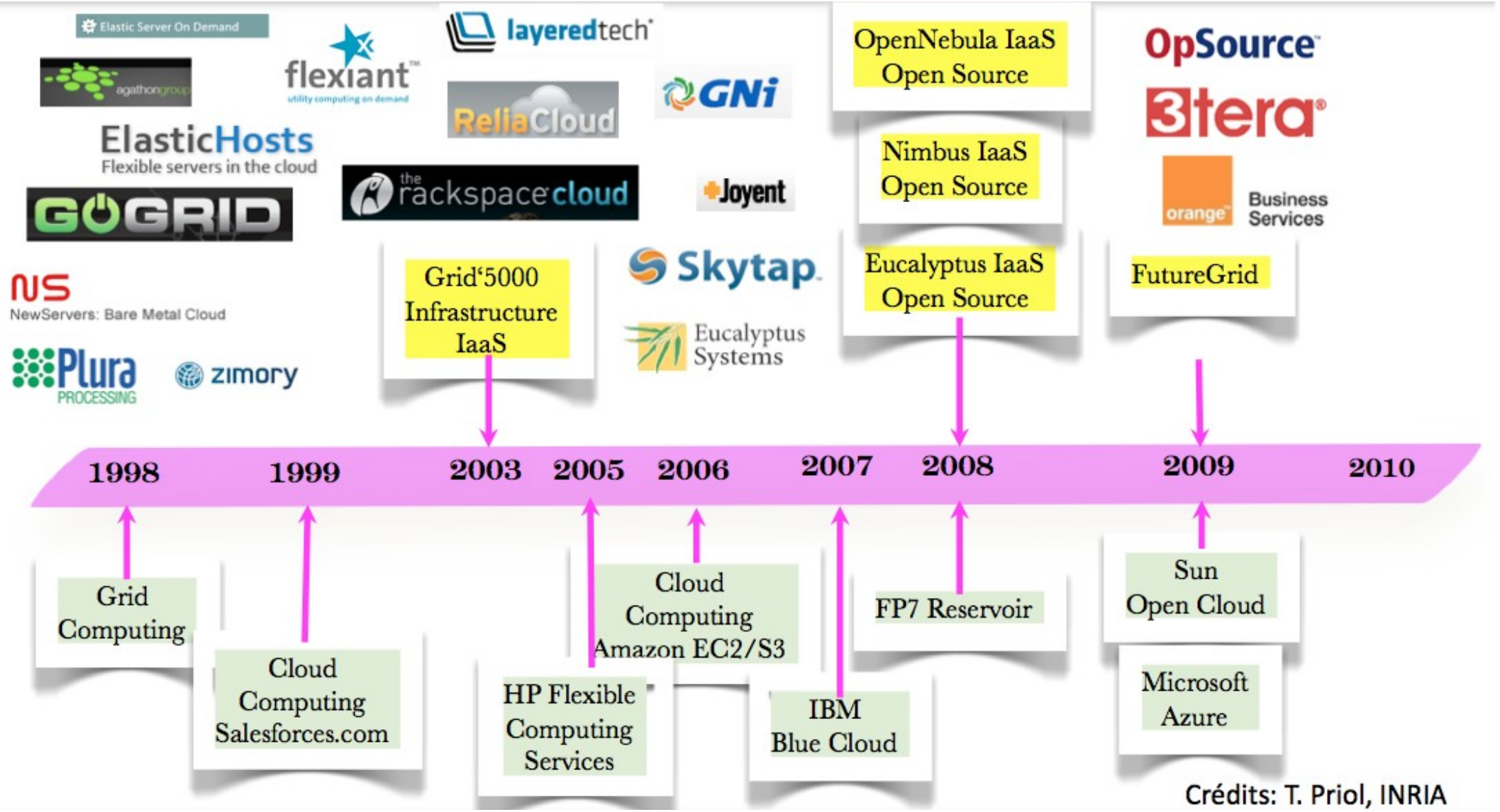
Clouds

- Externalized resources
  - Computing infrastructure provided as a service
  - Hosted in Data Centres (public cloud) or inside an organisation (private cloud)
  - Making extensive use of virtualization technologies
- Resource provider view
  - Mutualize and better exploit available resources
  - Provide a cost model for commercial offers (pay-per-use: per CPU core per unit of time, per GigaByte stored...)
- User view
  - Resources available on-demand
  - “Unbounded” capacity (elasticity)
  - No IT expertize required

Cloud computing

- Delivering computing resources as a service, on demand
  - Computing resources are allocated through a service interface
- Three levels of service commonly identified
  - IaaS: Infrastructure as a Service (compute, data and network resources)
  - PaaS: Platform as a Service (infrastructure + facilities for application development, deployment, maintenance)
  - SaaS: Software as a Service (complete application stack)
- Motivations
  - Convergence of services industry and infrastructure provision
  - Externalize management of computing resources to external resource providers
  - Scale-effect: benefit from very large data-centers
  - Business-centric: pay resources as you go

History of on-demand computing



Cloud and Grid computing

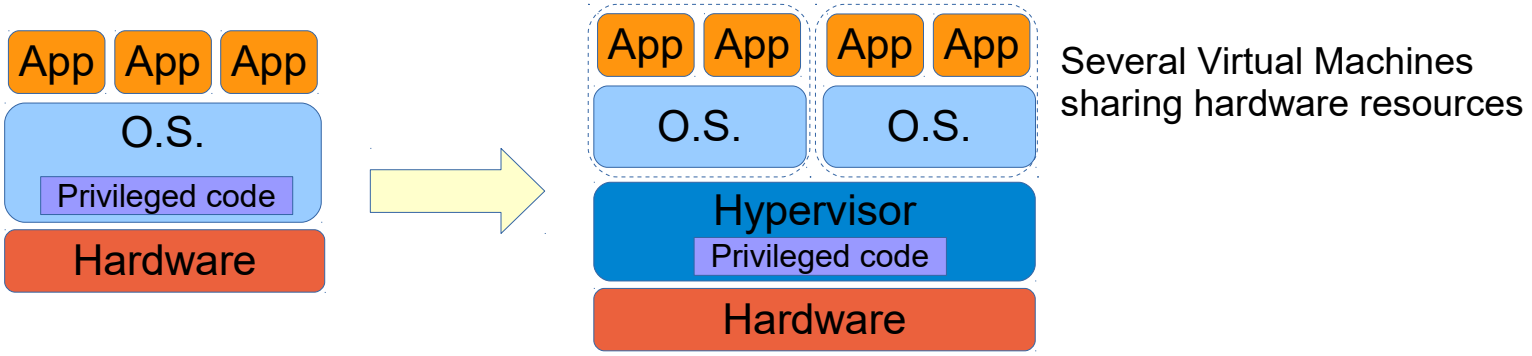
- Clusters, grid computing and cloud computing: the combination of multiple computers into larger metacomputers
- Clouds focus is on resources provision
- Grid focus is on middleware capability
- Complementary
  - Early cloud solutions deliver infrastructure
    - Complete middleware stacks now available for intensive computing
  - Cloud solutions may be generic or very application specific
- Convergence of technologies

Virtualization

- Abstraction of computer resources
  - Make it possible for a single physical computer to emulate several (different) virtual machines
  - Separate the OS from the underlying platform resources
  - Several OSes and software stack can co-exist on a physical resource
  - Resources (CPU time, memory, network...) are shared among virtual machines
- Why?
  - Hardware is underutilized, especially multi-core machines
  - Some services require permanent operation but limited machine resources
  - Grouping services on a single host is much more energy efficient
  - Data centers run out of space
  - Increased flexibility and lower system administration costs

Hardware virtualization principle

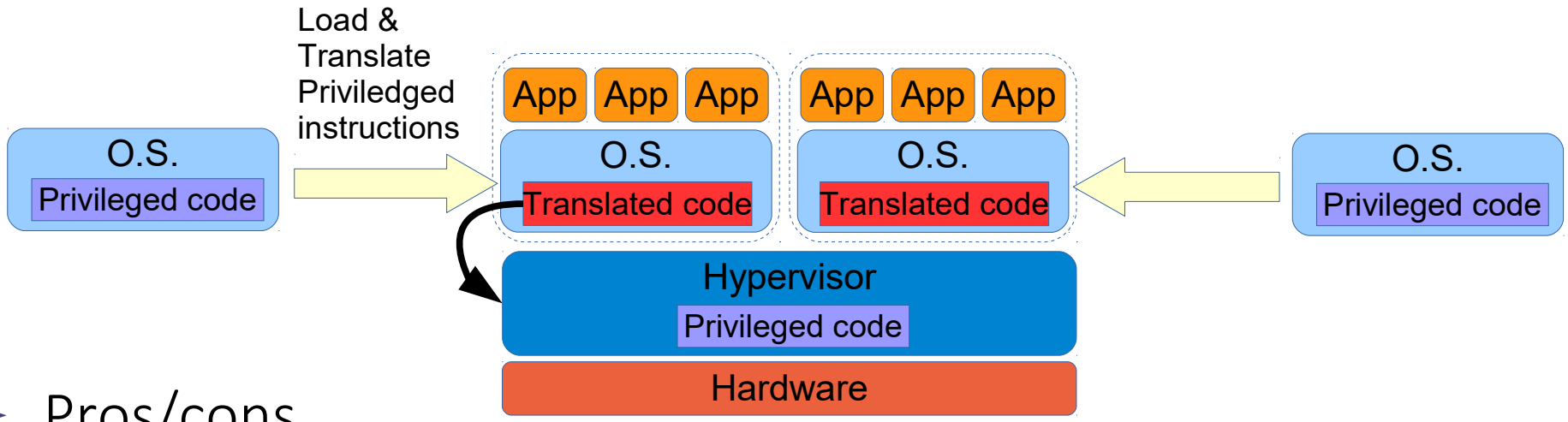
- Add an hypervisor layer between OS and hardware



- Different techniques
  - Full virtualization through binary translation or CPU-assistance
  - Kernel-level virtualization
  - Paravirtualization for hypervisor-aware systems

Unmodified OS virtualization

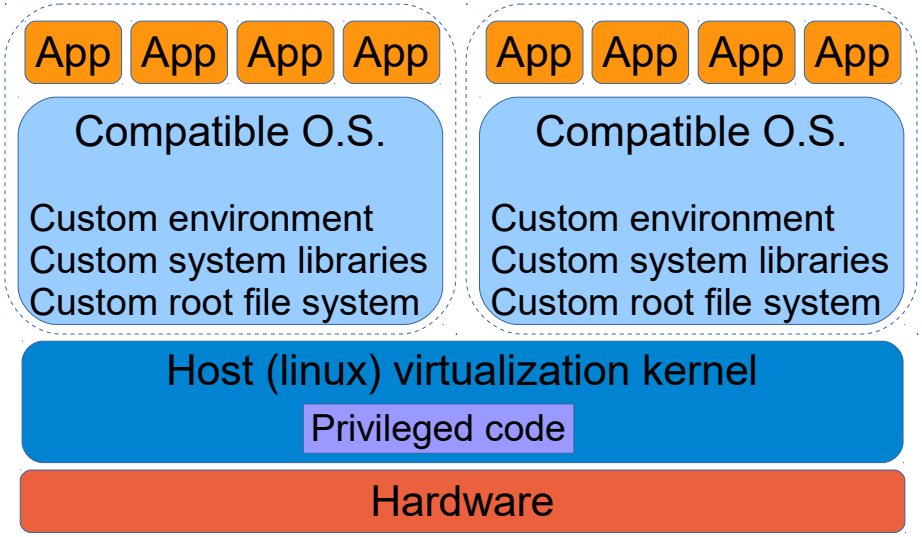
- Run native guest OSes on top of a host OS



- Pros/cons
  - + Works with any (unmodified) OS
  - + Host OS unaware of virtualization layer
  - - Complex virtualization process with binary code translation and CPU calls emulation
  - - All hardware devices usually not supported
  - - Performance impact of emulation
- Example: VMware

Kernel-level virtualization

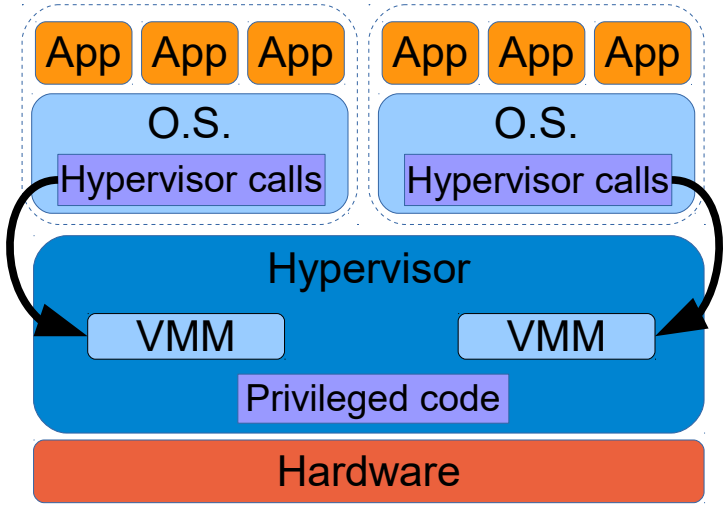
- Homogeneous OSes running on the same hardware



- Pros/cons
  - + reduced overhead of virtualization process
  - - limited to homogeneous OSes
- Example: KVM

Para-virtualization

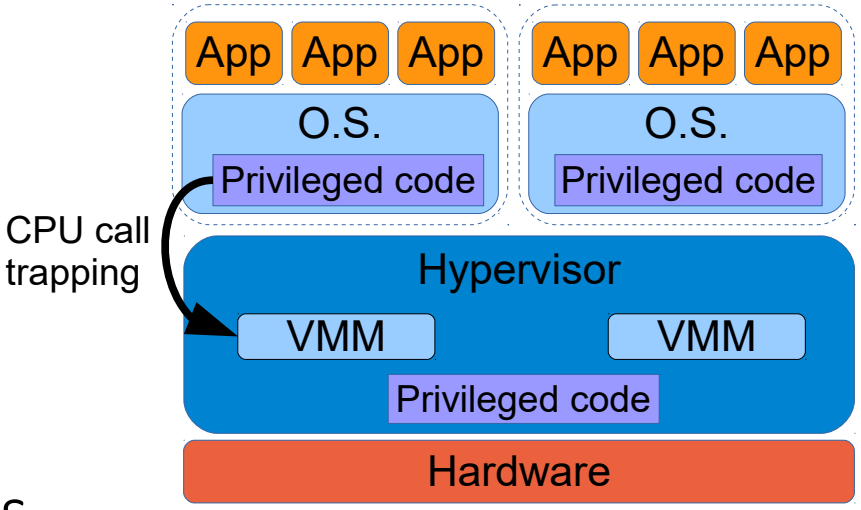
- Cooperating OSes-hypervisor
  - Each OS is controlled by a Virtual Machine Monitor (VMM) in the hypervisor



- Pros/Cons
  - + Low hypervisor overhead
  - - With compliant OSes only
- Example: Xen

Hardware-assisted virtualization

- CPU support for virtualization
  - Trapping privileged access from OS and notifying hypervisor



- Pros / Cons

  + Combine advantages of para-virtualization and full virtualization (unmodified OSes support)

  + Reduced privileged code trapping overhead

  - Still some performance penalty and several generations of hardware support

- Examples: Intel-VT and AMD-V compliant processors

More virtualization

- Virtual memory unit virtualization
  - Memory is the most critical component to be shared between several guest OSes
  - Hardware-assist memory virtualization in latest x86 CPUs
- Devices virtualization
  - Other devices may require virtualization (GPUs, network...)
  - Network virtualization gives more control / guarantee on network bandwidth dedicated to applications
  - Network devices increasingly programmable in software
- System manipulations by hypervisor
  - Suspend / resume / stop system activity
  - Write VM memory state on disk / restore
  - Migrate VM from one host to another
  - Checkpoint and run fail-safe backup VM...

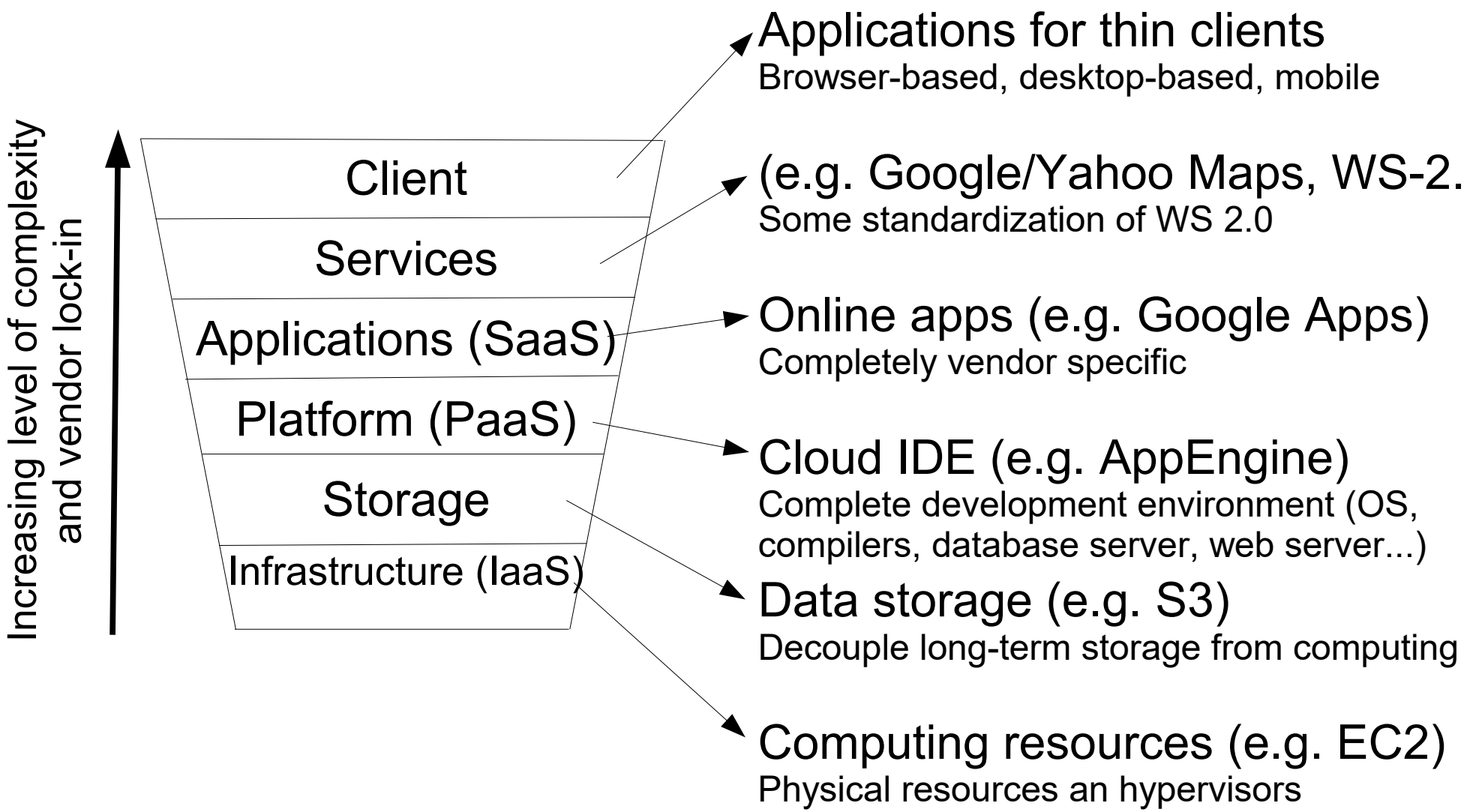
Use of virtualization technologies in Clouds

- Virtual Machine
  - A virtualized machine emulates a physical computing environment (including OS, libraries, applications) and request for hardware resources to an hypervisor
  - VMs can be started (OS boot) / paused / resumed / stopped
  - VMs are bundled in image files, easily transported
- Resources allocation
  - Physical resources can be allocated to a VM on-demand (e.g. amount of memory available, number of CPU cores...)
  - A computer can host as many VMs as its physical resources can bear
- Cloud infrastructures use VMs
  - For user environment installation and deployment
  - To easily balance load over existing physical servers

Cloud computing

- Strong industry involvement
- Use virtualization to adapt to all customer needs
  - To adapt hardware environments to software needs rather than the opposite
  - Free-up the relationship between software and hardware
  - Large hypervisor pool to allocate resources on-demand
    - Elastic resources provisioning framework
- Automation
  - Virtualized VMs can easily be managed to reach garanteed quality of services
- Service-oriented
  - The cloud itself is a service accessible / configurable through an API
  - Quality of service is usually guaranteed by contract

Heterogeneous interfaces



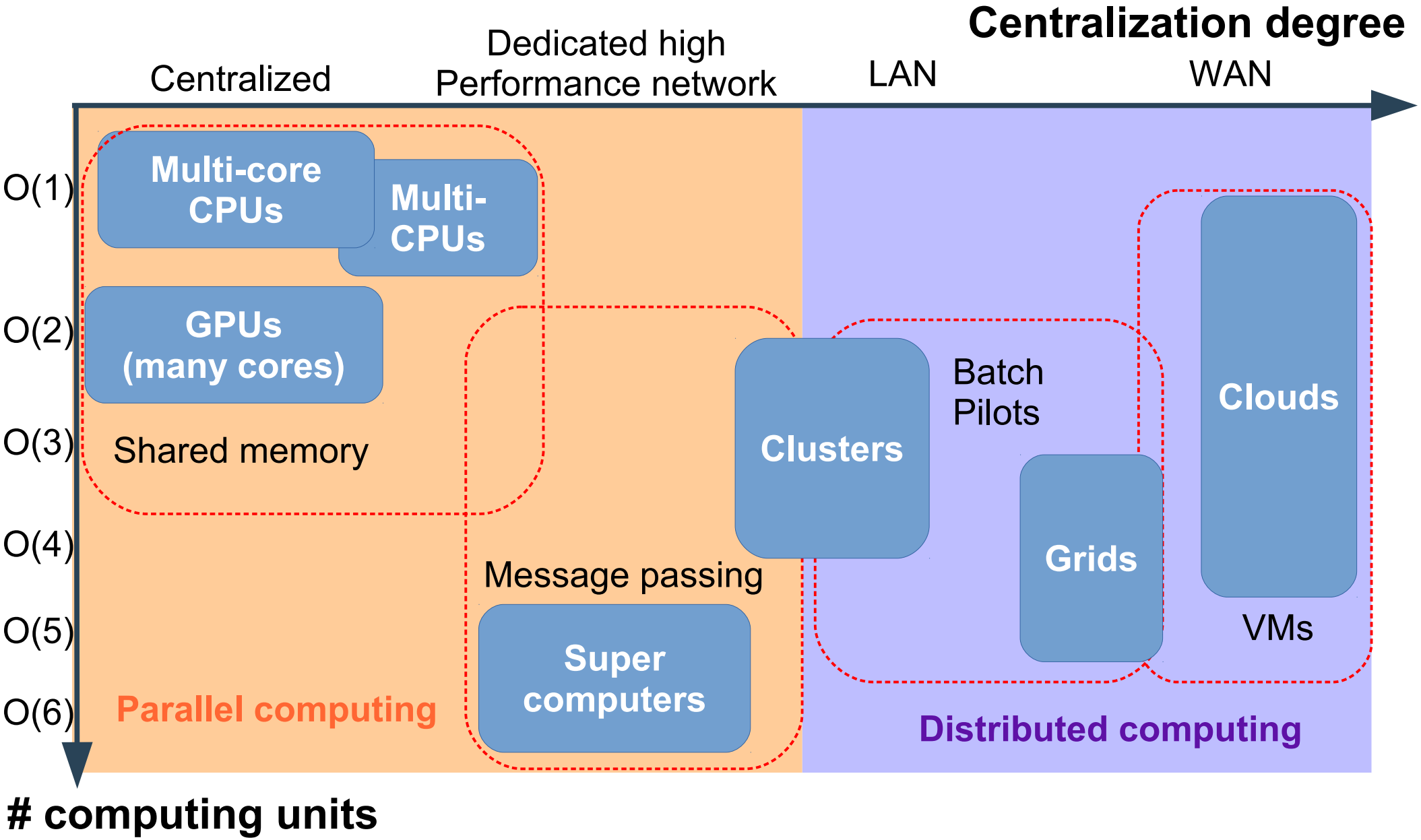
Amazon Web Services example

http://aws.amazon.com

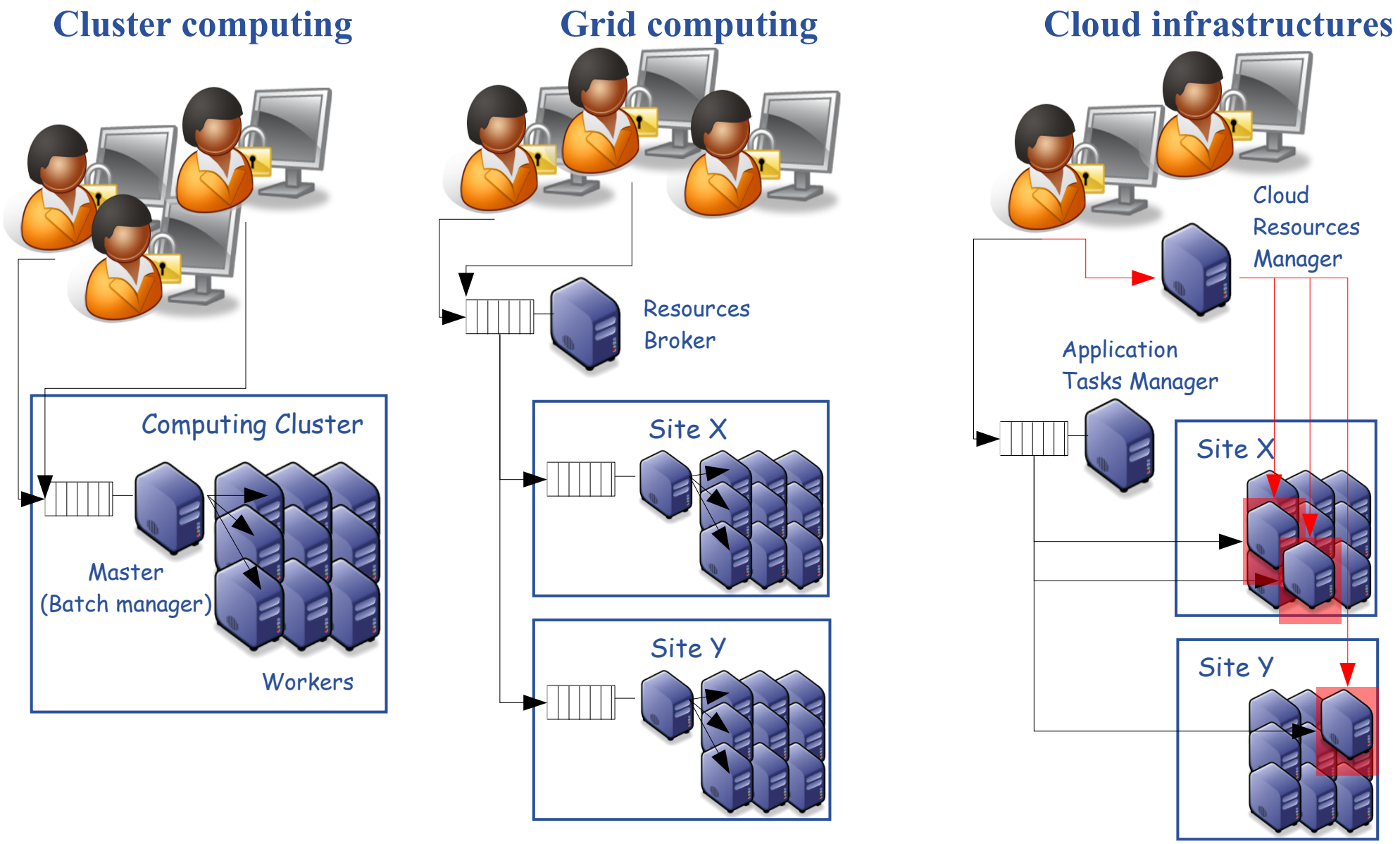
- Extensive cloud-oriented services portofolio
- Web Service-oriented API
  - SOAP and/or RESTful API
  - Bindings for several languages
- From simple VM delivery to complete business hosting
  - IaaS
    - Virtual servers, storage servers, VPN...
  - PaaS
    - Multi-platform development environments, integration tools
    - Identity management, data security
    - Elastic scaling, workload splitting, load balancing
    - Platform monitoring
    - Databases (relational and NoSQL), web server, email server...
  - SaaS
    - Data analysis services, workflow managers
    - Communication services, data transcoding and streaming

Clusters, grids and clouds

Parallel and distributed computing



Resources management for compute intensive applications



Distributed data management

Distributed data

- Scientific data records increase permanently
  - Astronomy / astrophysics observations
  - Satellite data, climate, atmosphere, geophysics data
  - Epidemiology data, medical records
  - Biological data, gene annotations and structure, genomes
  - Scientific instruments, *e.g.* high energy physics records
- Target PB repositories
- Usually distributed
- Potentially sensitive
- File systems limitations
  - 2^{31} bytes per file / inodes (towards 64 bytes file systems)
  - ~10000 files/directory

Requirements

- Very large scale distribution, transparent access
  - Heterogeneous formats
  - Virtualization of distributed resources
  - Coherency of remote data updates and replicated data
- Performance, scalability
  - Data transfers and data access strategies: often depend on access patterns
  - Parallel access, multiple users
- Fault tolerance
  - Resources failures and network service interruption
- Reliability
  - Long term availability of data, non repudiation
- Access control, data protection
  - Flexible access control, on-storage and on-network protection

Distributed data management

- Centralized approach: file catalogs, indexes
  - Handles heterogeneity, legacy storage
  - Direct access to data, bottlenecks (limited scalability), central point(s) of failure
  - Depend on external storage data management policies
  - Ease coherency and data protection
- Decentralized approach: peer-to-peer
  - Very scalable, no critical point, distribute dat search load
  - Strategies performance dependent on data access patterns
  - Robust, unreliable environment with many peer failures
  - Low data protection
- Hybrid centralized / decentralized
  - Replicated catalogs or P2P networks with redundancy, QoS...

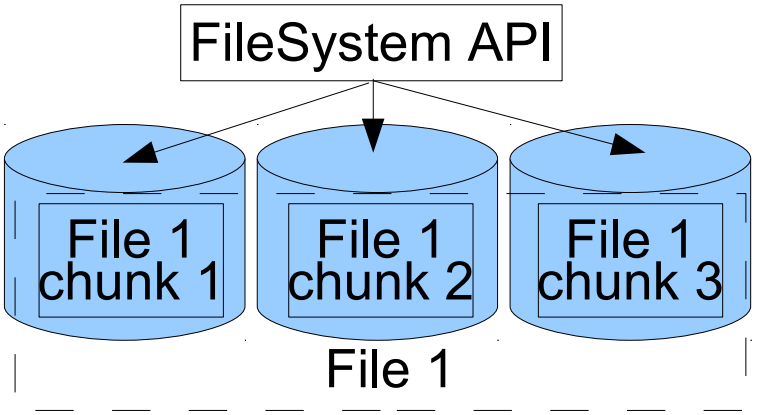
Distributed file systems

Distributed file systems

- First idea: extend existing approach (local file system) to distributed resources
- Parallel I/O
  - Performance
- Network File System
  - Network extension of file systems
- Andrew File System
  - Secured, large-scale extension with collaborative caching
- Grid File System
  - Emphasis on heterogeneity management
  - Ambitious objectives for information life-cycle management

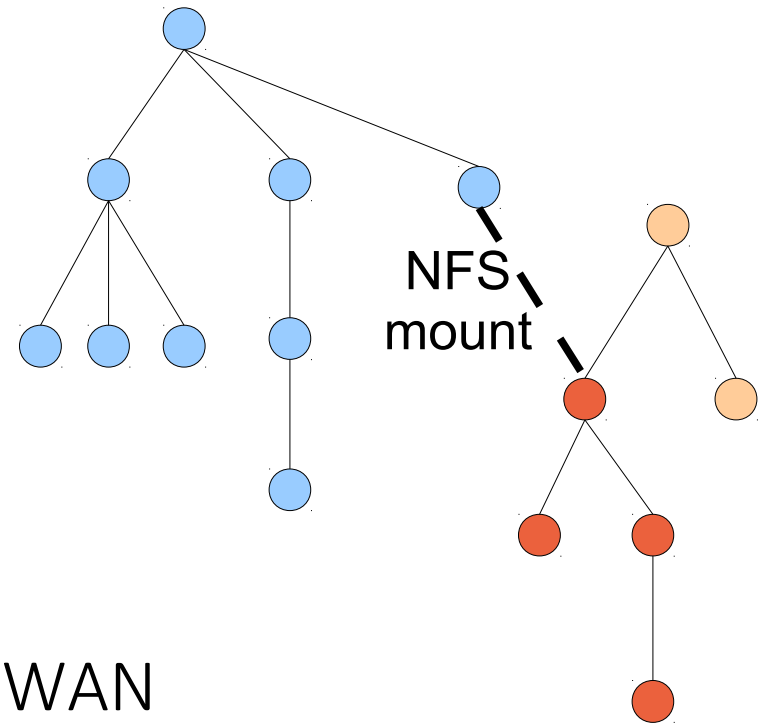
Parallel file systems

- Focus on high performance
  - Local resources with high connectivity
  - Independent industrial implementations (IBM, SGI...)
- Performance
  - Parallel I/O
  - Can be exploited by parallel programs...
  - ...or sequential programs in case of disk bus saturation
  - Dedicated architectures available
- Storage Area Network (SAN)
  - Network interfaced storage resources
  - High performance network as disk bus (fiber channel)



NFS: Network File System

- Multiple (partial) file systems viewed as one
- C/S model
  - Scalability limitation
  - Usually across LAN
- Security limitations
  - User IDs mapping?
  - Special control for UID 0 (root)
  - Transfers over WAN?
- Obvious performance limitations over WAN
  - Caches
  - Automount
  - ...

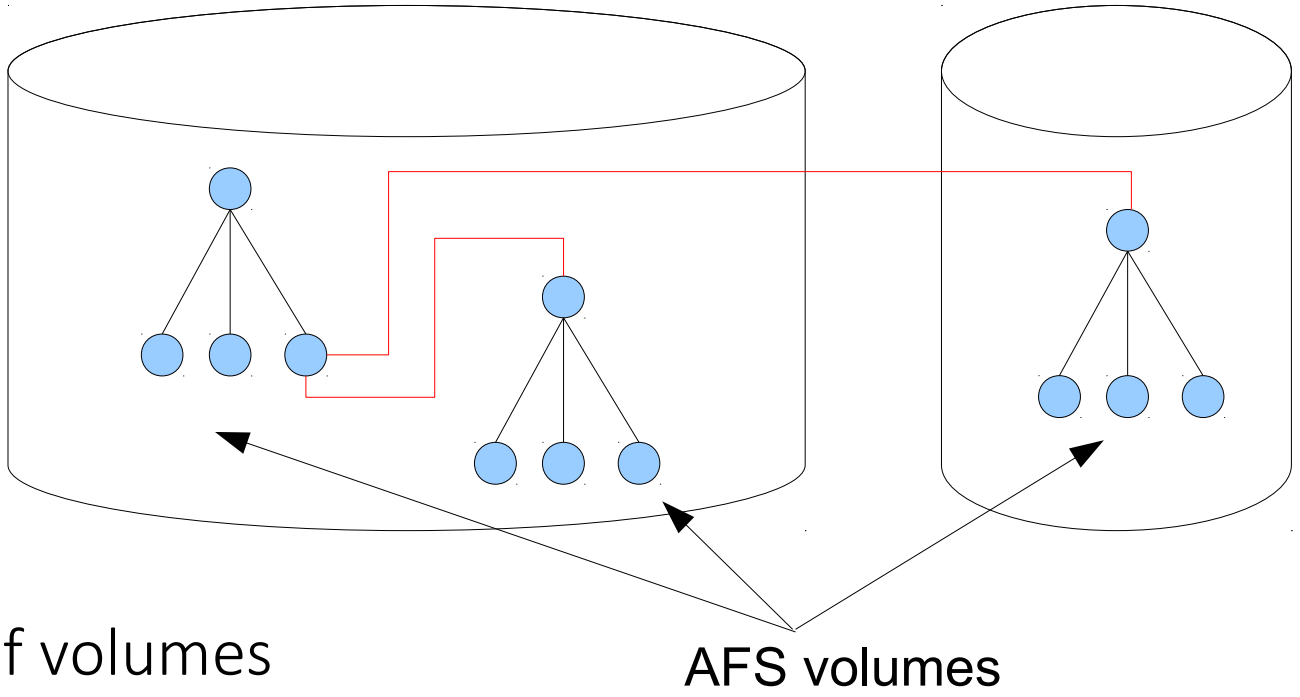


AFS: Andrew File System

- File system on the NFS model with focus on
  - Security (Kerberos authentication, ACL control)
  - Distribution (caches)
  - Scalability (tens of thousand client per cells)
- Collaborative caching
  - File locking strategy for ensuring coherency during updates (avoid too large, shared records)
  - Modification on local caches
  - Cached files listed on AFS server
  - Notification mechanisms in case of file modification to all caches (with recovery on network failure)

AFS: Andrew File System

- Space partitioning by AFS volumes
  - Files hierarchy hosted on a single storage device
  - Logical view (mountpoints, migration of volumes possible)



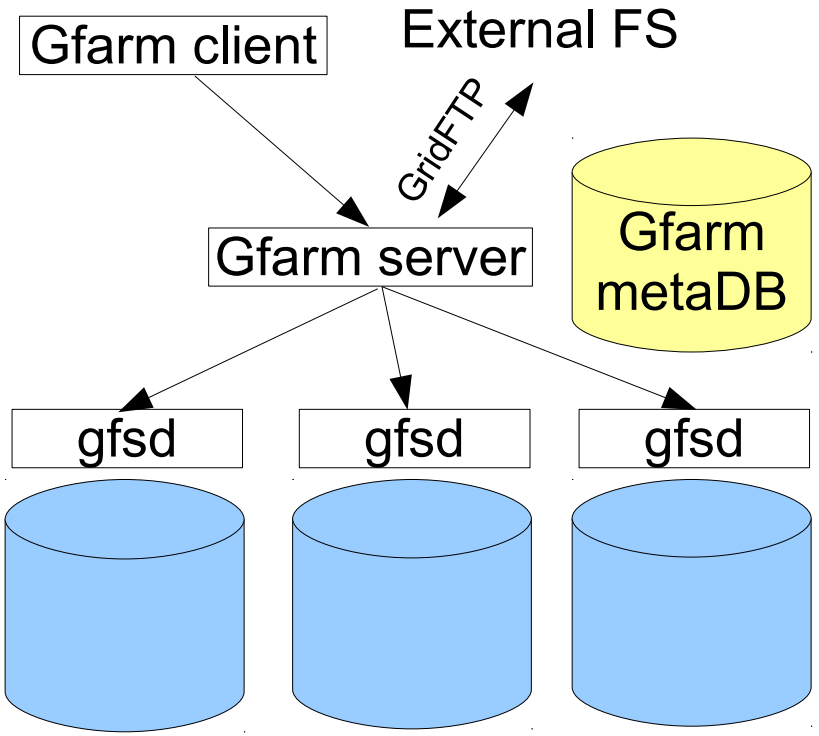
- Clones
  - Read-only copies of volumes
- Influenced NFSv4 features

GFS: Grid File System

- Open Grid Forum working group standard
- Targets
  - Standard interface for multiple resources
  - Plug-n-play resources
  - Federation of logical resource name space
  - Information lifecycle management (data placement and retention policies)
  - Object based storage
  - Context management
  - Bulk and asynchronous operations
- Storage resources virtualization
  - Abstraction layer to manage heterogeneity

GFS implementation: Gfarm

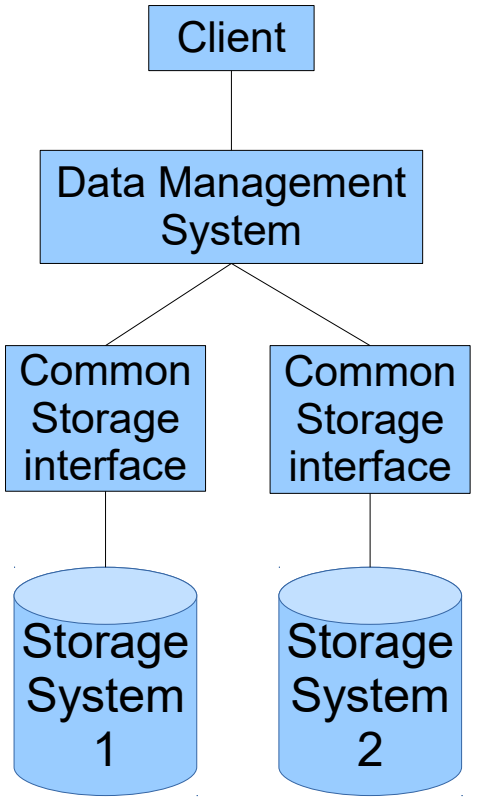
- Architecture
  - Local resource virtualization through gfarm daemon
  - Specific C/S protocol + external protocols supported (e.g. GridFTP)
  - System metadata store
- Reliability
  - Replication
- Performance
  - Parallel IO
- Interoperability
  - Grid credentials recognized
  - FUSE component to mount on UNIX file systems



File catalogs and replication

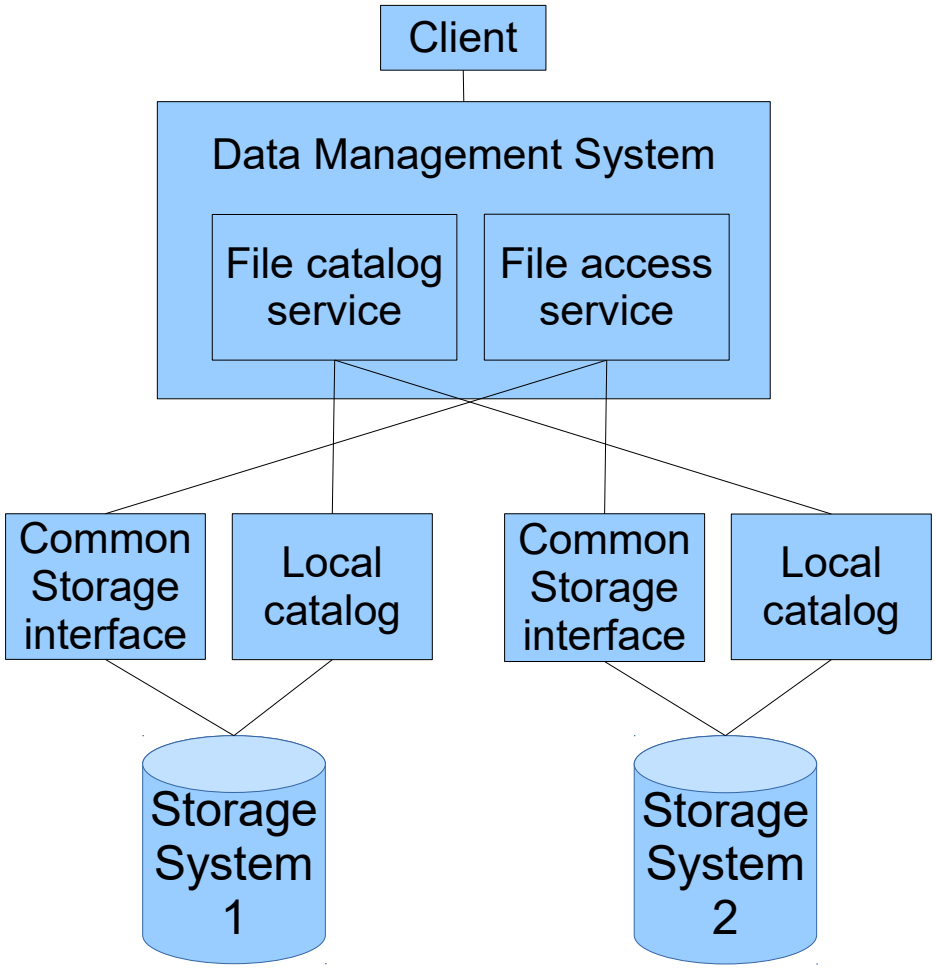
Splitting storage and data management

- Handle heterogeneity
  - Standard interface to storage resources
  - Storage service
- Manage data at a higher level: Additional services to handle:
  - Distribution, load management
  - Availability, replication
  - Performance, caching, transfers scheduling
  - ...
- Require adequate support at storage-level
  - Security (data access control, data protection)
  - Data locks...



File catalog

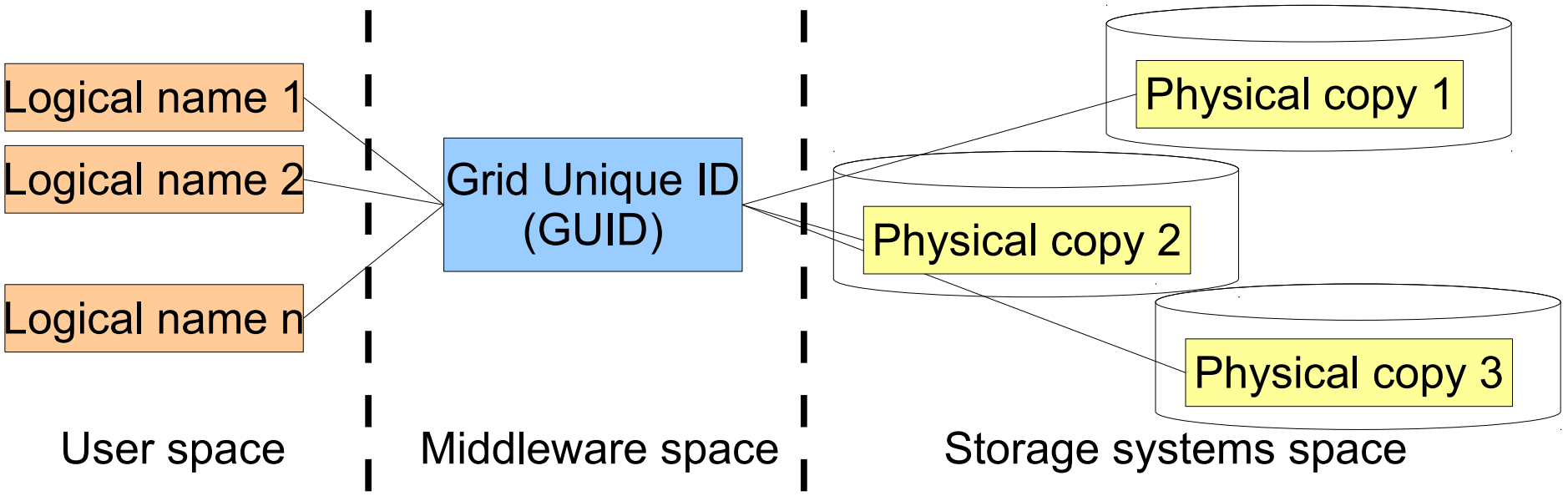
- Unique view of file hierarchy
  - (Local) sub-catalogs mapping to a single file tree view
  - Centralized entry point
- Additional services
  - Access control
  - System metadata (checksum...)
  - User-defined metadata



File replication

- Files replication over different sites enable
  - Improved performance (use closest replica)
  - Improved reliability (if a server is out of reach, replica may still be available)
  - Easy to set up (in read-only mode at least)
- Drawbacks
  - Multiple copies coherency problem
  - Define replication policies: by hand or automatic (mirror, partial mirror)
  - Does not solve the storage size granularity problem
- Distribution
  - Synchronize access controls on different storage
  - Give a logical view of several physical replica

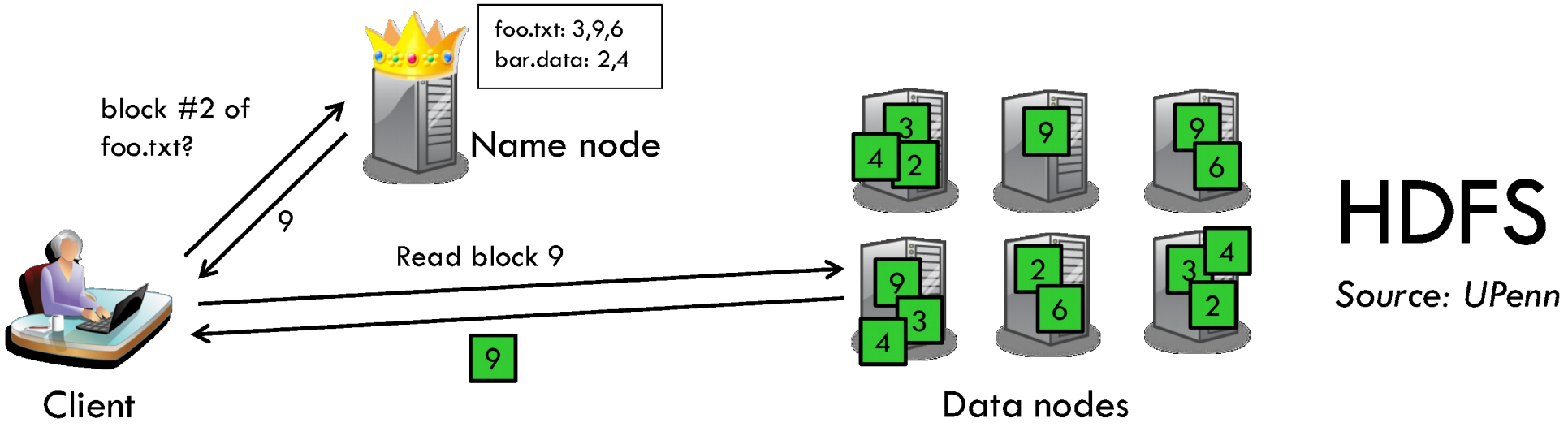
File Replication



- GUID: Grid-wide Unique IDentifier (system use)
- Logical names: user names, many-to-1 association
- Physical names: URI kind (location), 1-to-many association
- File catalogs map logical, system and physical spaces

Hadoop File System (HDFS)

- Core component of Hadoop
  - Designed to serve (massive) data pieces to Hadoop MapReduce
  - Data is divided in blocks and spread over different storage resources
    - A central catalog: Name Node
    - Many Data Nodes to store data



  - Hadoop allocates computing tasks on closest nodes to data

HDFS objectives

- Assumptions
  - Run on commodity hardware
    - Overlay for existing file systems
  - Unreliable hardware
    - Data replication
    - Data Node backup server
    - Journalized file system
    - Secondary Data Node to checkpoint file system operations
  - Very large data sets
    - Optimised for large files
    - Optimised for High Throughput of data rather than interactive use
  - Mostly read access
    - Write-once-read-many model: simple coherency management
    - Data can be extended but not modified

HDFS architecture

- Name Node
  - Centralized file catalog, with file hierarchy structure metadata
  - Monitor data nodes
  - Makes data placement decisions
    - Takes into account network topology
  - Critical point of failure, can be backed up by an alternate server
- Secondary Name Node
  - Persist file system transaction log operations frequently
  - Allows for quick restart on failure
- Data Nodes
  - Manipulate chunks of information
  - Take advantage of MapReduce (key, value) data indexing mechanisms

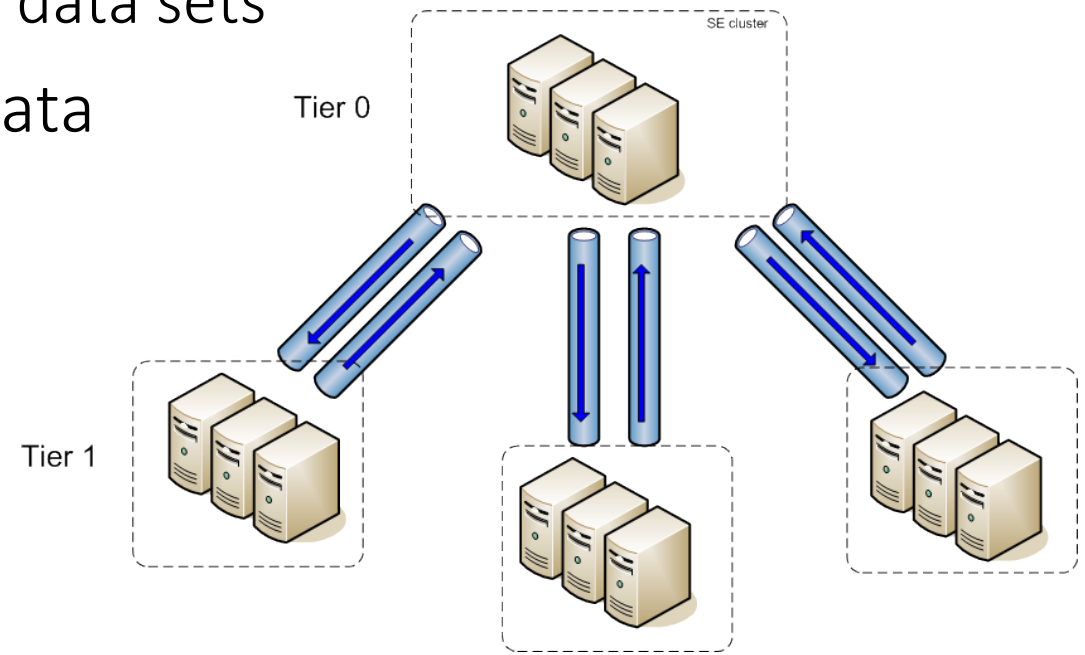
File transfer

GridFTP

- Security
  - Grid credentials-based authentication and authorization (single sign-on)
- Third-party transfer
  - Server-to-server file transfers for administration needs
- Performance
  - Multiple parallel TCP streams
- Striped
  - Data interleave
- Partial transfer
- Restart on failure
- QoS negotiation (buffer, window size)

File Transfer Service

- Sequential file transfer (e.g. FTP)
  - Unfair for smaller files
- File Transfer Service
  - Supports grid credential (single sign-on) and SRM protocol
  - Scheduled transfers
  - Error recovery
  - Simplified management of data sets
- For very large amounts of data



Conclusions

History

- 80s
  - ARPANet, WAN
  - Parallel computers
- Early 90s
  - Parallel computer crisis
  - Web
  - I-WAY
  - CASA
- Mid 90s
  - Web technologies
  - “Grid” term coined, GT2
- Early 2000's
  - Scientific grids (eScience)
  - Desktop grids
  - OGSA, WS-RF, GT4
  - GPU computing
- 2004
  - Multi-cores
- 2006
  - Virtualization, cloud computing
- 2008
  - Green computing
  - GPGPU, many-cores
  - CUDA
- 2010
  - Big Data era
  - Public clouds, IaaS
  - Data centres
  - Internet services
- 2015
  - Smart systems and IoT
- 2020: towards Exascale computing

Conclusions

- There is no limitation to the need for computing power
  - Except cost and energy consumption
- To increase power, architectures have evolved towards parallelism
  - Parallel micro-architectures
  - Aggregating resources at the largest scale
  - Mixed-levels parallelism increasingly needed
- All systems and compute intensive applications have been parallelized
- A convergence of complementary technologies is observed
  - Computing (batch, cloud…) grounded on clusters
  - Virtualization (systems, networks…) has become a key enabler
  - Global-scale (inter-domains) problems are being tackled

Publications

- Main journals
  - Journal of Grid Computing (JoGC)
  - Future Generation Computer Systems (FGCS)
  - Parallel Computing
  - Parallel and Distributed Computing
  - Int. Journal of High Performance Computing Appl. (IJHPCA)
- Main conferences
  - Int. Symposium on Cluster, Cloud and Grid Computing (CCGrid)
  - IEEE Grid
  - High Performance Distributed Computing (HPDC)
  - Int. Parallel and Distributed Processing Symposium (IPDPS)
  - SuperComputing (SC)
  - Europar