Operating system

Extensible Firmware Interface

Firmware

Hardware

BIOS, UEFI, EFI

BIOS: Die wichtigsten Tastenkombinationen

	AMI	Award	Phoenix
BIOS aufrufen	[Entf] [F1]	[Entf] [F2] [Strg Alt Esc] [Strg Alt S]	[Entf] [F1] [Strg Alt Esc] [Strg Alt S]
Beenden und speichern	[F10]	[F10]	[F10]
Beenden, ohne zu speichern	[Esc]	[Esc]	[Esc]
Standardwerte laden	[F6] [F9]	[F9]	[F7] [F9]
Hilfe	[F1]	[F1]	[F1]
Boot-Menü aufrufen	[F8] [F9]	[F12]	[Esc]

UEFI: Unified Extensible Firmware Interface

UEFI kann mehr

UEFI kann mehr

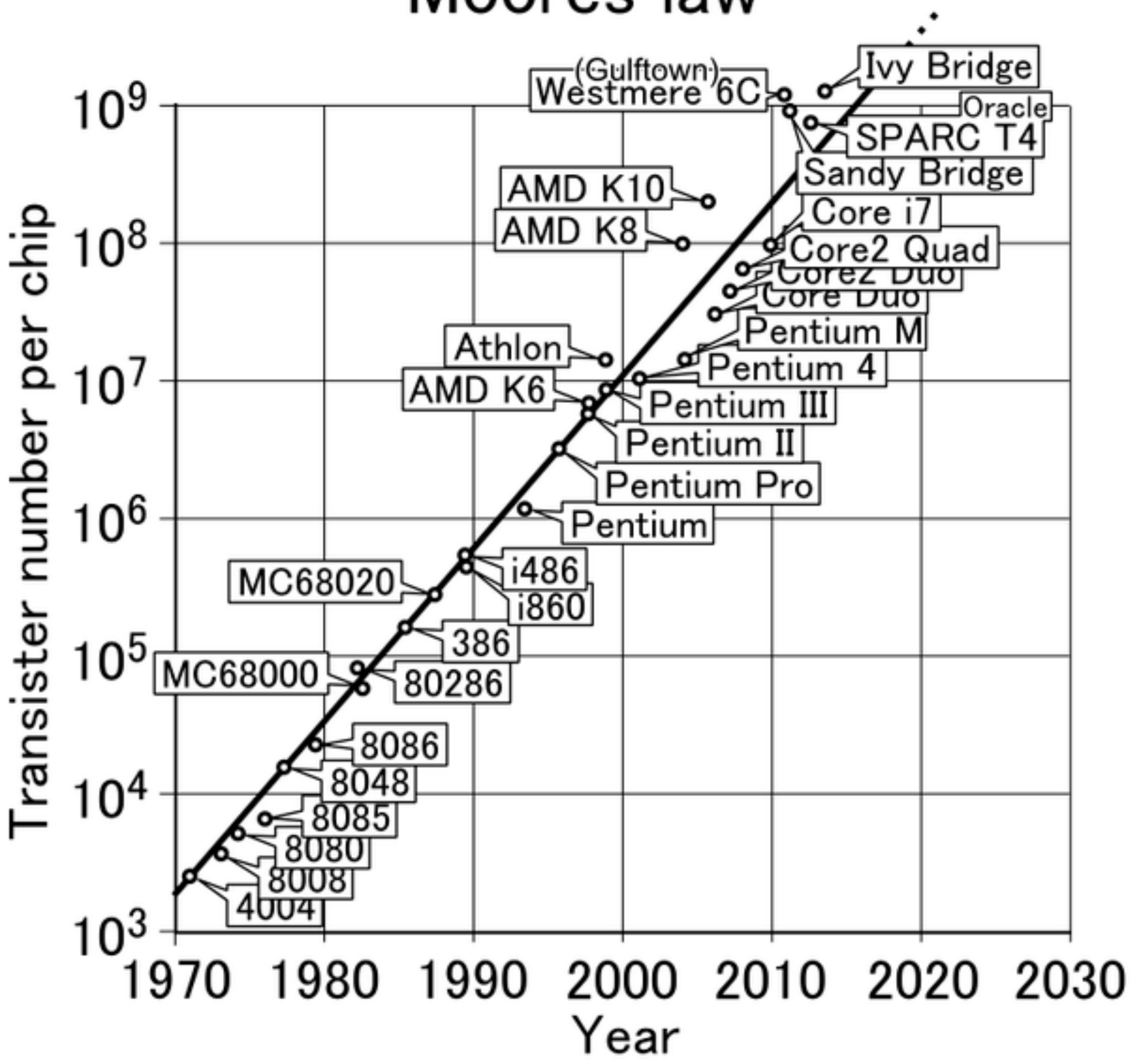
Das unterscheidet UEFI vom alten BIOS

	BIOS	UEFI
Partitionsschema	Master Boot Record (MBR)	Master Boot Record (MBR) GUID Partition Table (GPT)
Bootet Festplatten bis zu	2,2 Terabyte	8192 Exabyte
Mausunterstützung	○	●
Netzwerk- und Internetzugriff	○	●
Maximale Bildauflösung	800 x 600 Pixel, 16 Farben	Auflösung nur begrenzt durch die Grafikkarte
Kann Programme starten, etwa einen Browser	○	●

● = ja ○ = nein


msi
F12
Language
X
09:58
Thu 8 / 29 / 2013
Version E7758IMS V2.11
Intel(R) Core(TM) i5-3350P CPU @ 3.10GHz
Current Cpu Frequency 3.10 GHz (31 x 100 MHz)
Current DRAM Frequency 1600 MHz
Memory Size : 16384 MB
Temperature
CPU
42℃
Mainboard
System
31℃
Boot device priority
ECO mode
STANDARD mode
OC Genie II mode
HELP | HOT KEY |
Onboard LAN Configuration
Onboard LAN Controller [Enabled]
LAN Option ROM [Disabled]
Network stack [Disabled]
SATA Configuration
SATA Mode [AHCI Mode]
SATA1 Hot Plug [Disabled]
SATA2 Hot Plug [Disabled]
SATA3 Hot Plug [Disabled]
SATA4 Hot Plug [Disabled]
SATA5 Hot Plug [Disabled]
SATA6 Hot Plug [Disabled]
Audio Configuration
HD Audio Controller [Enabled]
HPET Configuration
HPET [Enabled]
Mainboard settings
SETTINGS
Overclocking setting
OC
Energy saving
ECO
Internet
Message and Mail
BROWSER
Live Update
M-Flash
UTILITIES
Password
SECURITY

Moores law

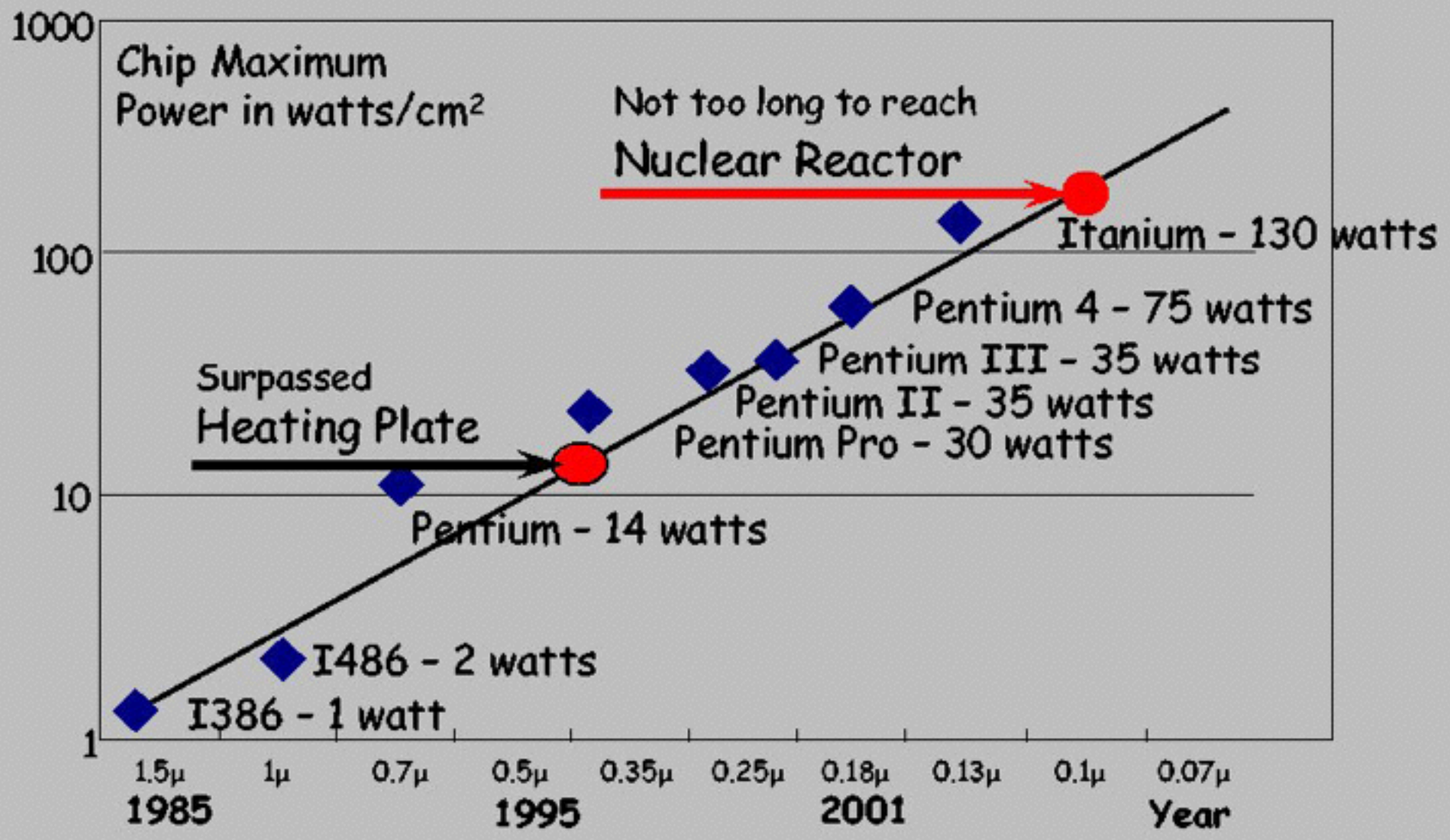

Transister number per chip
10^9
10^8
10^7
10^6
10^5
10^4
10^3
1970
1980
1990
2000
2010
2020
2030
Year
Ivy Bridge
(Gulftown) Westmere 6C
Oracle SPARC T4
Sandy Bridge
AMD K10
AMD K8
Core i7
Core2 Quad
Core2 Duo
Core Duo
Pentium M
Athlon
Pentium 4
AMD K6
Pentium III
Pentium II
Pentium Pro
Pentium
i486
MC68020
i860
386
MC68000
80286
8086
8048
8085
8080
8008
4004

Moore's law (cont'd)

- In past years: increase of the number of transistors was complemented by an increase of the processor frequency
- The combination lead to an annual increase of the processor performance of
  - 55% for integer operations
  - 75% for floating point operations
- By consequence: *If your problem is not too big you may want to wait until there is a machine that is sufficiently fast to do the job.*
- Or (maybe more realistic): However bad you write your code, the increased computer performance will hide it.
- (Un)fortunately this time has gone: The increase of the clock rate leads to an increase of power consumption. Most of the power is transformed in heat. This is called the power wall.

COMPUTER & COMPUTATIONAL SCIENCES

Moore's Law for Power



Source: Fred Pollack, Intel. New Microprocessor Challenges in the Coming Generations of CMOS Technologies, MICRO32 and Transmeta

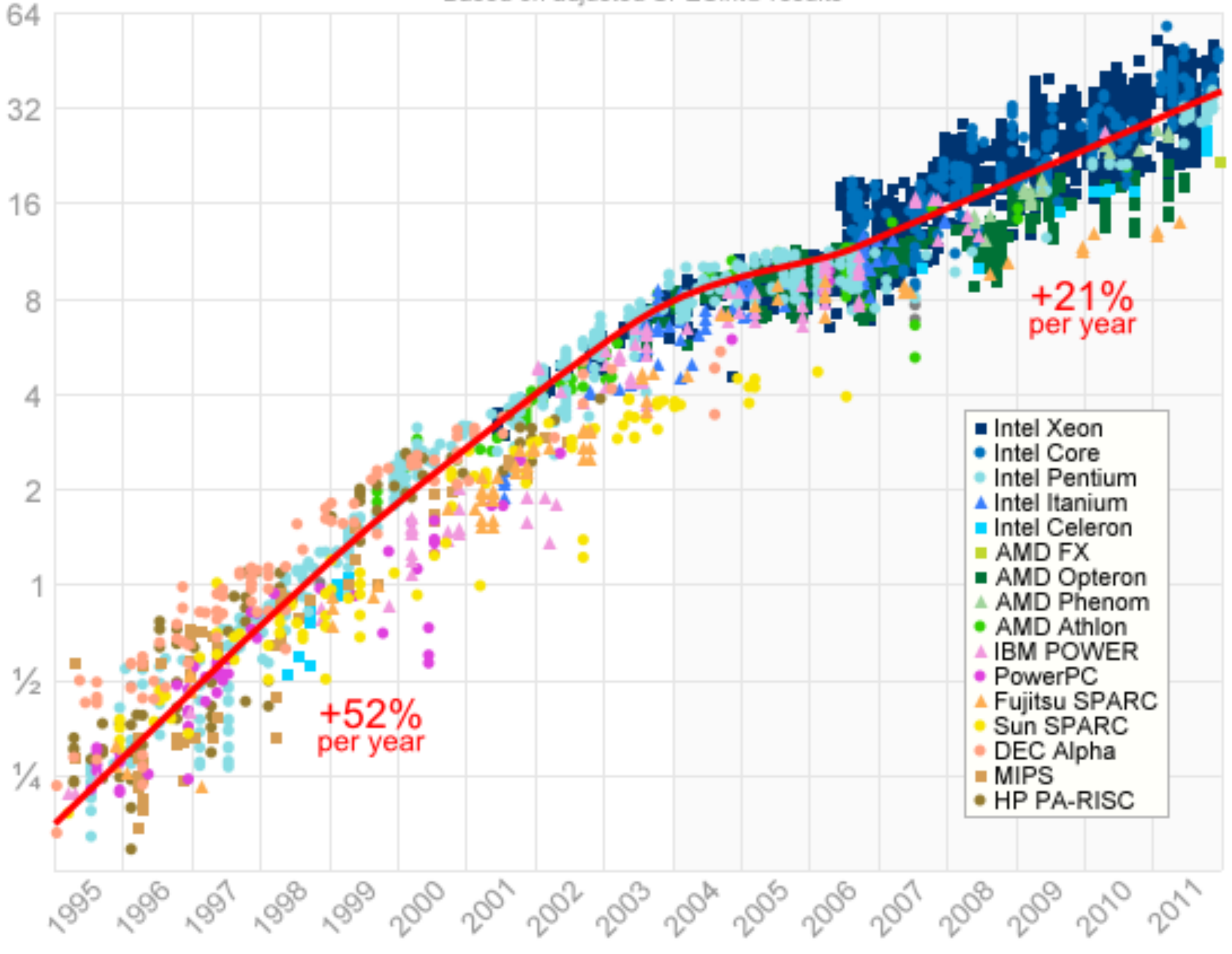
Wu FENG
feng@lanl.gov

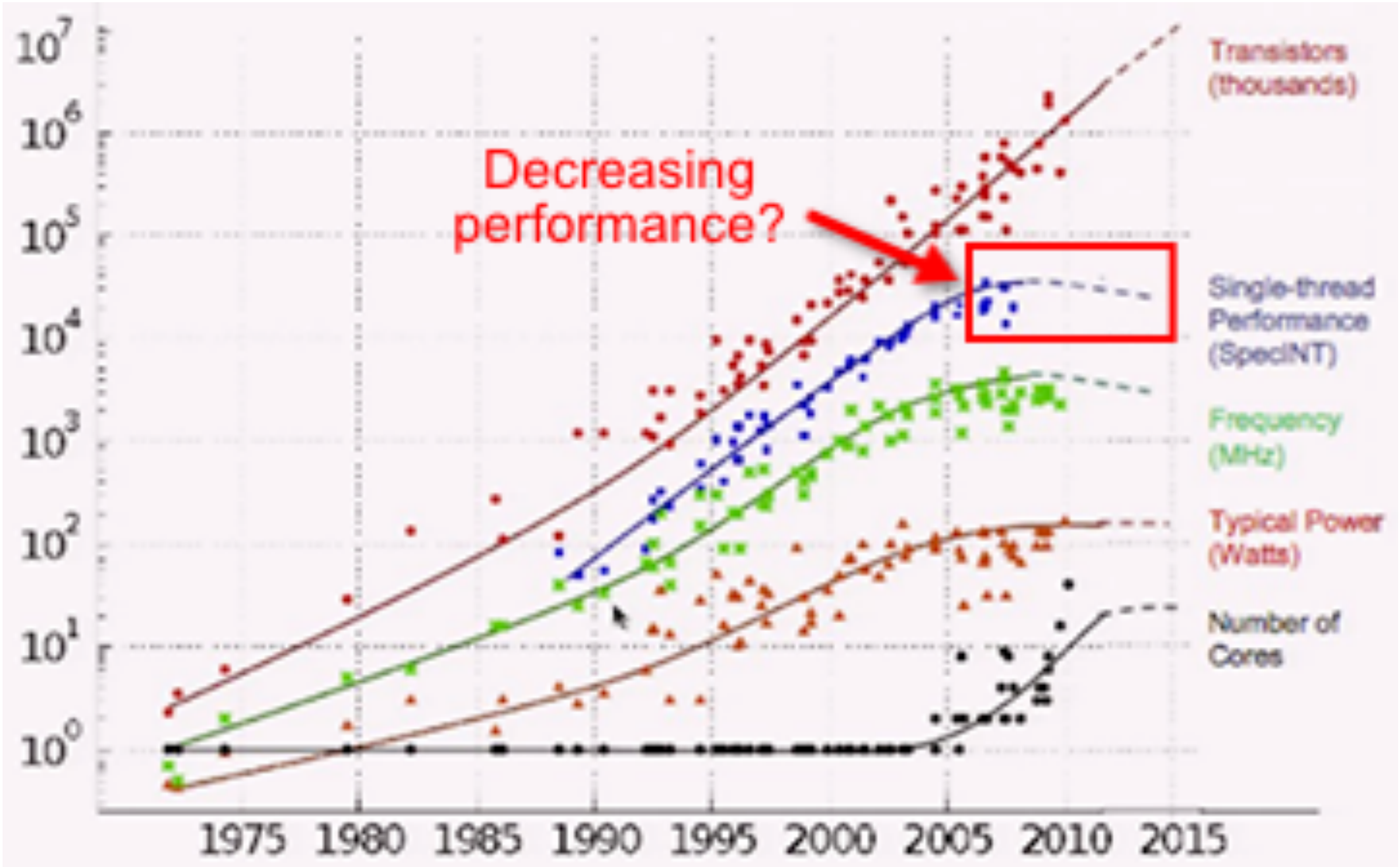
9

Los Alamos

Single-Threaded Integer Performance

Based on adjusted SPECint® results




Decreasing performance?
Transistors (thousands)
Single-thread Performance (SpecINT)
Frequency (MHz)
Typical Power (Watts)
Number of Cores
10^7
10^6
10^5
10^4
10^3
10^2
10^1
10^0
1975
1980
1985
1990
1995
2000
2005
2010
2015

Problems

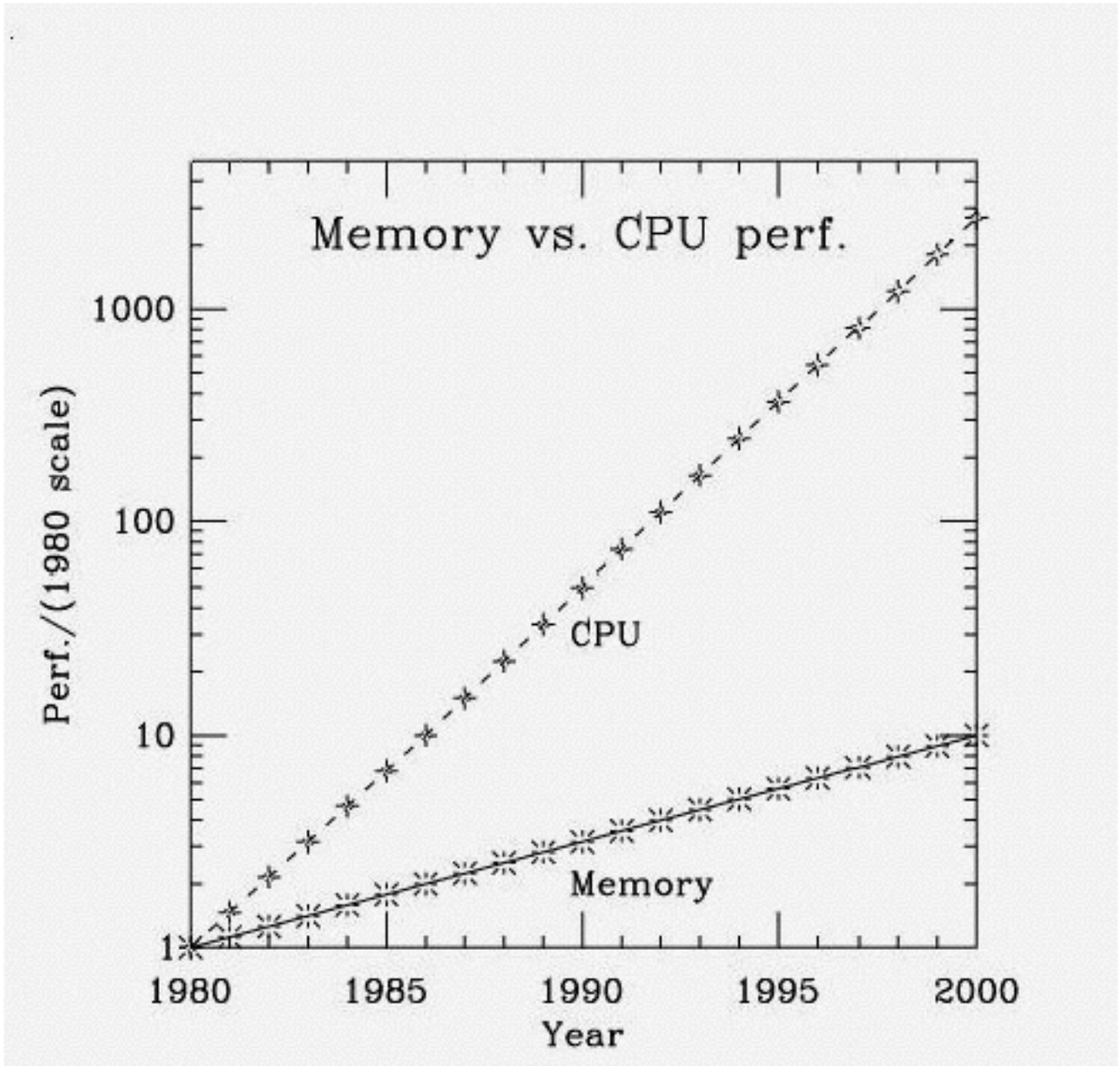
- Increase of performance based essentially on
  1. increase of clock rate, i.e., decrease of cycle time
  2. introduction of parallelism in processor through multiple functional units (ILP)
- Further multiplication of functional units is possible BUT
  1. speed of memory access did not increase according to the speed of the processors (see plot on next slide)

     Intel i486 (1990): 6–8 cycles to access data from memory

     Intel Pentium (2006): $>$ 220 cycles to access data from memory
  2. increase of numbers of transistors on same area increases packaging and power consumption.

     Sophisticated cooling necessary.

Problems (cont.)



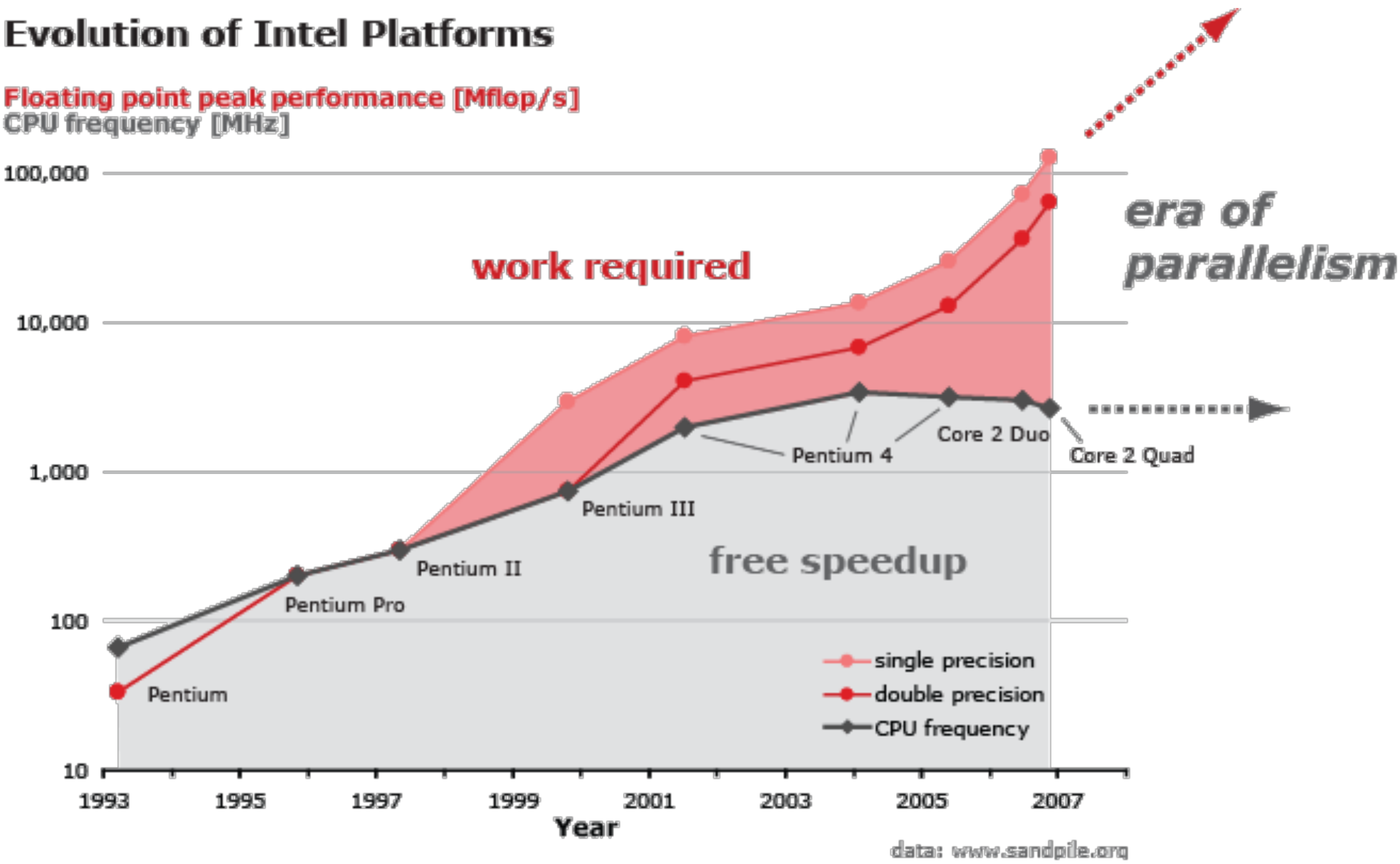
Biggest problem of high performance computing (and of parallel computing in particular):

memory access!

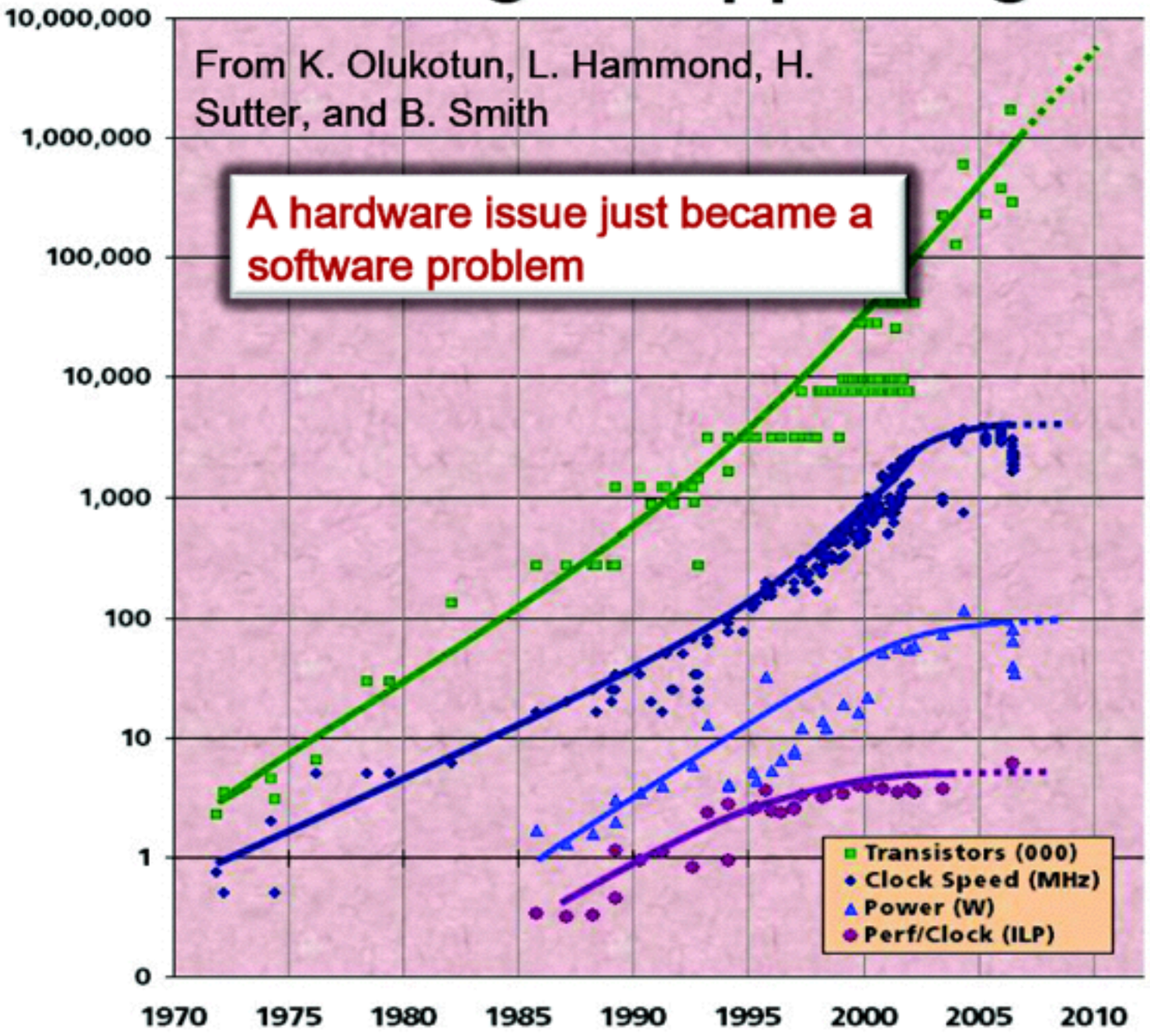
In contrast to CPU performance, memory performance doubles *in 6 years only*!

[Hennessy & Patterson, Computer Architecture, Morgan Kaufmann, 2006]

Evolution of Intel Platforms



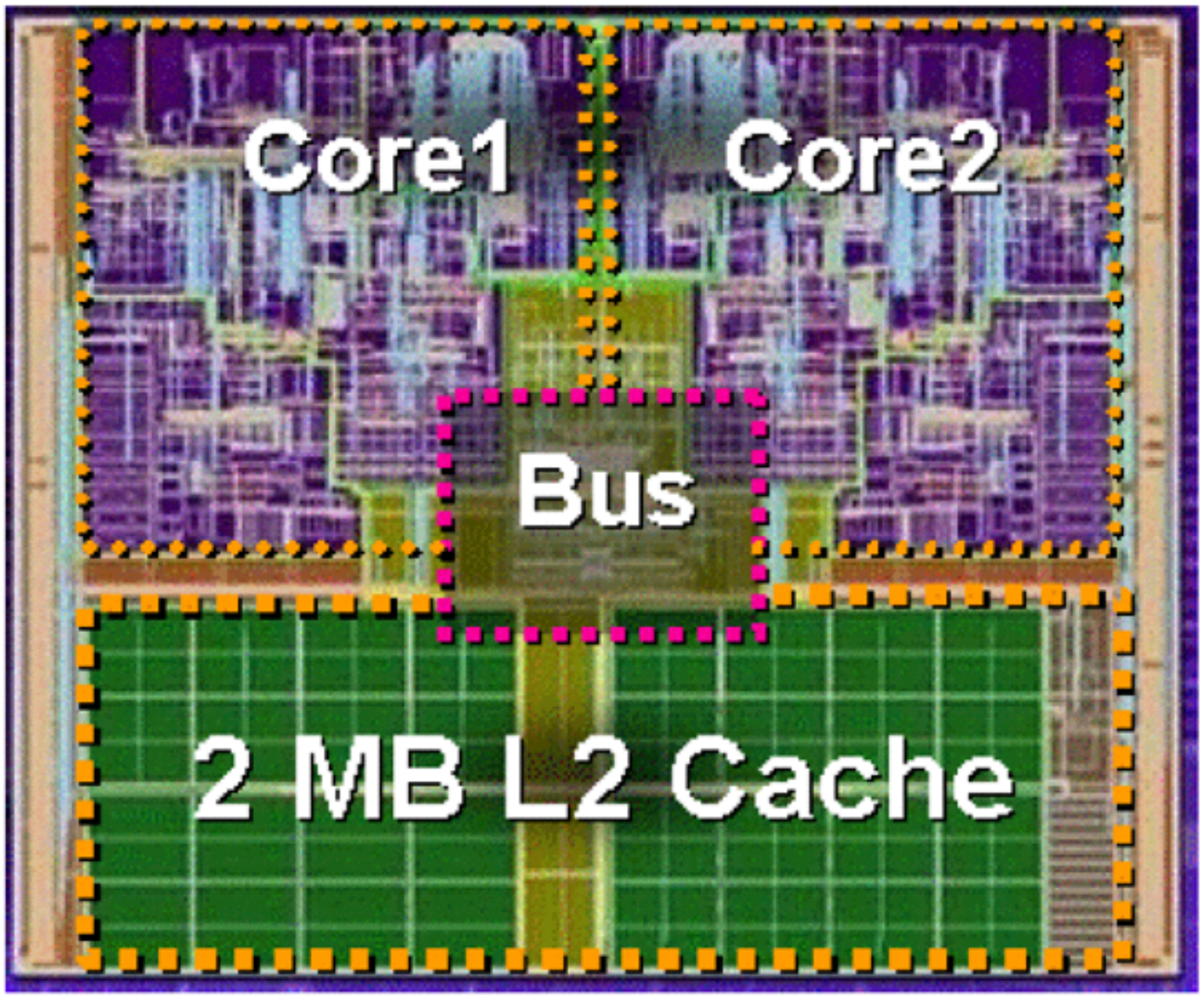
Something's Happening Here...



- In the "old days" it was: each year processors would become faster
- Today the clock speed is fixed or getting slower
- Things are still doubling every 18 -24 months
- Moore's Law reinterpretated.
  - Number of cores double every 18-24 months

8

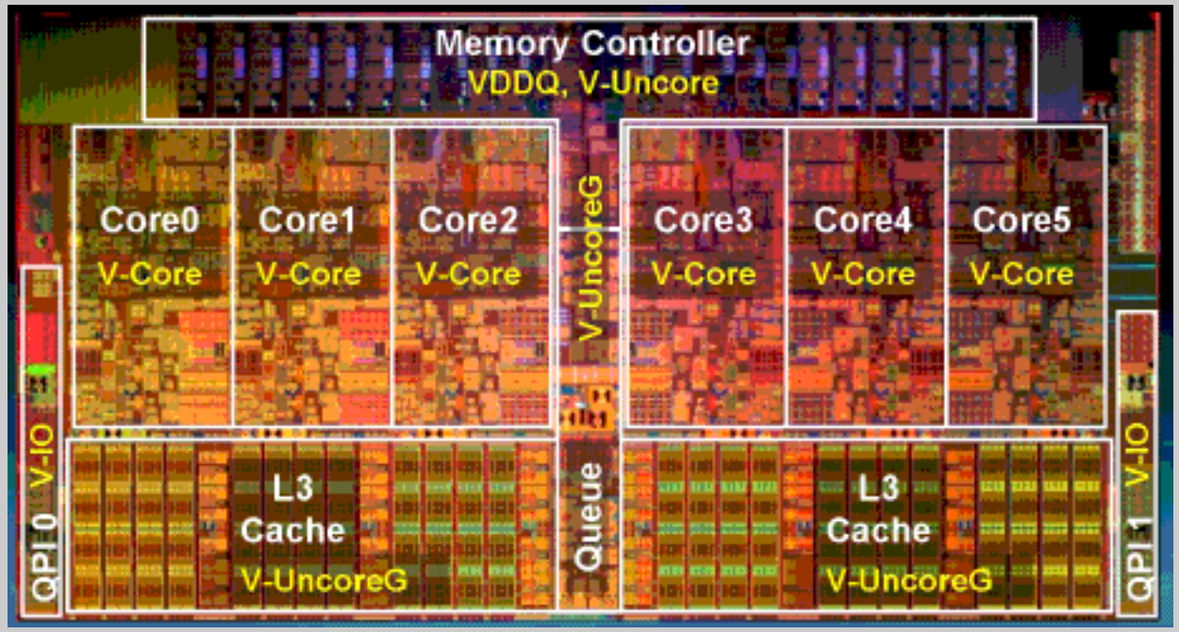
An example: the Intel Core 2 processor

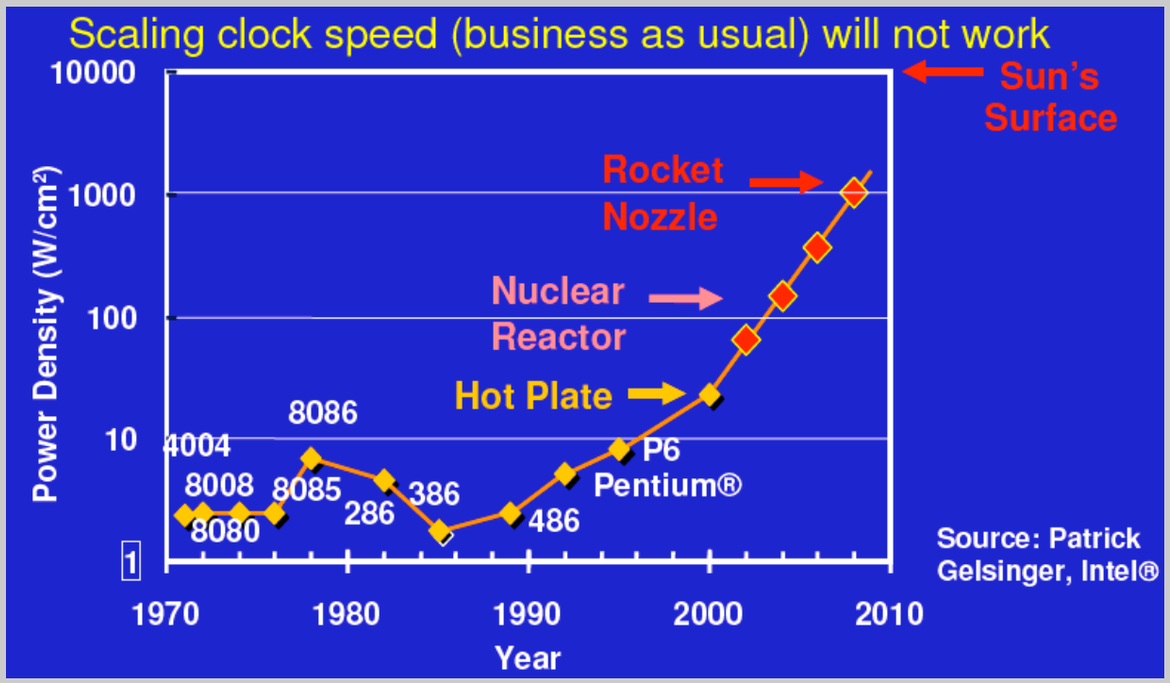


Intel Core Duo processor floor plan

See Intel Technology Journal, Volume 10, Issue 2, 2006.

Computing core: CPU / GPU





- **Always multicore**
  - ✓ power constraints
  - ✓ *Ex*: Intel Xeon L5640@2.26GHz : 6 cores / 60W: 54 GFlops
- **GPGPU**
  - ✓ optimized for vector instructions
  - ✓ *Ex*: Nvidia Tesla M2070: 448 cores / 225W

    515 Gflops (double precision)

Evolution of Computer Power/Cost

MIPS per $1000 (1998 $)

Billion

Million

1000

1

1/1000

1/Million

1/Billion

1900 1920 1940 1960 1970 1980 1990 2000 2010 2020 2030 2040 2050

Monroe Calculator
IBM Tabulator
Burroughs Class 16
Zuse-1
ASCC (Mark 1)
Colossus
ENIAC
UNIVAC I
Whirlwind
IBM 704
IBM 7090
IBM 1130
DEC PDP-10
CDC 7600
DG Eclipse
Apple II
Sun-2
IBM PC
Commodore 64
Macintosh-128K
Mac II
Gateway-486DX2/66
PowerMac 8100/80
Gateway G6-200
Mac G3/266
Dell 340/2G
IBM 650
IBM 1620
Burroughs 5000
IBM 7040
IBM 360/75
SDS 920
DG Nova
DEC-KL-10
DEC VAX 11/780
Vax 11/750
Sun-3
Mac IIfx
IBM PS/2 90
AT&T Globalyst 600
Power Tower 180e
Mac G4 dual 500

1965 Trend
1975 Trend
1985 Trend
1995 Trend

MIPS

10^9

10^6

10^3

1

10^{-3}

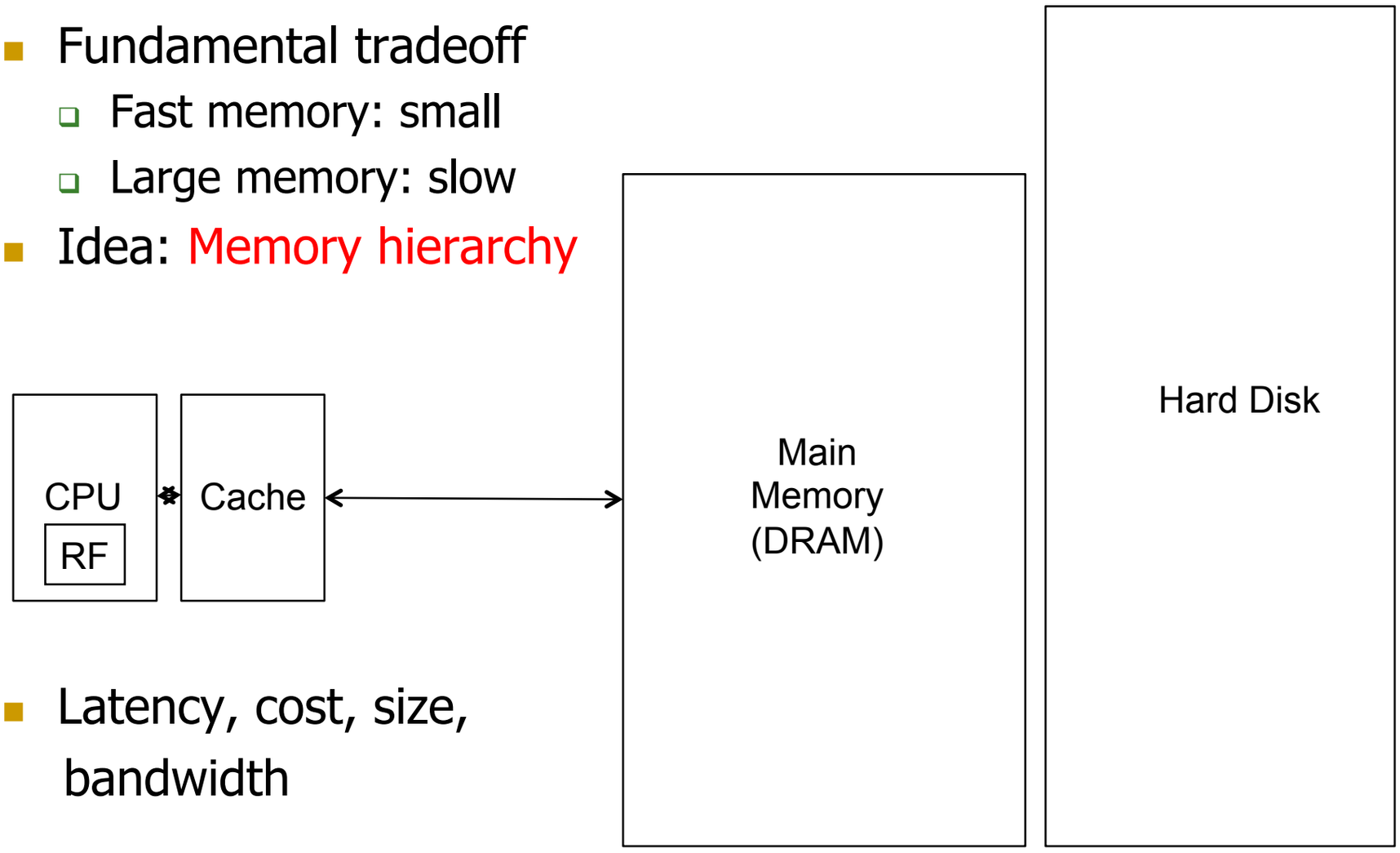
10^{-6}

Brain Equivalent	First Similar Organisms	Comparable Machines
		G4 eta 2050 (reasoning)
Human (100 kg 2 m, 100G neurons)	1 MYBP	
		G3 eta 2040 (imagination)
Monkey (3kg 60 cm, 3G neurons)	60 MYBP	
		G2 eta 2030 (adaptation)
Mouse (25g 15 cm, 60M neurons)	200 MYBP	
		G1 eta 2020 (skills)
Lizard (100g 25 cm, 2M neurons)	350 MYBP	
		Utility Robot eta 2010-2015
Guppy (30mg 1 cm, 100K neurons)	450 MYBP	
		3D perception 2000 in use 2005
		2D mapping robots 1990
Worm (0.5μg 1 mm, 300 neurons)	550 MYBP	
		SRI Shakey Stanford Cart 1970-1980
		Hopkins Beast 1960
Bacterium (0.5picogram 1 μm, 1 "neuron")	3,500 Million Years Before the Present	
		Grey Walter Tortoise 1950
Manual Calculation		

Universal Robots

Memory Hierarchy

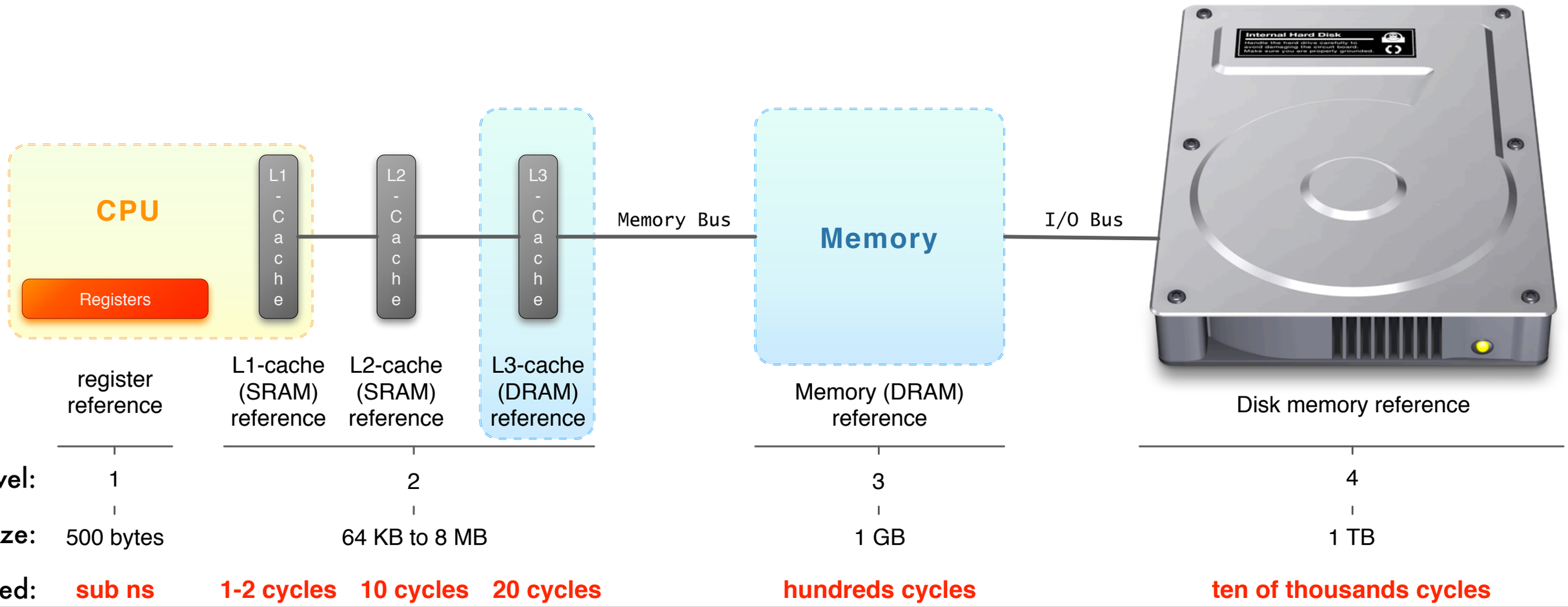
- Fundamental tradeoff
  - Fast memory: small
  - Large memory: slow
- Idea: Memory hierarchy



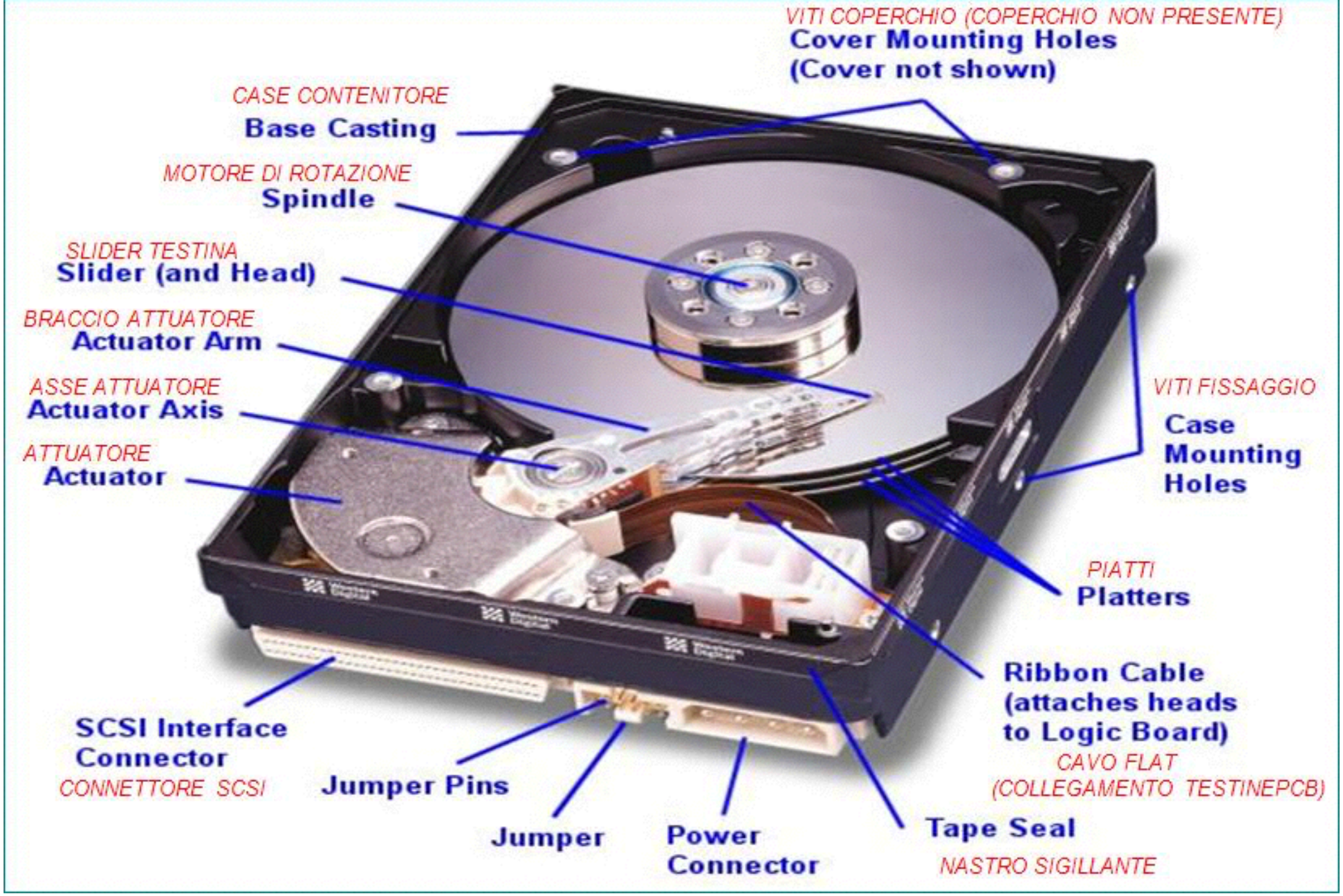
- Latency, cost, size, bandwidth

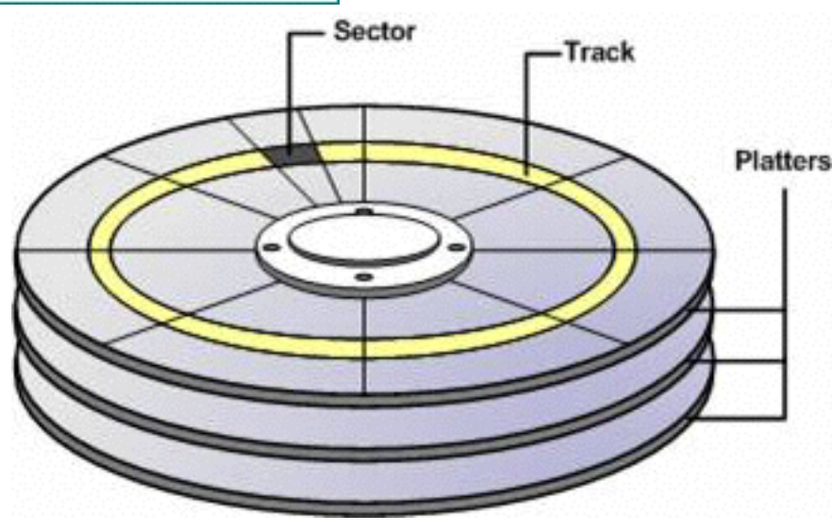
Storage / Local Memory

Larger, slower and cheaper



- **HDD**: (SATA @ 7,2 krpm) R/W: 100 MB/s; 190 IOps
- **SDD**: R/W: 560 MB/s; 85000 IOps


VITI COPERCHIO (COPERCHIO NON PRESENTE)
Cover Mounting Holes
(Cover not shown)
CASE CONTENITORE
Base Casting
MOTORE DI ROTAZIONE
Spindle
SLIDER TESTINA
Slider (and Head)
BRACCIO ATTUATORE
Actuator Arm
ASSE ATTUATORE
Actuator Axis
ATTUATORE
Actuator
VITI FISSAGGIO
Case
Mounting
Holes
PIATTI
Platters
Ribbon Cable
(attaches heads
to Logic Board)
CAVO FLAT
(COLLEGAMENTO TESTINEPCB)
SCSI Interface
Connector
CONNETTORE SCSI
Jumper Pins
Jumper
Power
Connector
Tape Seal
NASTRO SIGILLANTE

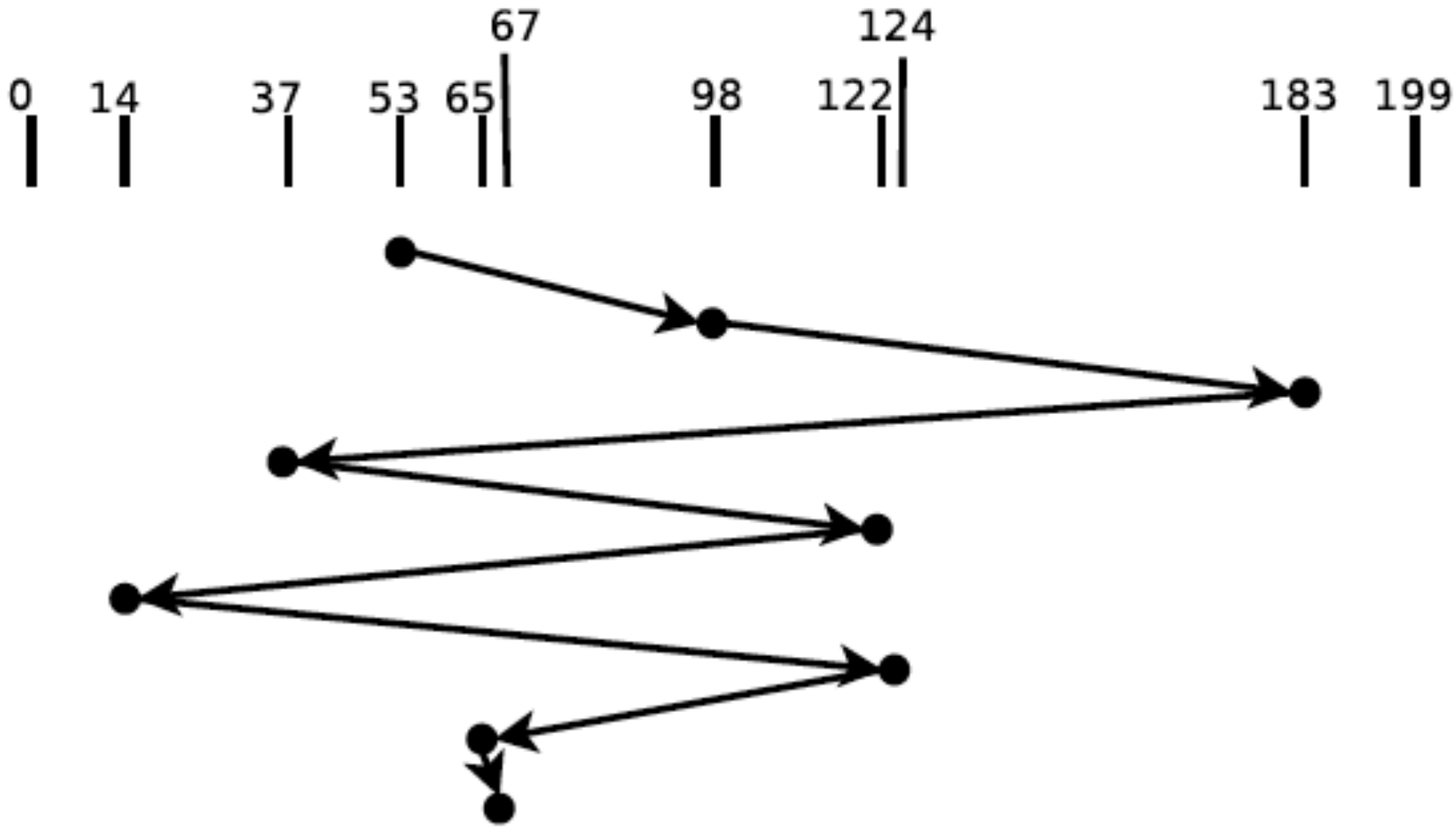

Sector
Track
Platters

Disk scheduling

- Model: queued requests to read/write disk blocks
- How should the OS schedule the requests so that they complete as quickly as possible?
  - $\Longrightarrow$ Minimize seek time (head movement)
- In theory, we could also try to minimize rotational latency
  - Hard to do, because the OS generally doesn't know the real disk geometry

FCFS: First Come, First Served

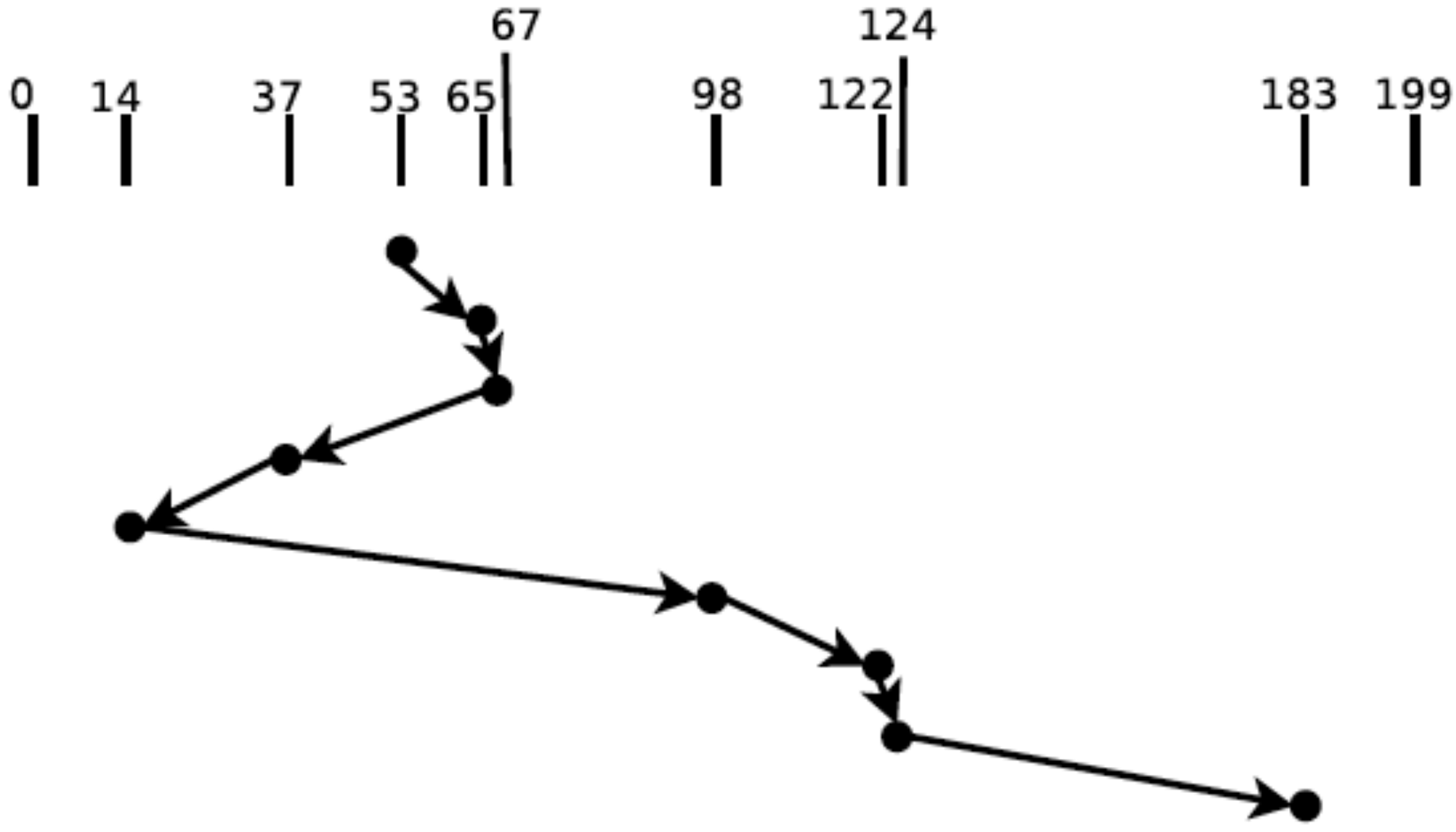
Lots of unnecessary head movement—can do better



Start: 53, Requests: 98, 183, 37, 122, 14, 124, 65, 67

SSTF: Shortest Seek Time First

Better, but requests to far cylinders may be starved

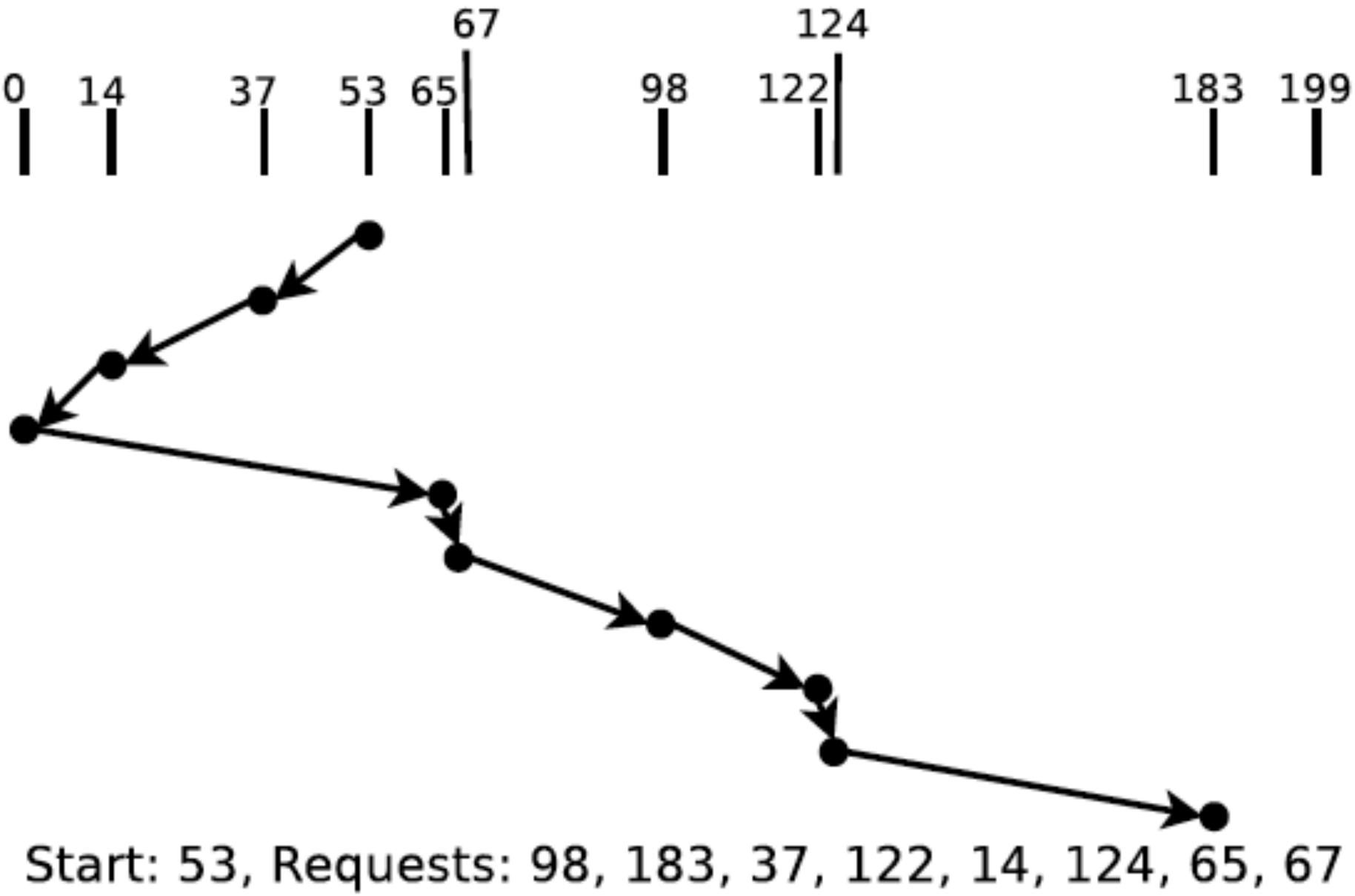


Start: 53, Requests: 98, 183, 37, 122, 14, 124, 65, 67

SCAN (Elevator algorithm)

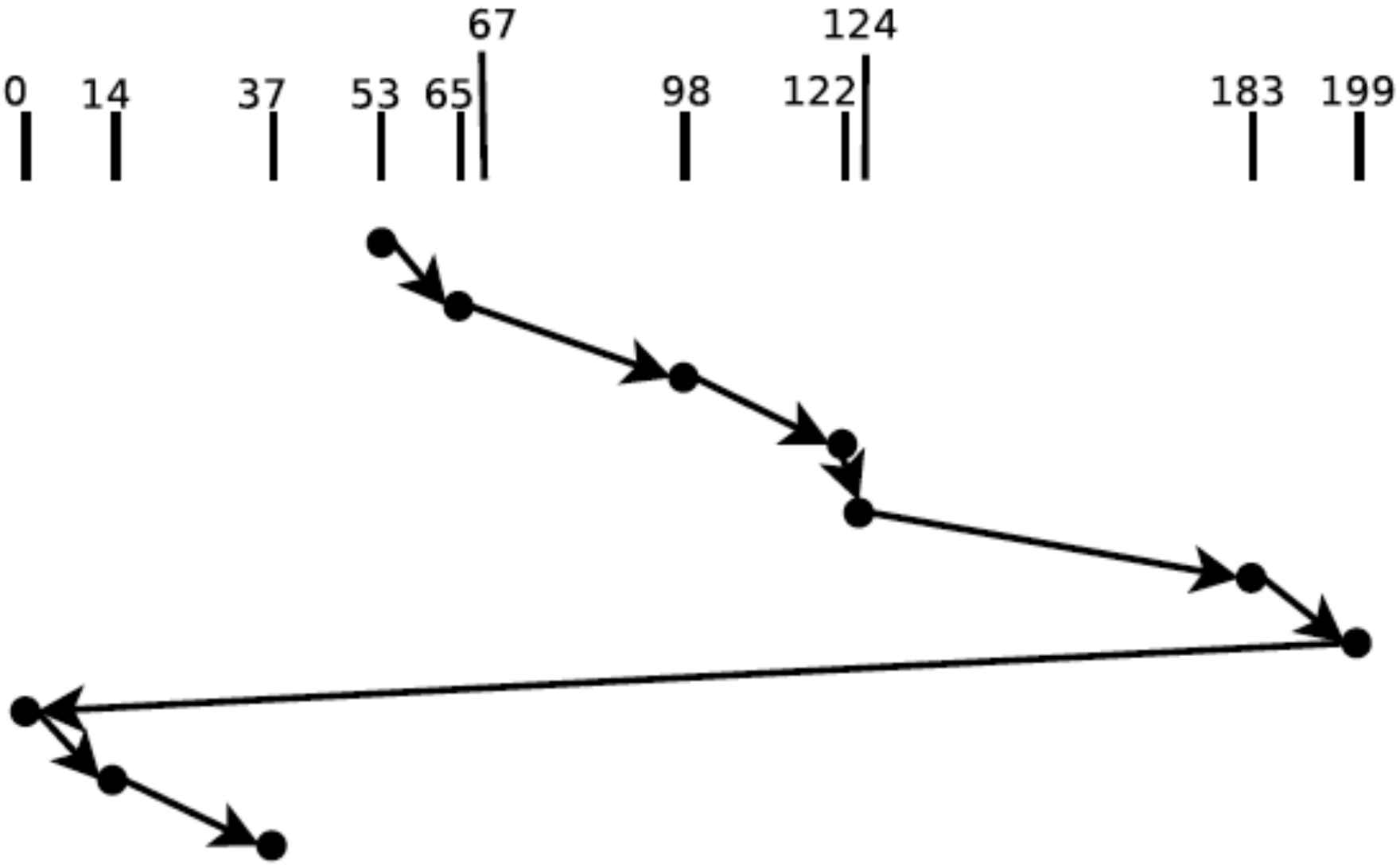
Move head back and forth across disk

Less prone to starvation, but outer cylinders wait longer



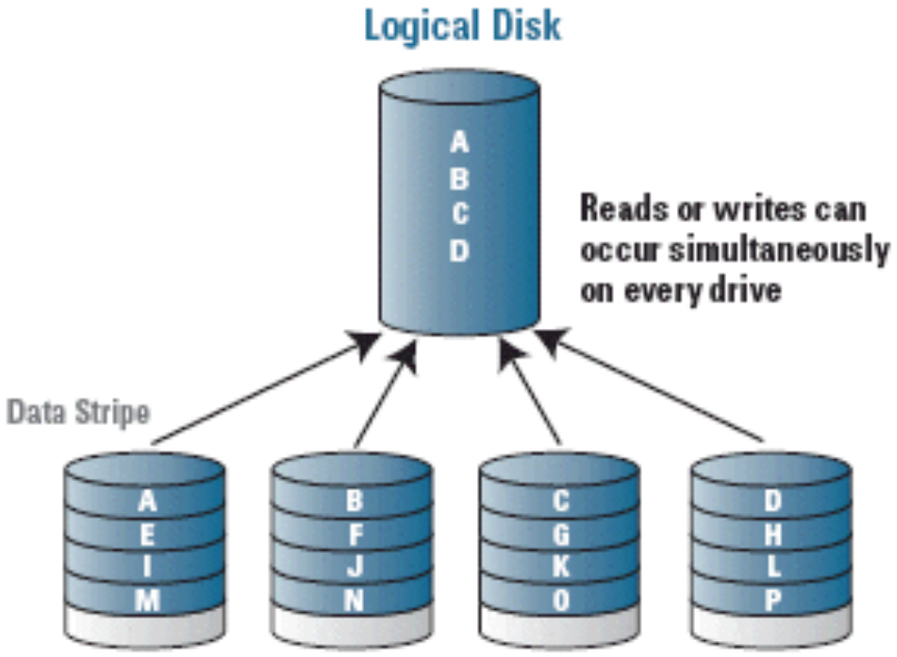
C-SCAN ("Circular SCAN")

Only handle requests in one direction, then wrap
All cylinders serviced with uniform frequency

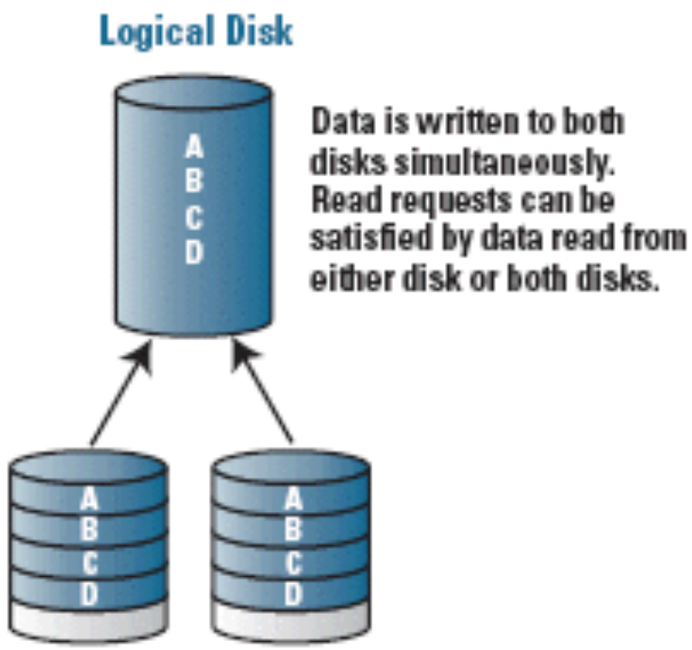


Start: 53, Requests: 98, 183, 37, 122, 14, 124, 65, 67

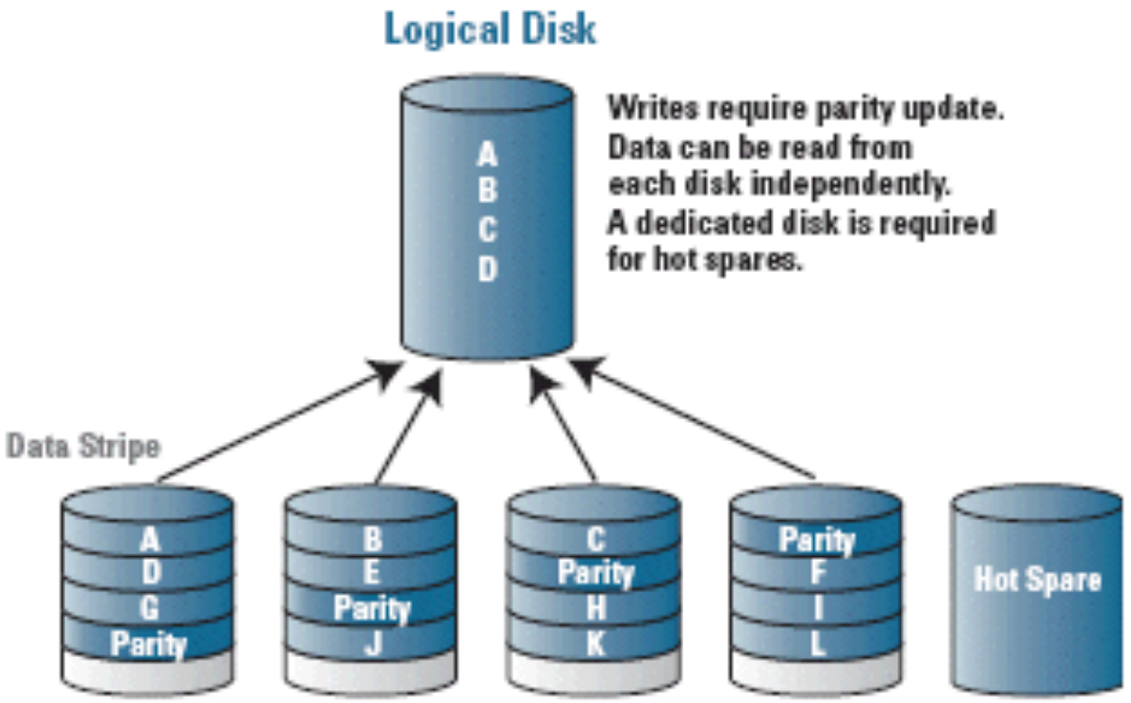
RAID 0 (Striping)



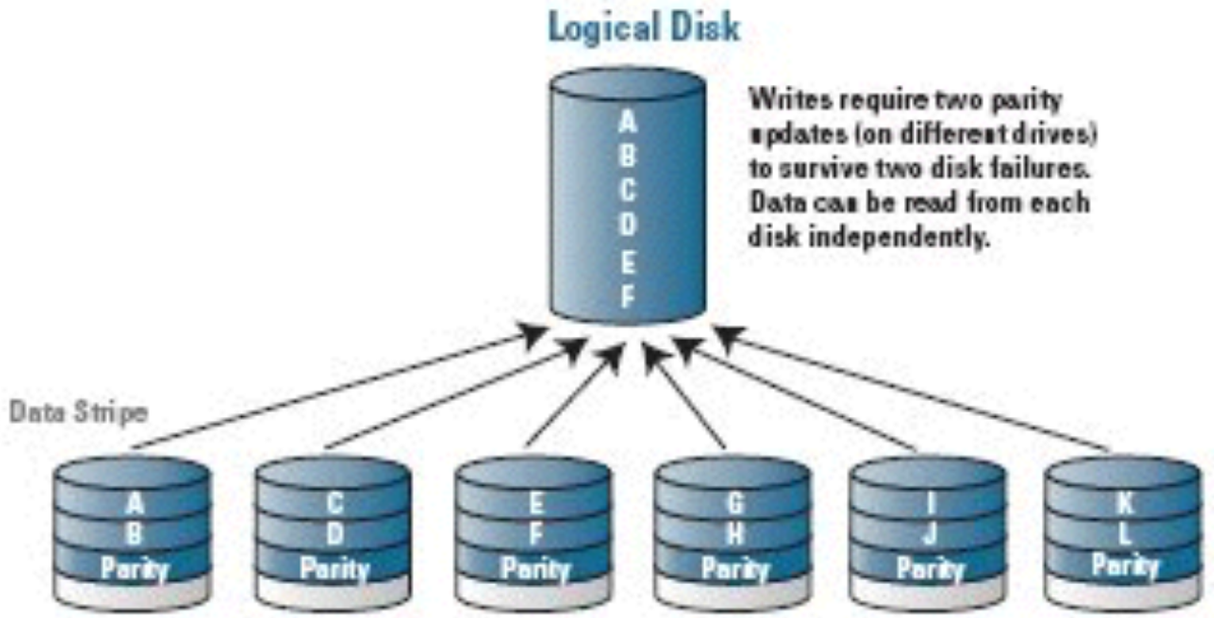
RAID 1 (Mirroring)



RAID 5 (Striping with parity)

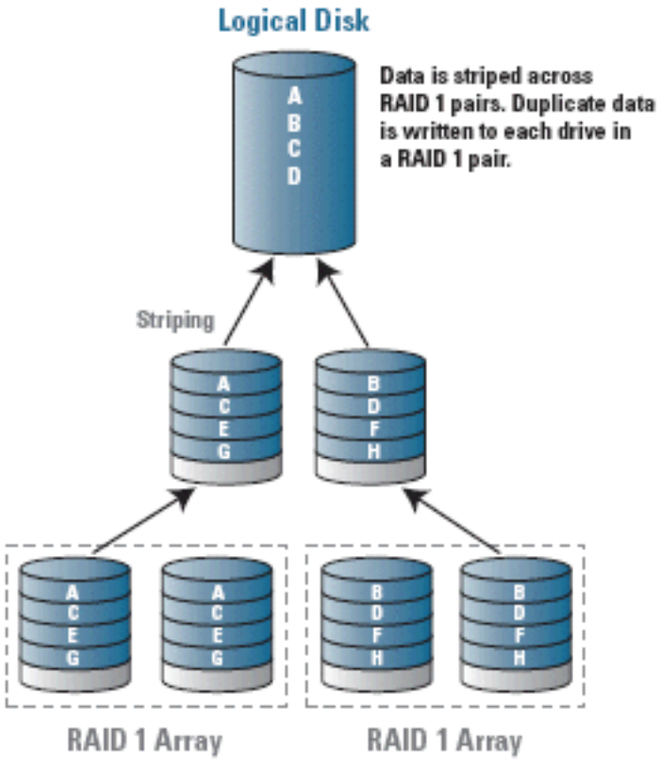


RAID 6 (Striping with dual parity)



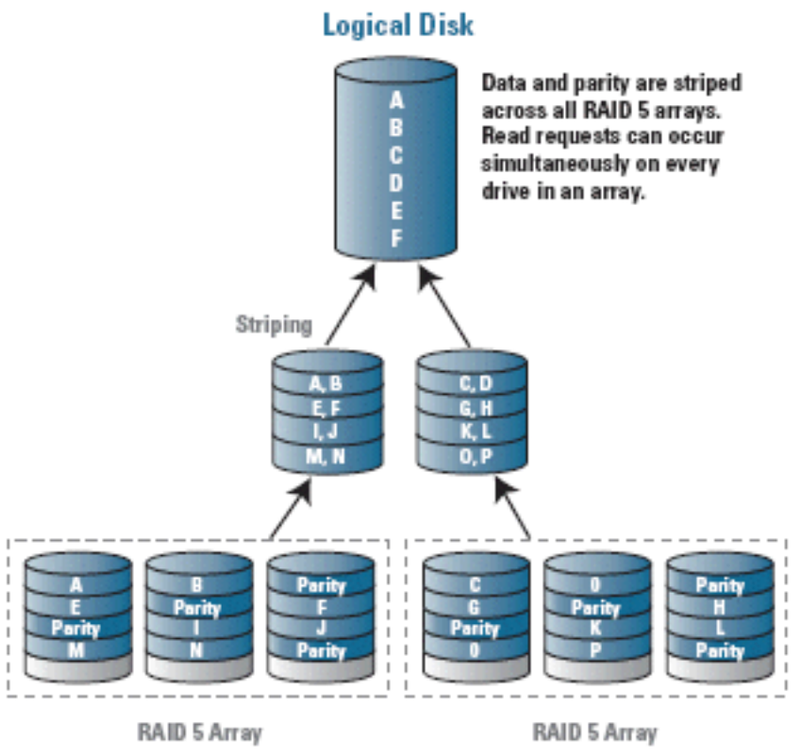
http://www.adaptec.com/en-us/_common/compatibility/_education/raid_level_compar_wp.htm

RAID 10 (Striping and mirroring)



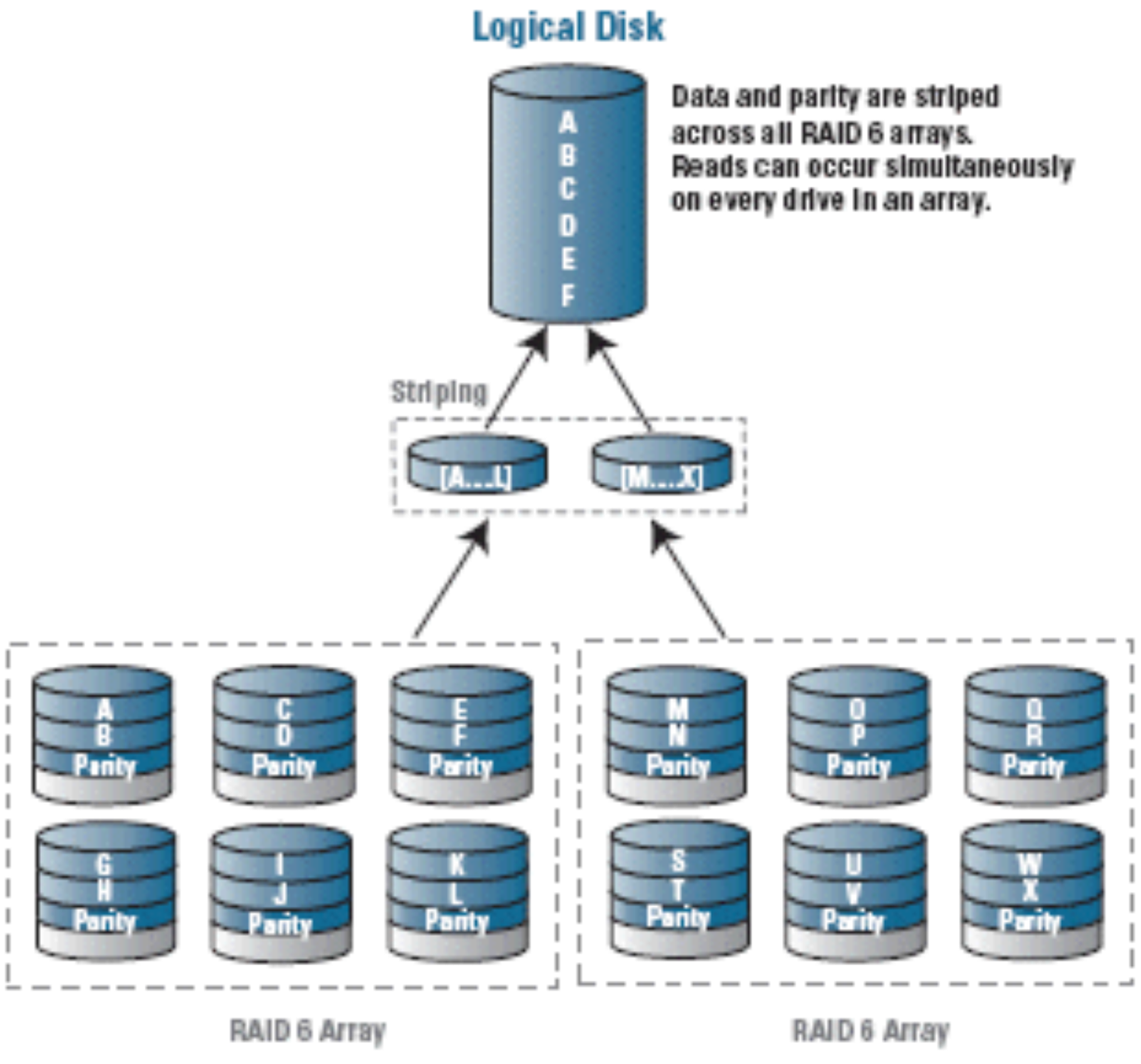
Combines RAID 0 striping and RAID 1 mirroring. This level provides the improved performance of striping while still providing the redundancy of mirroring.
RAID 10 is the result of forming a RAID 0 array from two or more RAID 1 arrays. This RAID level provides fault tolerance - up to one disk of each sub-array may fail without causing loss of data.
Usable capacity of RAID 10 is 50% of available disk drives.

RAID 50 (Striping)



Combines multiple RAID 5 sets with RAID 0 (striping). Striping helps to increase capacity and performance without adding disks to each RAID 5 array (which will decrease data availability and could impact performance when running in a degraded mode).
RAID 50 comprises RAID 0 striping across lower-level RAID 5 arrays. The benefits of RAID 5 are gained while the spanned RAID 0 allows the incorporation of many more disks into a single logical drive. Up to one drive in each sub-array may fail without loss of data. Also, rebuild times are substantially less then a single large RAID 5 array.
Usable capacity of RAID 50 is between 67% - 94%, depending on the number of data drives in the RAID set.

RAID 60 (Striping and striping with dual party)



Combines multiple RAID 6 sets with RAID 0 (striping). Dual parity allows the failure of two disks in each RAID 6 array. Striping helps to increase capacity and performance without adding disks to each RAID 6 array (which would decrease data availability and could impact performance in degraded mode).

Features	RAID 0	RAID 1	RAID 1E	RAID 5	RAID 5EE
Minimum # Drives	2	2	3	3	4
Data Protection	No Protection	Single-drive failure	Single-drive failure	Single-drive failure	Single-drive failure
Read Performance	High	High	High	High	High
Write Performance	High	Medium	Medium	Low	Low
Read Performance	N/A	Medium	High	Low	Low
Write Performance	N/A	High	High	Low	Low
Capacity Utilization	100%	50%	50%	67% - 94%	50% - 88%
Typical Applications	High End Workstations, data logging, real-time rendering, very transitory data	Operating System, transaction databases	Operating system, transaction databases	Data warehousing, web serving, archiving	Data warehousing, web serving, archiving

http://www.adaptec.com/en-us/_common/compatibility/_education/raid_level_compar_wp.htm

Features	RAID 6	RAID 10	RAID 50	RAID 60
Minimum # Drives	4	4	6	8
Data Protection	Two-drive failure	Up to one disk failure in each sub-array	Up to one disk failure in each sub-array	Up to two disk failure in each sub-array
Read Performance	High	High	High	High
Write Performance	Low	Medium	Medium	Medium
Read Performance (degraded)	Low	High	Medium	Medium
Write Performance (degraded)	Low	High	Medium	Low
Capacity Utilization	50% - 88%	50%	67% - 94%	50% - 88%
Typical Applications	Data archive, backup to disk, high availability solutions, servers with large capacity requirements	Fast databases, application servers	Large databases, file servers, application servers	Data archive, backup to disk, high availability solutions, servers with large capacity requirements

Data Management / File System (FS)

- Logical manner to store, manipulate and access data
- **Disk file systems**
  - ✓ FAT32, NTFS, HFS, ext3, ext4, xfs...
- **Network file systems**
  - ✓ NFS, SMB
- **Distributed and/or Parallel file systems**
  - ✓ data are stripped over multiple servers for high performance
  - ✓ generally add robust failover and recovery mechanisms
    - Lustre, GPFS, FhGFS, GlusterFS
    - GFS (Google), HDFS (Hadoop), OneFS (Isilon) etc.

Interconnect

- **Latency**

  ✓ time to send a minimal (0 byte) message from A to B

- **Bandwidth**

  ✓ max amount of data communicated per unit of time

Technology	Effective Bandwidth		Latency
Gigabit Ethernet	1 Gb/s	125 MB/s	40μs to 300μs
Myrinet (Myri-10G)	9.6 Gb/s	1.2 GB/s	2.3μs
10 Gigabit Ethernet	10 Gb/s	1.25 GB/s	4μs to 5μs
Infiniband QDR	40 Gb/s	5 GB/s	1.29μs to 2.6μs
SGI NUMAlink	60 Gb/s	7.5 GB/s	1μs

A little experiment

- How bad can you write code?

- How fast is a little laptop today?

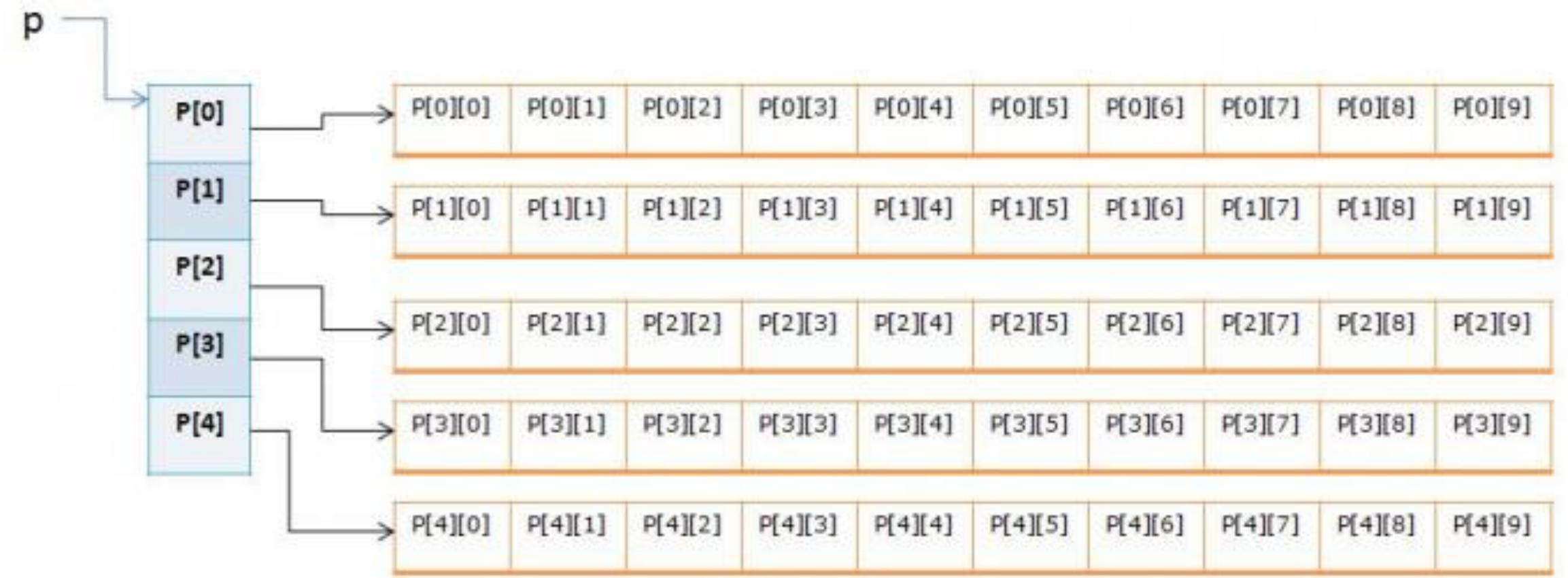
How many MOPS can we do?
Is the performance dependent of the loop size?

Basis=16, Max =19

```
Do Length = 0, max
    LoopSize = BASIS * (2**Length)
    LoopIterations = 2**(max - Length)
    Do Iterations = 1, LoopIterations
        DO X = 1, LoopSize
            ARRAY(X) = ARRAY(X) + 3
        ENDDO
    ENDDO
ENDDO
```

```
DO R = 1, ROWS
      DO C = 1, LENGTH
            ARRAY(R, C) = ARRAY(R, C) + 3
      ENDDO
ENDDO
```

```
DO C=1, COLS
      DO R=1, LENGTH
            ARRAY(R, C)= ARRAY(R, C) + 3
      ENDDO
ENDDO
```


p
P[0]
P[1]
P[2]
P[3]
P[4]
P[0][0] P[0][1] P[0][2] P[0][3] P[0][4] P[0][5] P[0][6] P[0][7] P[0][8] P[0][9]
P[1][0] P[1][1] P[1][2] P[1][3] P[1][4] P[1][5] P[1][6] P[1][7] P[1][8] P[1][9]
P[2][0] P[2][1] P[2][2] P[2][3] P[2][4] P[2][5] P[2][6] P[2][7] P[2][8] P[2][9]
P[3][0] P[3][1] P[3][2] P[3][3] P[3][4] P[3][5] P[3][6] P[3][7] P[3][8] P[3][9]
P[4][0] P[4][1] P[4][2] P[4][3] P[4][4] P[4][5] P[4][6] P[4][7] P[4][8] P[4][9]

HPC Platforms @ UL : Overview (as of Nov. 2014)

UNIVERSITÉ DU LUXEMBOURG

HPC @ UL

http://hpc.uni.lu

Organization Chart

Rectorate
University of Luxembourg

F. Leprevost

Steering Commitee

SIU | UL Faculties | LCSB | SnT

HPC Users

President: F. Leprevost
P. Bouvry, S. Varrette, V. Muller, L. Briand, L. Wirtz, B. Peters, R. Schneider

UL HPC Department

HPC Management
(S. Varrette, P. Bouvry)

IT Purchases

SIU

Computer Help Information Point

First line support

Users

Bug tracking

Helpdesk, docs

Platform Maintenance

Research & Development

Second line support

Management (puppet)

Storage (Lustre, GPFS)

Virtualization (Xen)

Network (Cisco)

Cloud (OpenStack...)

Parallel computing (MPI, TBB, CUDA)

Deployment (FAI...)

Monitoring

Expertises

HPC/Cloud Marketing & Sales

Accounting / platform usage

Marketing strategy

Licence management

Internal/external customers

Sysadmins Team: S. Varrette, H. Cartiaux, V. Plugaru, J-F. Le Fillâtre

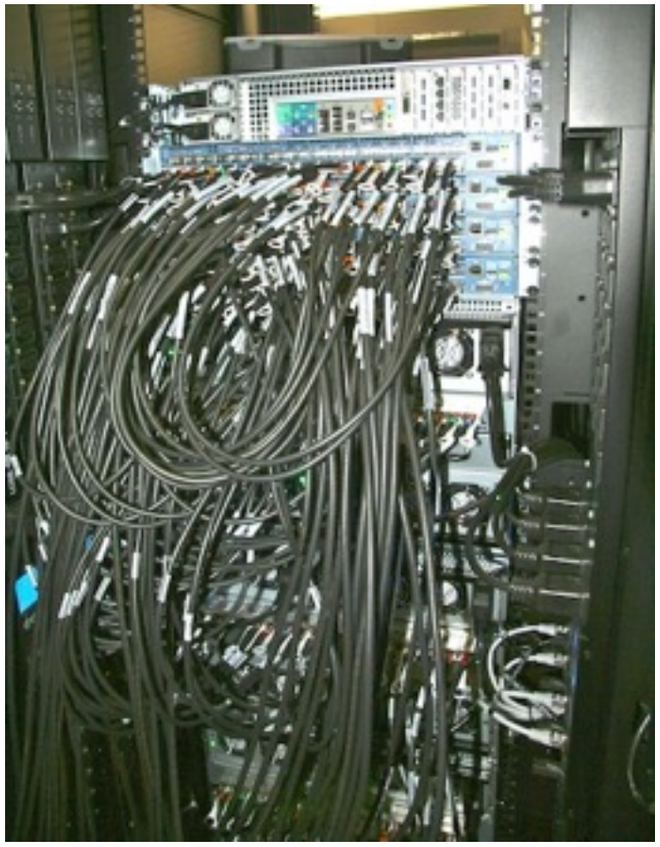
UL HPC Platform

http://hpc.uni.lu

- 2 geographical sites, 3 server rooms
- 4 admins, 4 clusters, ~281 users
  - ✓ **404** nodes, **4316 cores** (**49.92 TFlops**)
    - 18 dual [GP]GPU / visualization nodes
  - ✓ Cumul. *shared* raw storage: **3,13 PB**
    - user/research data storage: 1,42 PB
    - backup: 1,71 PB
- > **6,21 M€** HW investment so far
- Mainly Open-Source software stack
  - ✓ Debian, SSH, OpenLDAP, Puppet, FAI...

UL HPC Platform

http://hpc.uni.lu



Problem - Data Movement / Life Cycle

Access

Scientists

Storage

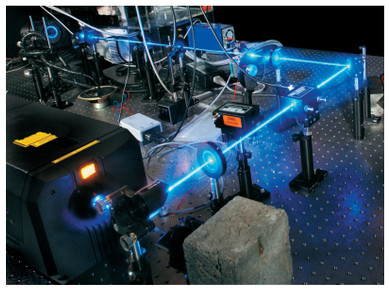
Disk

Backup / Archive

FedEx !?!

Other Centers, ext. Labs / Customers

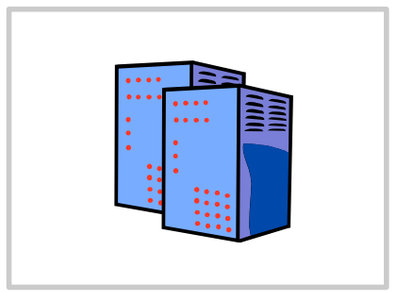
Data acquisition



Instruments

Data processing

HPC



Servers & DBs

Network

Data Center

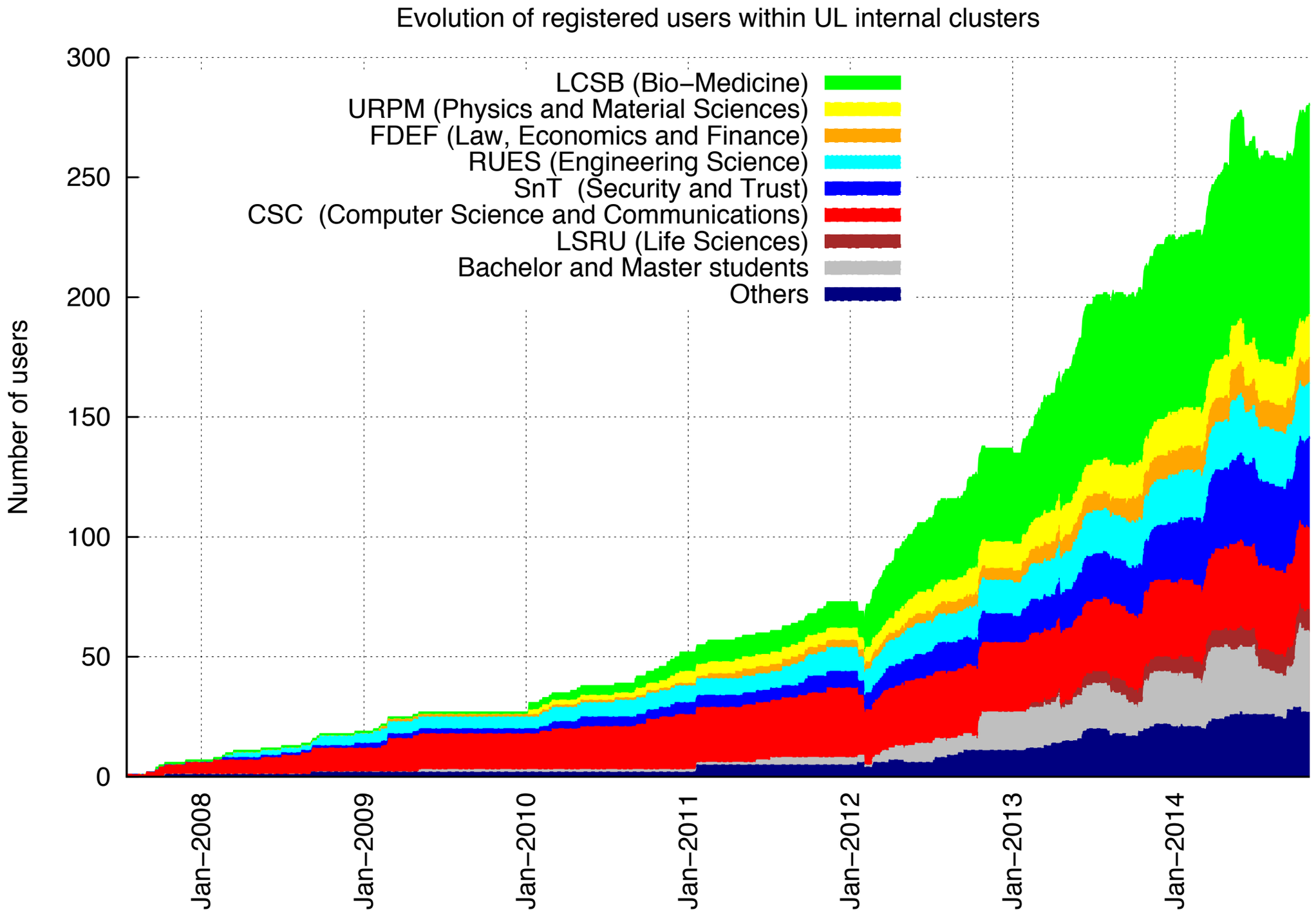
https://hpc.uni.lu

We currently operate a total of 371 computing nodes (3908 cores, 43.751 TFlops) and a shared storage capacity of 934.4 TB (+ 1064 TB for backup).

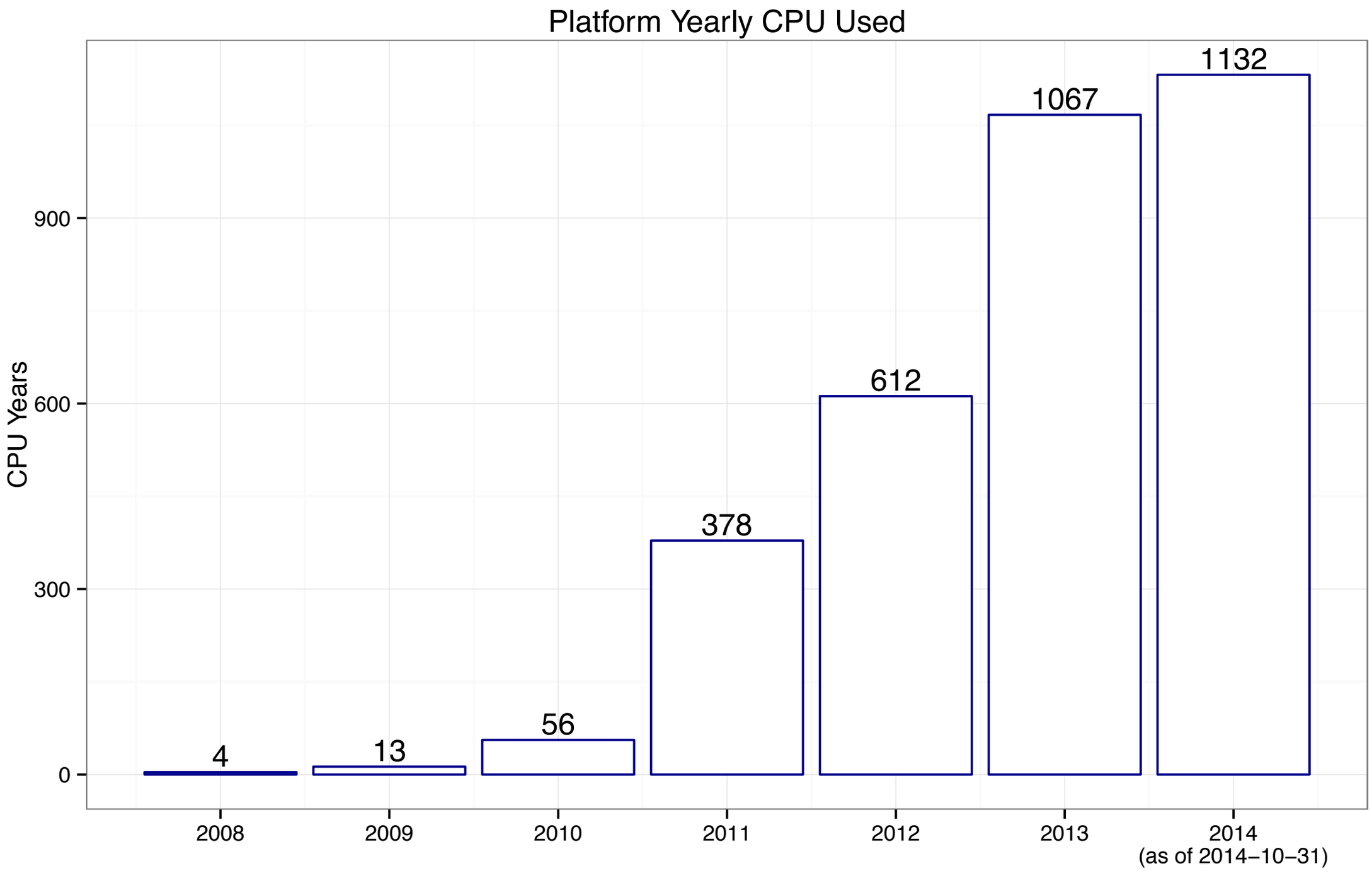
We are just adding another PetaByte

- https://hpc.uni.lu/gaia/ganglia/

UL HPC Users


Evolution of registered users within UL internal clusters
Number of users
300
250
200
150
100
50
0
LCSB (Bio–Medicine)
URPM (Physics and Material Sciences)
FDEF (Law, Economics and Finance)
RUES (Engineering Science)
SnT (Security and Trust)
CSC (Computer Science and Communications)
LSRU (Life Sciences)
Bachelor and Master students
Others
Jan–2008
Jan–2009
Jan–2010
Jan–2011
Jan–2012
Jan–2013
Jan–2014

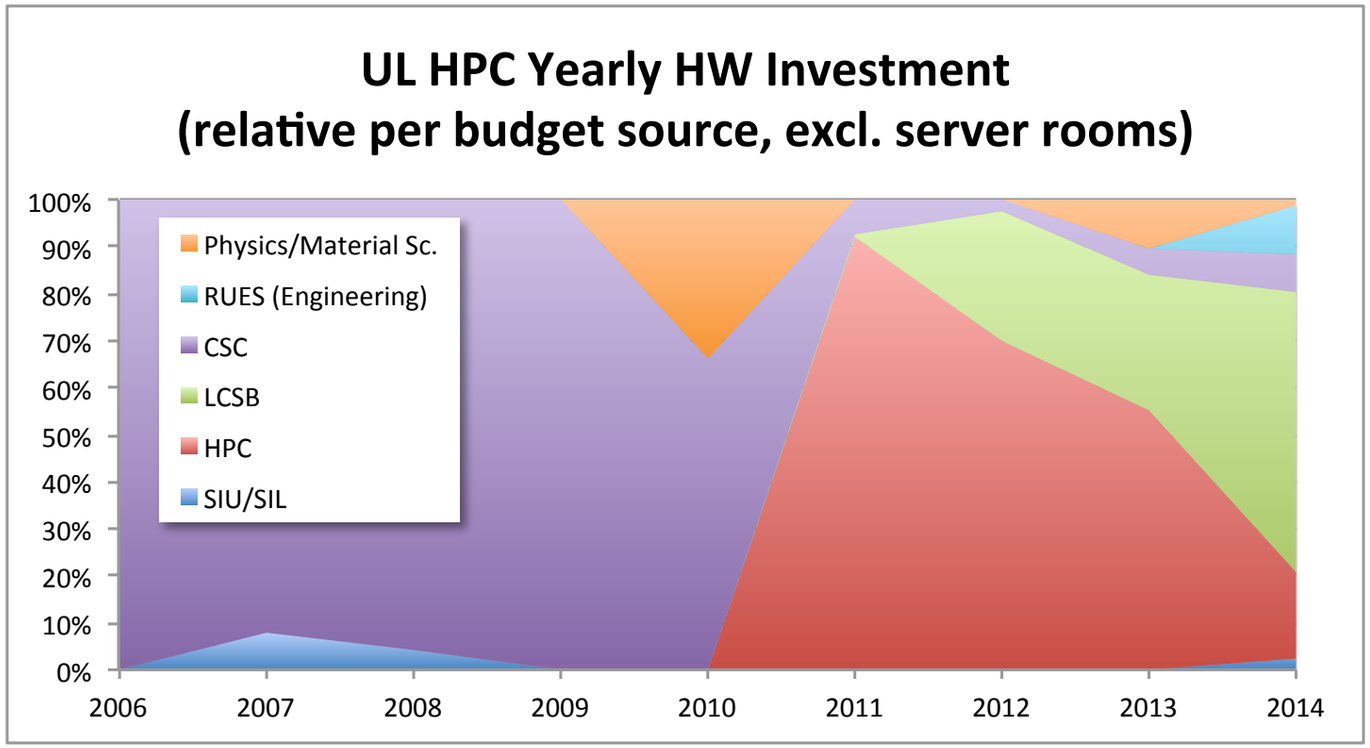
UL HPC Usage

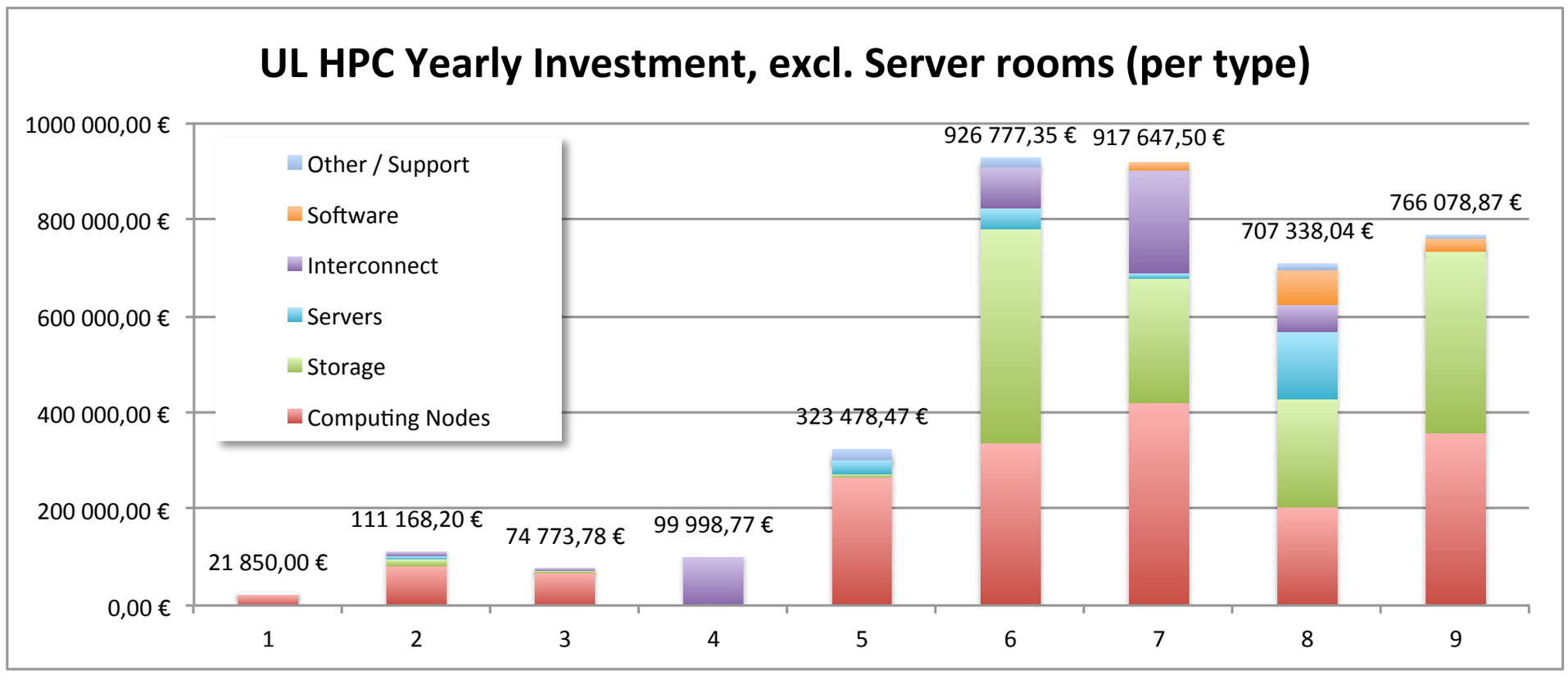


- **Used** computing effort in 2014: **1132 CPUYears** (→ oct. 31th)

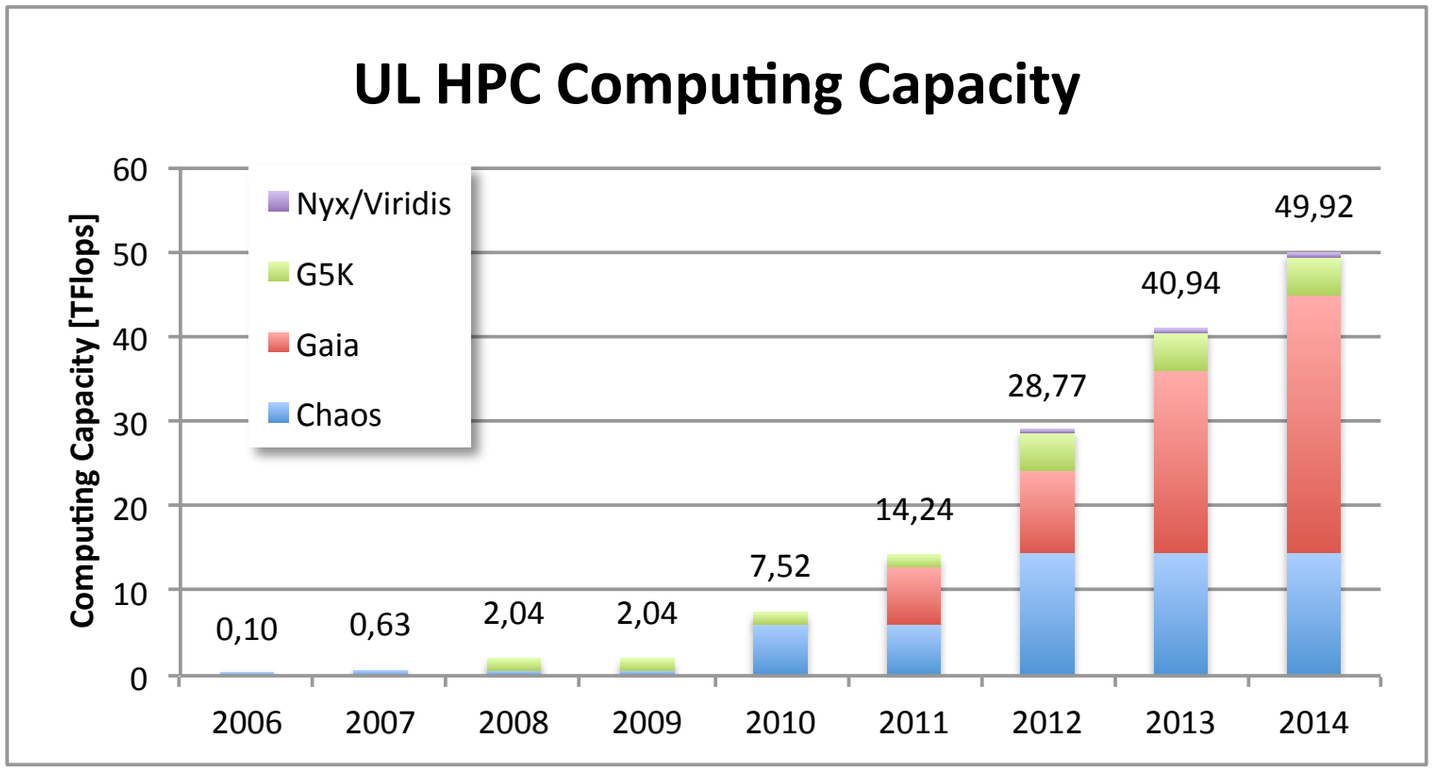
UL HPC investment

- Cumul. HW investment
  - 6.2 M€
  - **3.9 M€** excl. server rooms

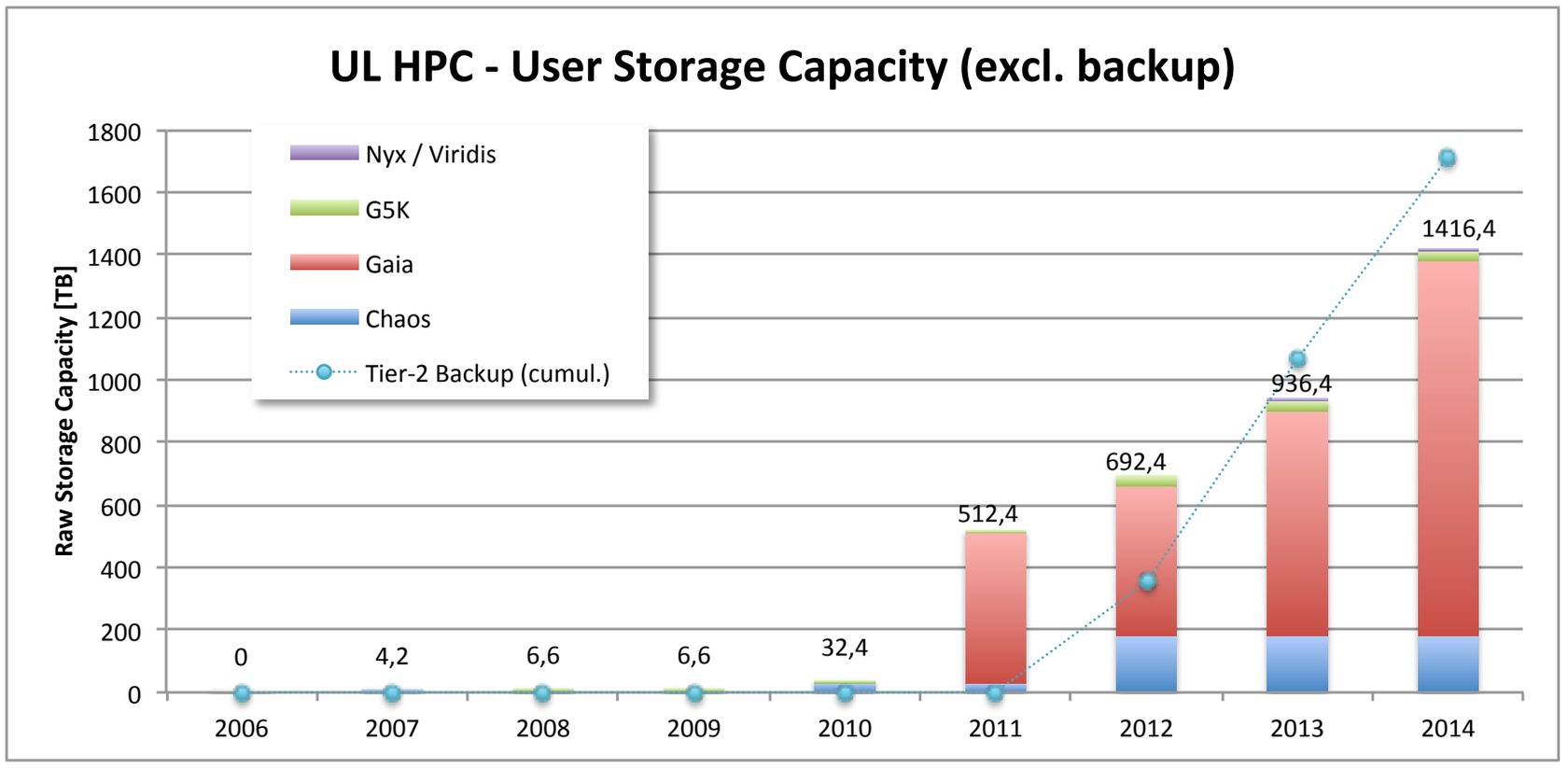




UL HPC Evolution



+ 1.3 PB arriving
1.Quarter 2015



LCSB internal project storage

650 TB project space

Certon block	user	disc space [TB]	used [TB]	available [TB]
101	hcs	51	43	8
102	hcs	51	50	1
103	hcs/ESB	129	101	28
104	ESB	129	0	129
105	ESB/seq.	129	69	60
106	hcs	161	13	148

+ 600 TB “cold-space” storage (Dec. 2014)

Data sources

all kind of -omics data from internal and external projects like Horizon 2020, Innovative Medicine initiative (eTRIKS, Aetionomy), Epilepsie and Parkinson consortia

example: Genomics

Whole genome/exome sequencing @ LCSB

- WGS (Complete Genomics)
  - Current: 500 genomes over last 3 years
  - Planned: 300 genomes per year
  - 300 GB per genome (raw + processed data)
- WES (Illumina)
  - Done with service provider and collaborators:
    - Cologne Center for Genomics, Rotterdam and Illumina, US
  - Current: 600 epilepsy, 2000 control exomes
    - 38 TB raw + processed data
  - Planned: 10000 PD exomes

Challenges

- Data, big Data and dirty Data
- IT performance
- Analysis and Visualization

Size alone does not define big data

it is best defined as a combination of

- Volume (big and dirty)
- Velocity
- Variety
- Value

People who can deal with the 4V’s are a new class of “data scientists”

Changes how we store data

Relational

Row/Column structureHyperTable

Foreign KeyHypertable Key

NoSQL

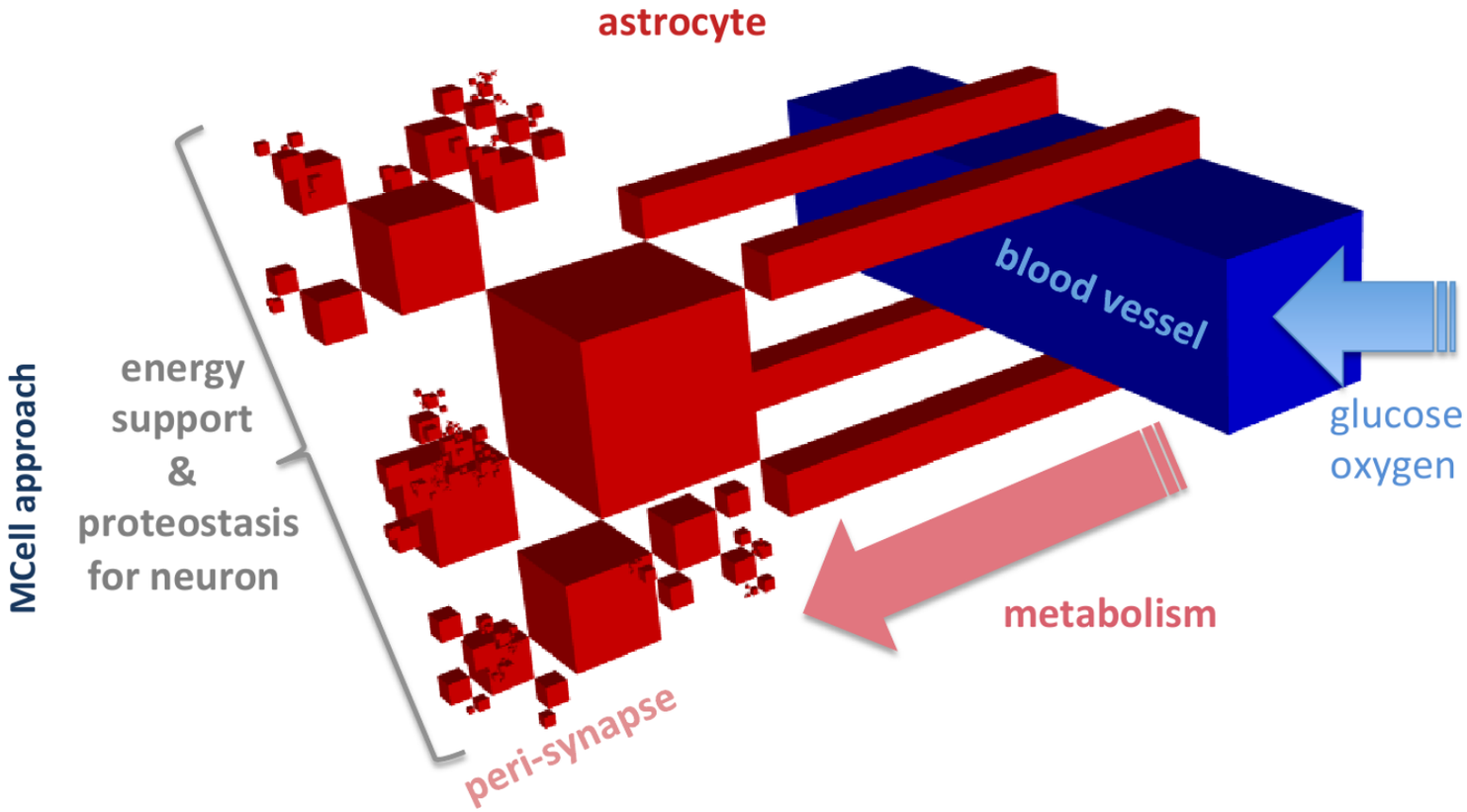
Rigid SchemaNo Schema

Scale UpScale Out

SQLHiveSQL

Relational	NoSQL
Row/Column structure	HyperTable
Foreign Key	Hypertable Key
Rigid Schema	No Schema
Scale Up	Scale Out
SQL	HiveSQL

Brain Energy Metabolism to approach PD
Multiscale Simulation of astrocytic metabolism



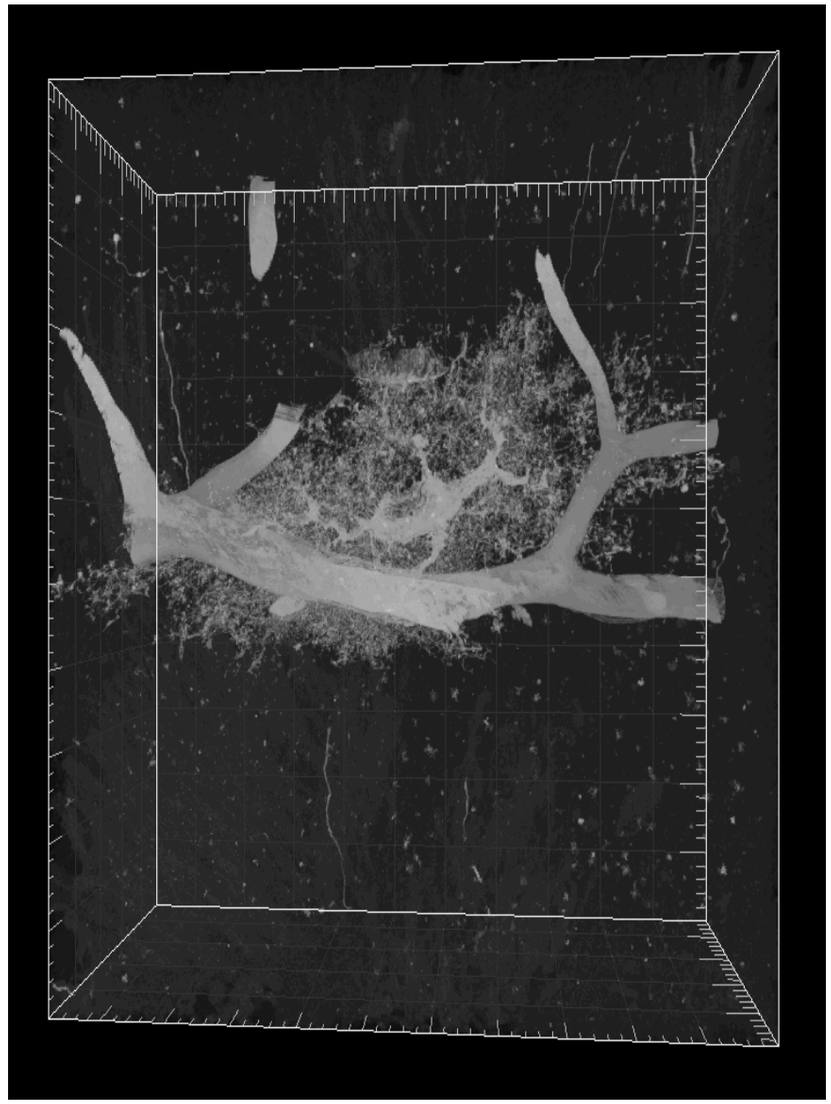
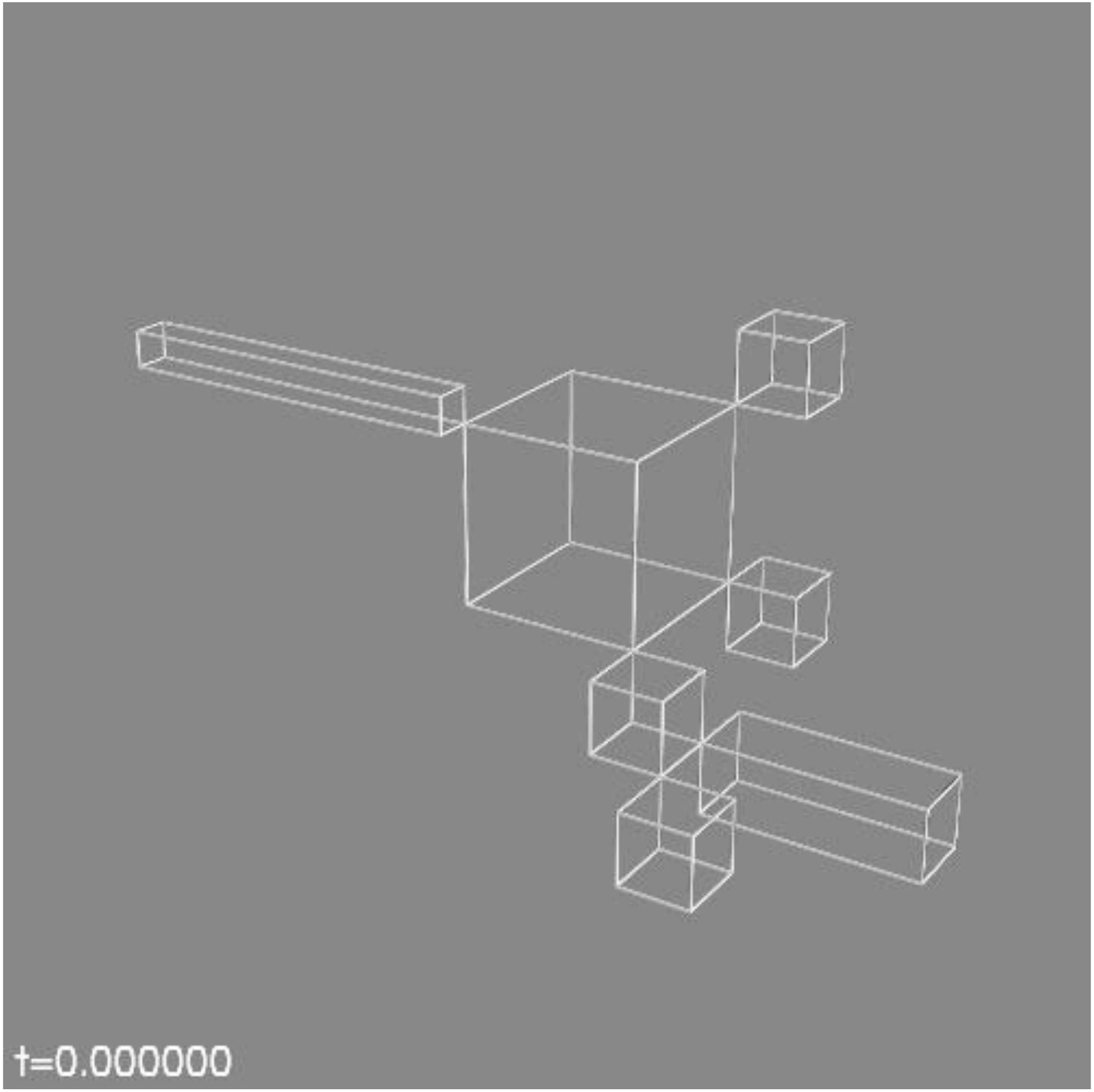


Figure 4) Astrocyte Reconstruction: Astrocyte will be reconstructed by cuboids. Here a first simple artificial form that can be used for testing purposes.

A. Skupin, LCSB



The system is parallelized (**MPI**) by grouping boxes into sets with equal number of reaction centers, where each connection between boxes is one reaction center in each box. Only the two connected boxes running in different processes exchange messages. Duration is highly dependend on MPI implementation (compare table 1).

MPI implementation	duration
MPICH	5h04m
MPICH2	5h09m
OPENMPI	5h12m
MVAPICH2	8h25m
IMPI	8h29m

Big Data impacting and Economic sectors

Healthcare:

Applications range from comparative effectiveness research to the next generation of **clinical decision support systems** which make use of comprehensive heterogeneous health data sets as well as advanced analytics of clinical operations. Of particular importance are **patient involvement, privacy and ethics**.

European Big Data Value cPPP - Strategic Research and Innovation Agenda - April 2014

Priority	Time —->			
Privacy and anonymisation	Generalisation of Secure Remote Data Access Centre techniques.	Method for deletion of data and data minimization. Robust anonymisation algorithms		
Deep analysis	Improved statistical models by enabling fast non-linear approximations in very large datasets	Predictive modeling Graph mining techniques applied on extremely large graphs	Real-time analytics Semantic analysis in near-real-time Algorithms for multimedia data	Deep learning techniques Descriptive language for deep analytics.
Architectures for analytics of data at rest and in motion	Optimized tools for the integration of existing components to new types of platforms with both data at rest and in motion.	Synergies between massively parallel architectures (MPP) and batch processing/stream processing architectures		
Advance visualisation	New data search solutions / paradigms	Semantic driven data visualisation – stronger links between visualization and analytics	User adaptation	Collaborative real-time, dynamic 3D solutions
Engineering data management	APIs for improving the process of data transformation	Collaborative Tools and techniques for Data Quality (including integrity and veracity check) Harmonized description format for meta-data	Methodology, models and tools for data lifecycle management	Data management as a service

- End

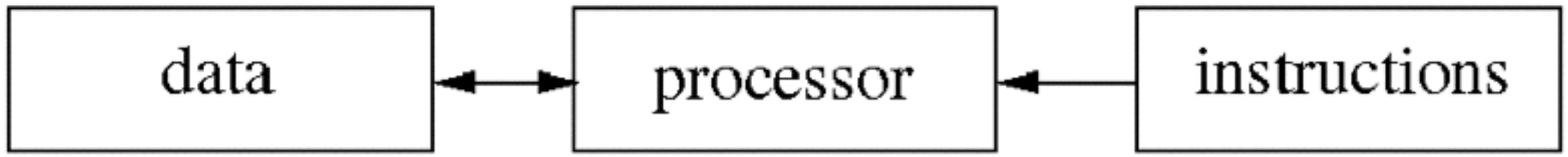
Flynn's Taxonomy of Parallel Systems

In the taxonomy of Flynn parallel systems are classified according to the number of instruction streams and data streams.

instruction stream \ data stream		
	SISD	SIMD
	MISD	MIMD

M. Flynn: Proc. IEEE 54 (1966), pp. 1901–1909.

SISD: Single Instruction stream - Single Data stream



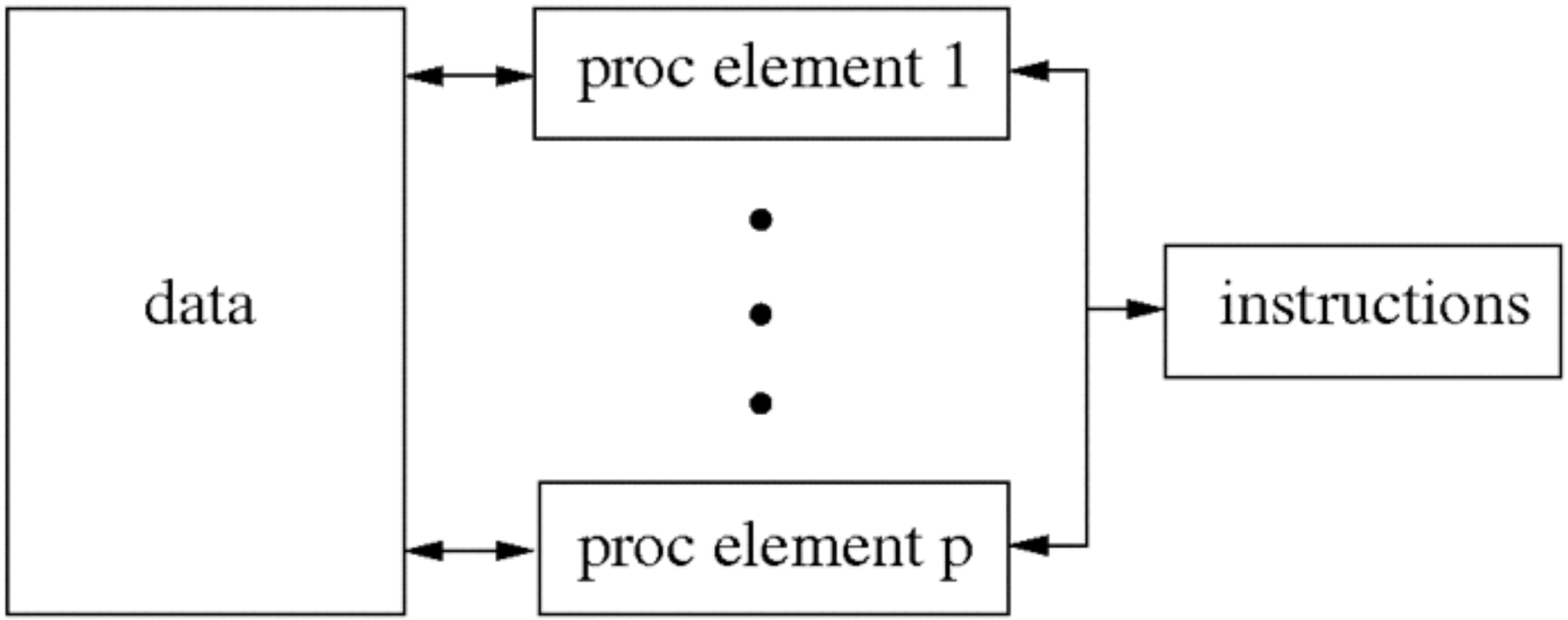
The **classical von Neumann machine**.

- processor: ALU, registers
- memory holds data and program
- bus (a collection of wires) = von Neumann bottleneck

Today's PCs or workstations are no longer true von Neumann machines:

- superscalar processors, parallel functional units
- pipelining
- Memory interleaving (memory banks)

SIMD: Single Instruction stream - Multiple Data stream



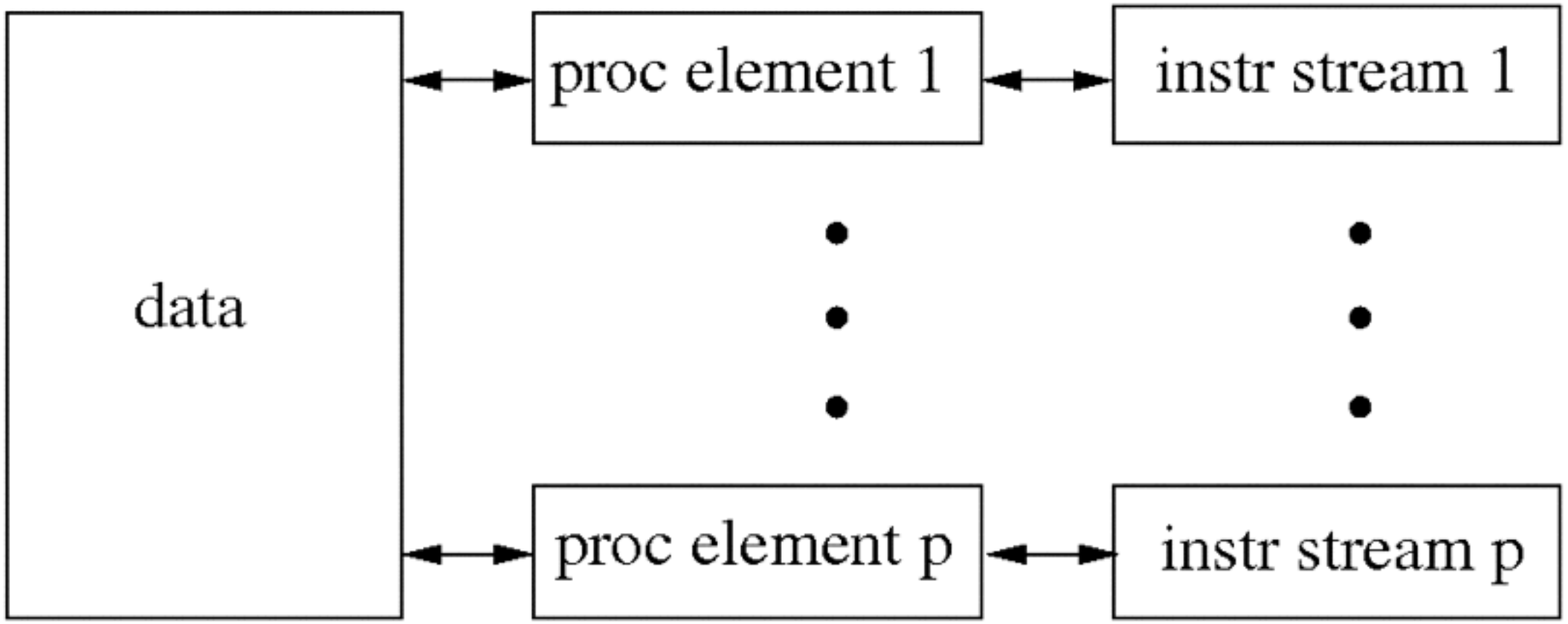
During each instruction cycle the central control unit broadcasts an instruction to the subordinate processors and each of them either executes the instruction or is idle.
At any given time a processor is "active" executing exactly the same instruction as all other processors in a completely synchronous way, or it is idle.

Example SIMD machines

- Vector computers like the Cray-1, -2, X-MP, Y-MP,... These machines pipelined operations with vectors of 64 floating point numbers.
- Streaming SIMD extensions (SSE, SSE2, SSE3, SSE4) Intel Pentium III and Pentium 4 have vector registers, also called multimedia or MMX registers that support operations with short vectors.
- Graphics processing units (GPUs).

More on SIMD processors: next week.

MIMD: Multiple Instruction stream - Multiple Data stream



Each processor can execute its own instruction stream on its own data independently from the other processors. Each processor is a full-fledged CPU with both control unit and ALU. MIMD systems are **asynchronous**.

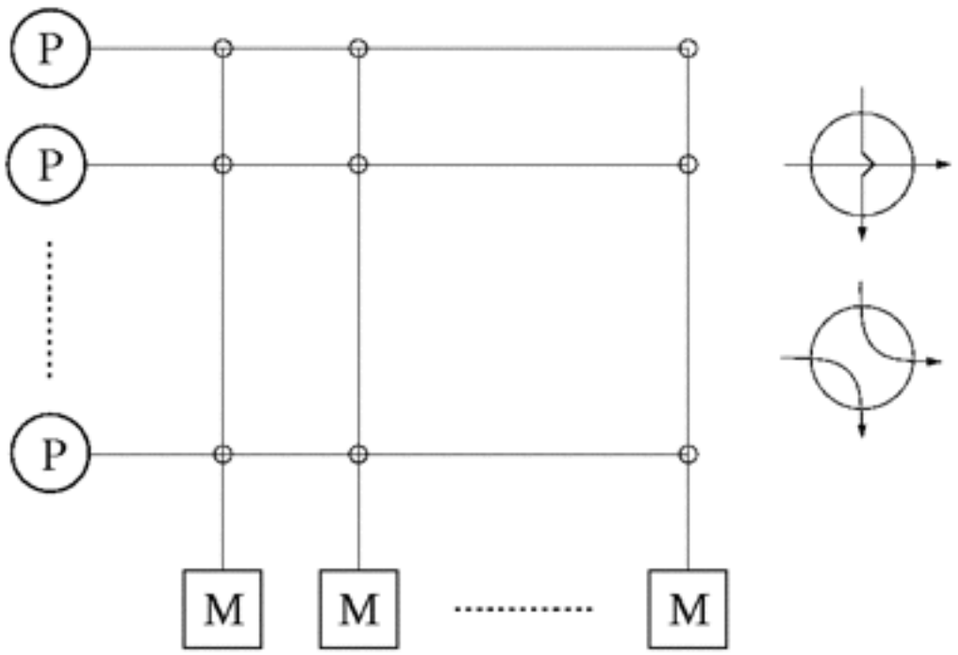
Memory organization

Most parallel machines are MIMD machines.
MIMD machines are classified by their memory organization

- shared memory machines (multiprocessors)
  - parallel processes, threads, OpenMP
  - communication by means of shared variables
  - data dependencies possible, race condition
  - *multi-core processors*
- distributed memory machines (multicomputers)
  - all data are local to some processor
  - programmer responsible for data placement
  - communication by message passing, MPI
  - easy / cheap to build $\longrightarrow$ Beowulf clusters (a cluster that is built of commodity hardware)

Interconnection network

Multiprocessors usually have a dynamic network, e.g, a *crossbar switch* to connect precessing elements and memory banks.

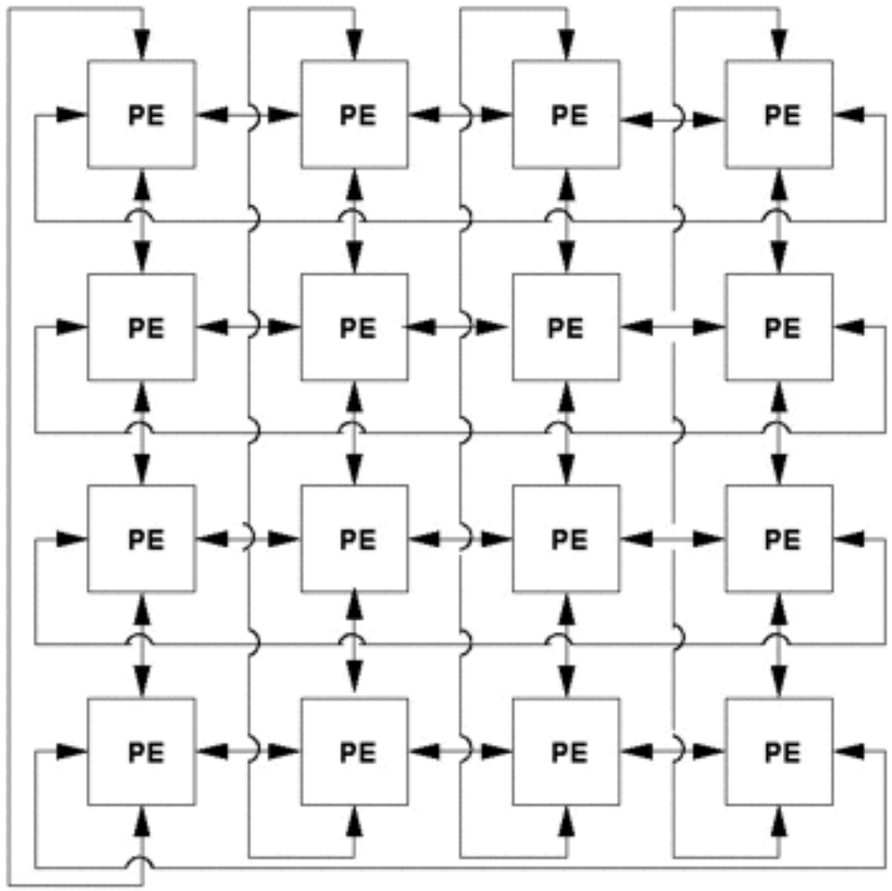


Crossbar switch with n processors and m memory modules. On the right the possible switch states.

Uniform access, scalable, very many wires $\rightarrow$ very expensive, used for limited number of processors only.

Interconnection network (cont.)

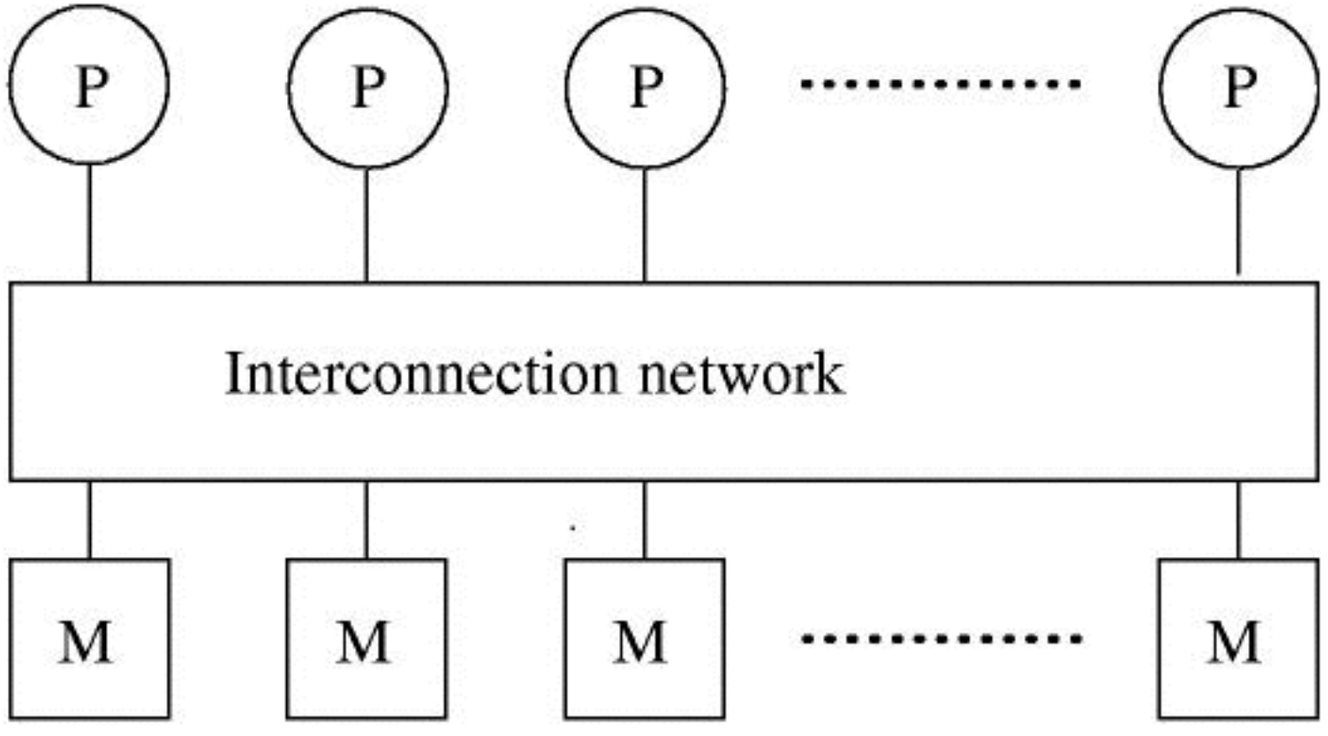
Multicomputers usually have static networks, like meshes, tori, hypercubes, or trees.



Processing elements are usually connected to the network through routers. Routers can pipeline messages.

Shared memory machines (multiprocessors)

- parallel processes / threads
- single address space accessible by all processes
- communication by means of shared variables
- data dependencies possible, race condition



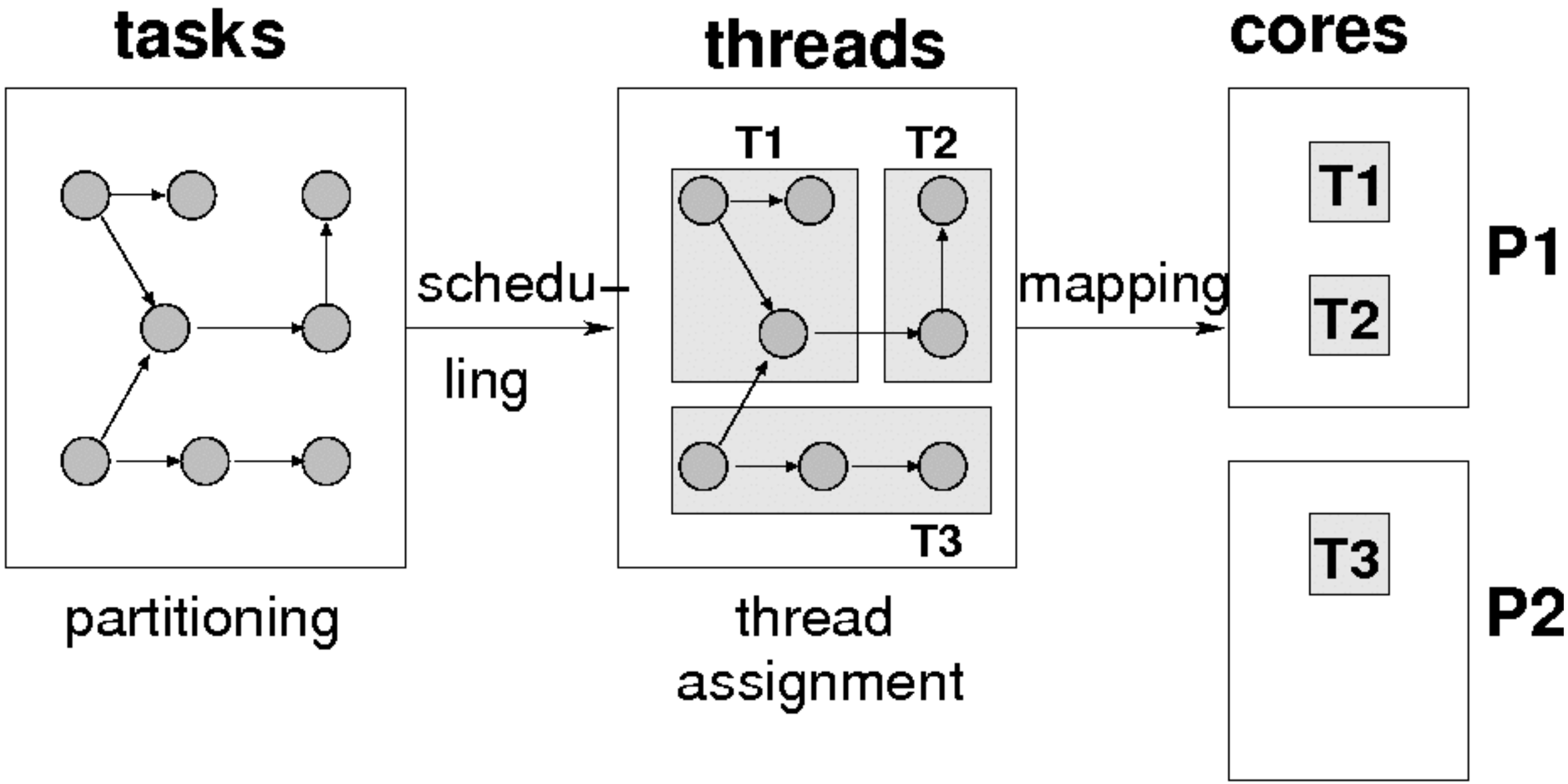
Parallel Programming Concepts

- The new generation of multicore processors get their performance from the multitude of cores on a chip.
- Unlike the situation until recently, the programmer has to take action to be able to exploit the improved performance.
- Techniques of parallel programming are known for years in scientific computing and elsewhere.
- What is new: parallel programming has reached main stream.
- The essential step in programming multicore processors is in providing multiple streams of execution that can be executed simultaneously (concurrently) on the multiple cores.
- Let us introduce a few concepts and notions.

Design of parallel programs

- Basic idea of parallel programming: generate multiple instruction streams that can be executed in parallel.
  If we have a sequential code available, we my parallelize it.
- In order to generate independant instruction streams we partition the problem that we want to solve into tasks. Tasks are the smallest units of parallelism.
- Tasks may depend on each other in one way or another. They may access the same data concurrently (data dependence) or they may need to wait for another task to finish (flow dependance) as its results are needed.
- The size of a task is called granularity (fine/coarse grain).

Design of parallel programs (cont.)



Execution of parallel programs

- **Multitasking (time sharing)**

  In operating systems that support multitasking several threads or processes are executed on the same processor in time slices (time sharing).
  In this way, latency due to, e.g., I/O operations can be hidden. This form of executing multiple tasks at the same time is called concurrency. Multiple tasks are executed at the same time, but only one of them has access to the compute resources at any given time. No simultaneous parallel excution is taking place.

Execution of parallel programs (cont.)

- **Multiprocessing**
  Using multiple physical processors admits the parallel execution of multiple tasks.
  The parallel hardware may cause overhead though.

Execution of parallel programs (cont.)

- **Simultaneous Multithreading (SMT)**

  In simultaneous multithreading (SMT) or hyperthreading multiple flows of control are running concurrently on a processor (or a core). The processor switches among these so-called threads of control by means of dedicated hardware. If multiple logical processors are executed on one physical processor then the hardware resources can be better employed and task execution may be sped up. (With two logical processors, performance improvements of up to 30% have been observed.)

Multicore programming

- Multicore processors are programmed with multithreaded programs.
- Although many programs use multithreading there are some notable differences between multicore programming and SMT.
- SMT is mainly used by the OS to hide I/O overhead. On multicore processors the work is actually distributed on the different cores.
- Cores have individual caches. False sharing may occur: Two cores may work on different data that is stored in the same cacheline. Although there is no data dependence the cache line of the other processor is marked invalid if the first processor writes its data item. (Massive) performance degradation is possible.

Multicore programming (cont.)

- Thread priorities.
  If multithreading programs are executed on a single processor machines, always the thread with the highest priority is executed.
  On multicore processors, threads with different priorities can be executed simultaneously. This may lead to different results!
  Programming multicore machines therefore requires techniques, methods, and designs from parallel programming.

Parallel programming models

The design of a parallel program is always based on an abstract view of the parallel system on which the software shall be executed.
This abstract view is called **parallel programming model**.
It does not only describe the underlying hardware. It describes the the whole system as it is presented to a software developer:

- System software (operating system)
- parallel programming language
- parallel library
- compiler
- run time system