Primena super-računara u astronomiji

“**High Performance Computing** most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.”

The Forms of HPC

- The commodity HPC cluster
- Dedicated supercomputer
- HPC cloud computing
- Grid computing

The commodity HPC cluster

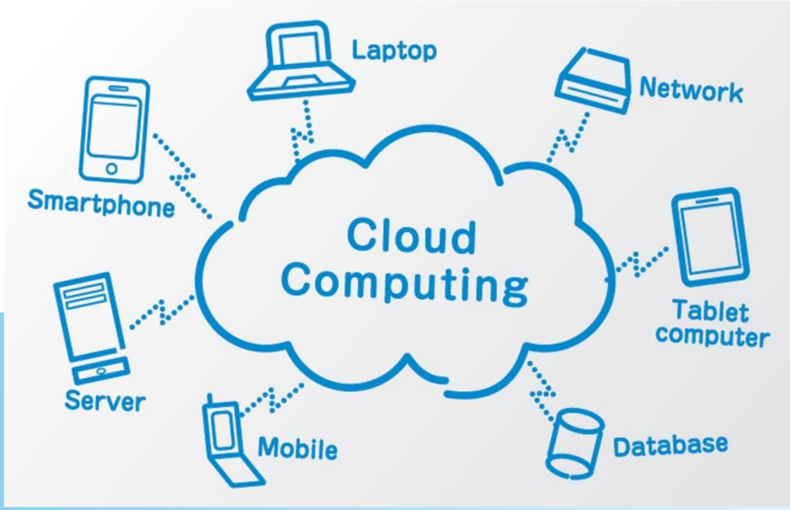
- Over the last ten years, the HPC cluster has disrupted the entire supercomputing market. Built from standard off-the-shelf servers and high speed interconnects, a typical HPC system can deliver industry-leading, cost-effective performance.
- A typical cluster can employ hundreds, thousands, and even tens of thousands of servers all working together on a single problem (this is the high tech equivalent of a “divide and conquer” approach to solving large problems).
- Because of high performance and low cost, the commodity cluster is by far the most popular form of HPC computing. Also keep in mind the compatibility advantage — x86 commodity servers are ubiquitous.

Dedicated supercomputer

- In the past, the dedicated supercomputer was the only way to throw a large number of compute cycles at a problem.
- Supercomputers are still produced today and often use specialized non-commodity components.
- Depending on once needs, the supercomputer may be the best solution although it doesn’t offer the commodity price advantage.

HPC cloud computing

- This method is relatively new and employs the Internet as a basis for a cycles-as-a-service model of computing. The compute cycles in question live in the cloud somewhere allowing a user to request remote access to cycles on-demand.
- An HPC cloud provides dynamic and scalable resources (and possibly virtualization) to the end-user as a service.
- Although clouds can be cost effective and allow HPC to be purchased as an expense and not a capital asset, it also places some layers between the user and hardware that may reduce performance.



Grid Computing

- Grid is similar to cloud computing, but requires more control by the end-user. Its main use is academic projects where local HPC clusters are connected and shared on a national and international level.
- Some computational grids span the globe while others are located within a single organization.

Cloud vs Grid Computing

- Cloud computing is simply a method of storing and accessing data or software over the Internet rather than a local hard drive. “The Cloud” is tech jargon for a virtual, seamless connection. The term comes as a result of flowcharts and presentations that often visualize this virtual connection using a cloud.
- At its core, grid computing is a computer network in which each computer’s resources are shared. Processing power, memory, and data storage are all community resources. Therefore authorized users can tap into and leverage these resources for specific tasks.

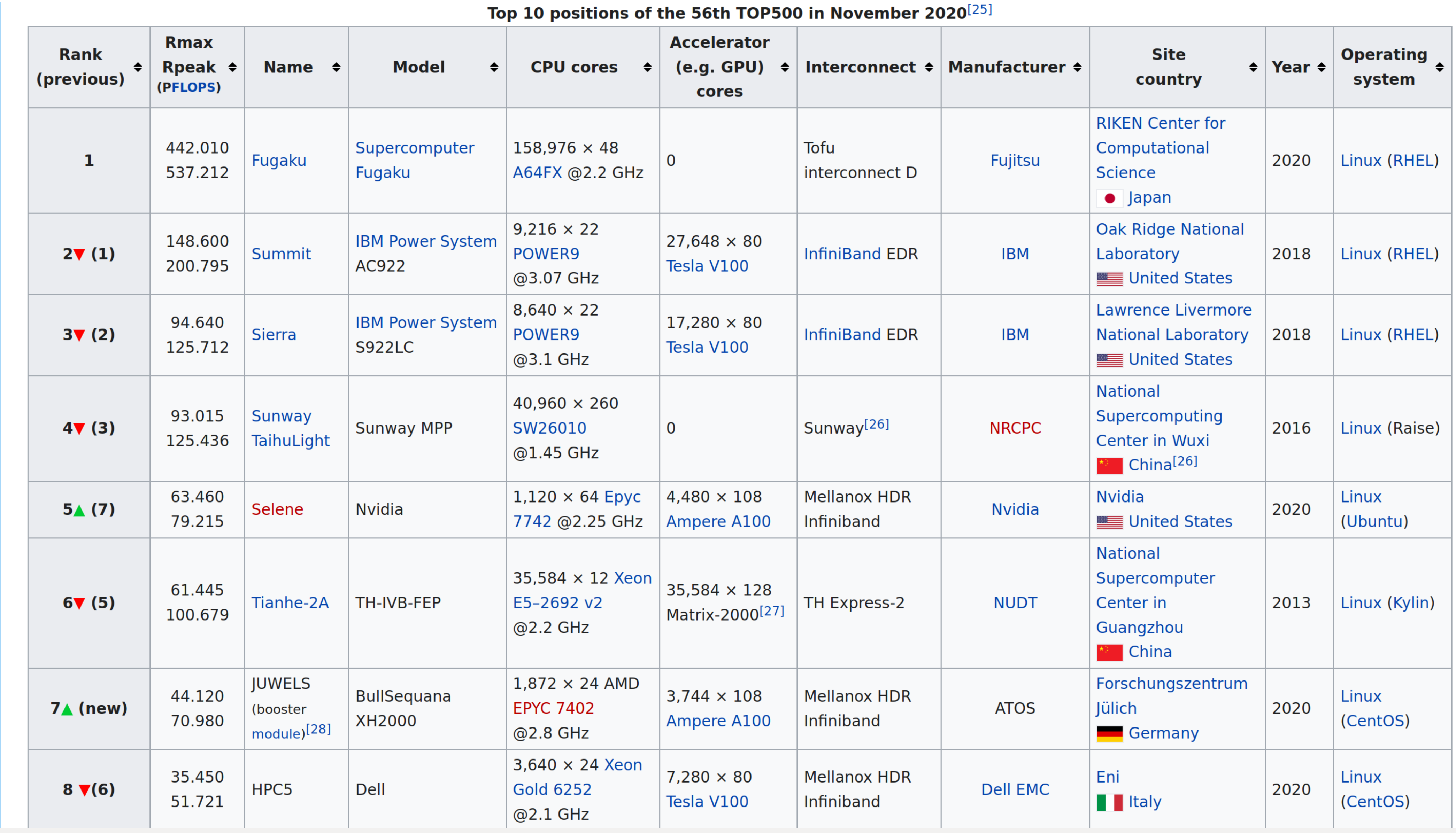
Cloud vs Grid Computing

- **Cloud computing:**
  - Adventages: Disaster Recovery, Increased Collaboration and Flexibility, Eco Friendly
  - Disadventages: Internet Connectivity, Learning Curve
- **Grid computing:**
  - Adventages: Cheaper Servers, More Efficient, Fail-safe
  - Disadventages: May Still Require Large SMP, Requires Fast Interconnect, Some Applications Require Customization, Licensing

Cloud vs Grid Computing

- At face value, Cloud Computing and Grid Computing are very similar, but often serve very specific needs, projects and use cases.
- Cloud computing is great for flexibility, ease-of-use, and security, while Grid Computing makes utilizing physical hardware more economical when used in the right way.
- So, Cloud Computing vs Grid Computing, which is better? The answer really comes down to what you are trying to do and the resources you have at your disposal.

Top 500

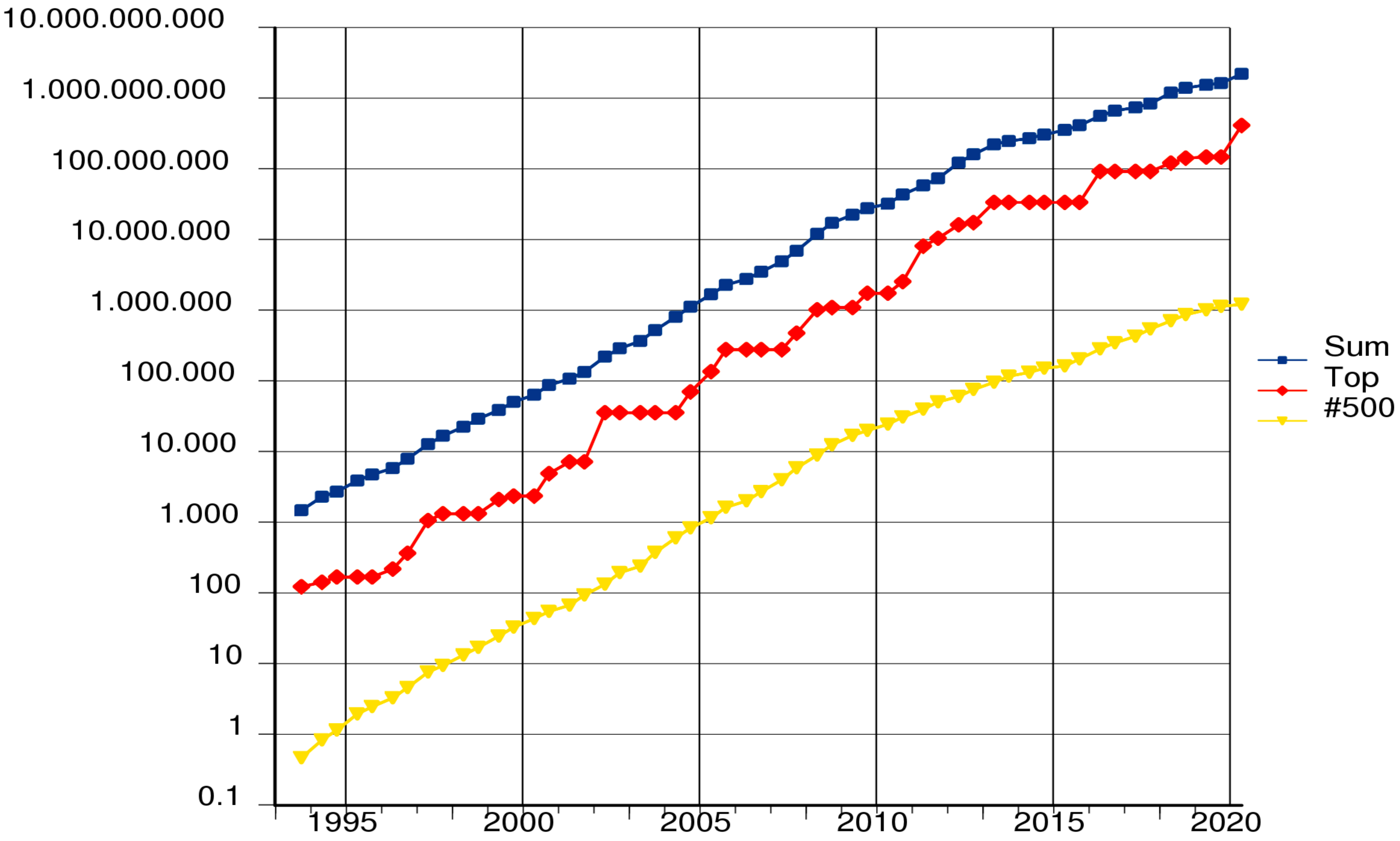


Top 10 positions of the 56th TOP500 in November 2020[25]

Rank (previous)	Rmax Rpeak (PFLOPS)	Name	Model	CPU cores	Accelerator (e.g. GPU) cores	Interconnect	Manufacturer	Site country	Year	Operating system
1	442.010 537.212	Fugaku	Supercomputer Fugaku	158,976 × 48 A64FX @2.2 GHz	0	Tofu interconnect D	Fujitsu	RIKEN Center for Computational Science Japan	2020	Linux (RHEL)
2▼ (1)	148.600 200.795	Summit	IBM Power System AC922	9,216 × 22 POWER9 @3.07 GHz	27,648 × 80 Tesla V100	InfiniBand EDR	IBM	Oak Ridge National Laboratory United States	2018	Linux (RHEL)
3▼ (2)	94.640 125.712	Sierra	IBM Power System S922LC	8,640 × 22 POWER9 @3.1 GHz	17,280 × 80 Tesla V100	InfiniBand EDR	IBM	Lawrence Livermore National Laboratory United States	2018	Linux (RHEL)
4▼ (3)	93.015 125.436	Sunway TaihuLight	Sunway MPP	40,960 × 260 SW26010 @1.45 GHz	0	Sunway[26]	NRCPC	National Supercomputing Center in Wuxi China[26]	2016	Linux (Raise)
5▲ (7)	63.460 79.215	Selene	Nvidia	1,120 × 64 Epyc 7742 @2.25 GHz	4,480 × 108 Ampere A100	Mellanox HDR Infiniband	Nvidia	Nvidia United States	2020	Linux (Ubuntu)
6▼ (5)	61.445 100.679	Tianhe-2A	TH-IVB-FEP	35,584 × 12 Xeon E5–2692 v2 @2.2 GHz	35,584 × 128 Matrix-2000[27]	TH Express-2	NUDT	National Supercomputer Center in Guangzhou China	2013	Linux (Kylin)
7▲ (new)	44.120 70.980	JUWELS (booster module)[28]	BullSequana XH2000	1,872 × 24 AMD EPYC 7402 @2.8 GHz	3,744 × 108 Ampere A100	Mellanox HDR Infiniband	ATOS	Forschungszentrum Jülich Germany	2020	Linux (CentOS)
8 ▼(6)	35.450 51.721	HPC5	Dell	3,640 × 24 Xeon Gold 6252 @2.1 GHz	7,280 × 80 Tesla V100	Mellanox HDR Infiniband	Dell EMC	Eni Italy	2020	Linux (CentOS)

Supercomputer Fugaku

Top 500



10.000.000.000
1.000.000.000
100.000.000
10.000.000
1.000.000
100.000
10.000
1.000
100
10
1
0.1
1995
2000
2005
2010
2015
2020
Sum
Top
#500

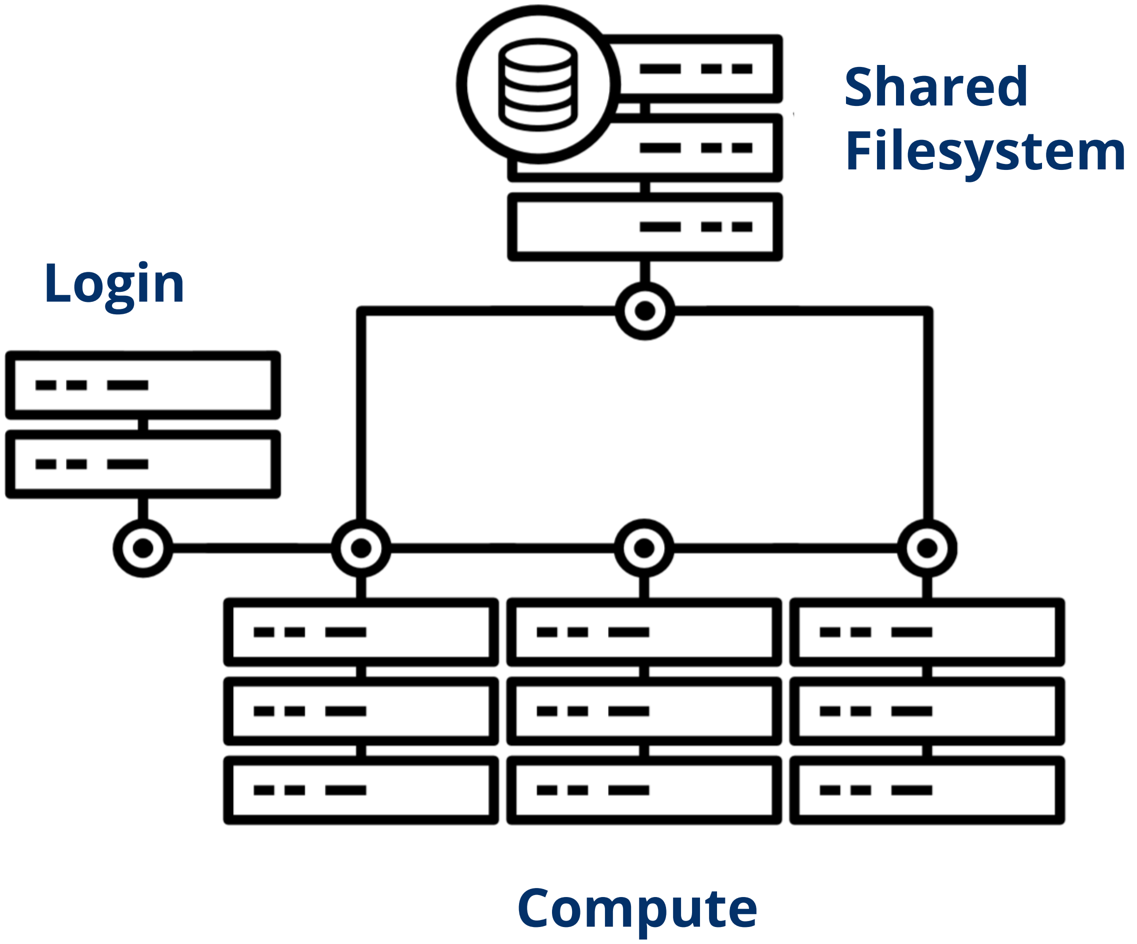
HPC Clusters

What is a Cluster?

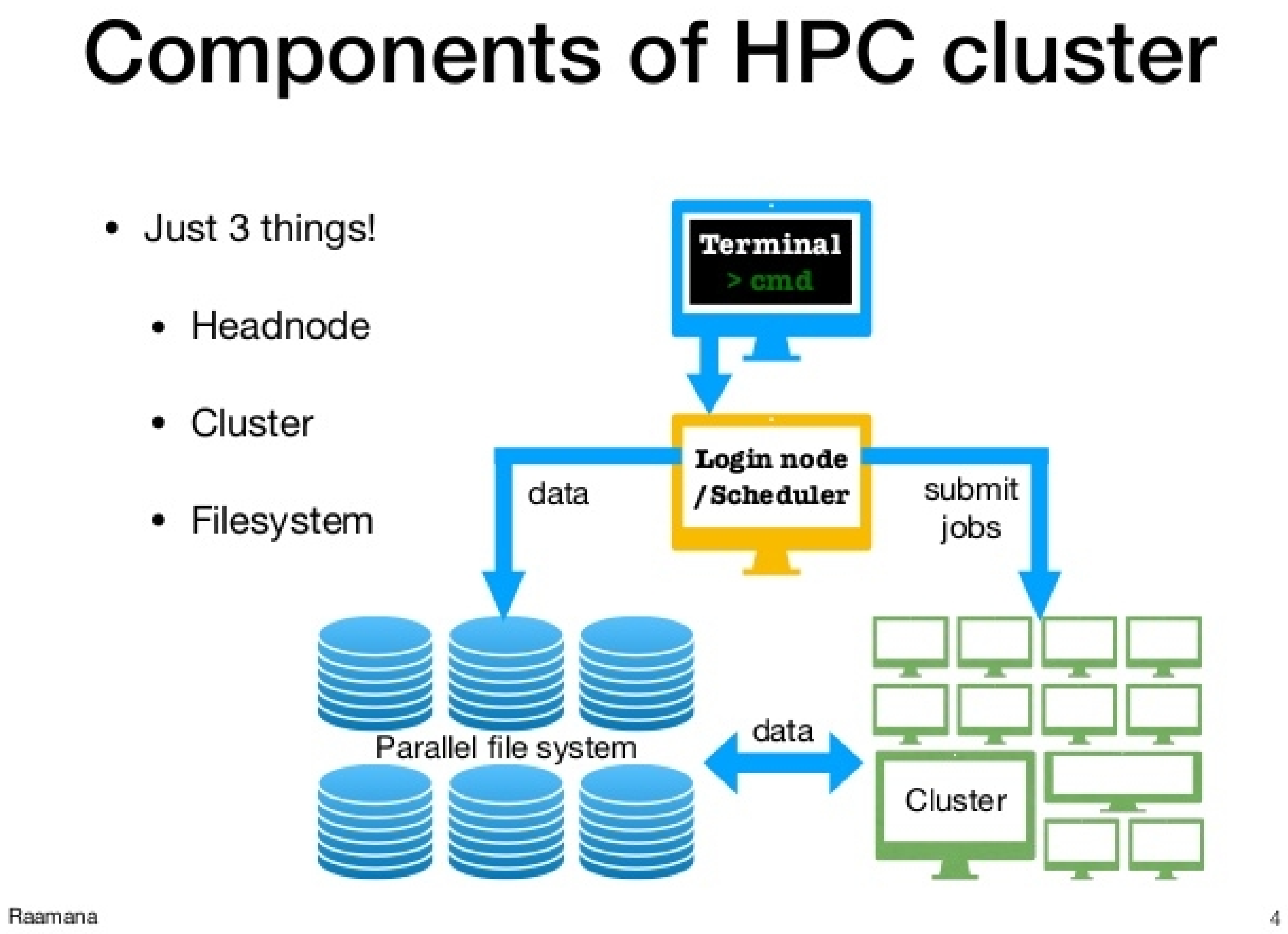
- 100s-1000s of rack-mounted computers
- Networking
- Storage

Abstract Cluster Diagram

- Access via the login nodes
- Shared filesystem presents data across all nodes
- Submit jobs scheduled to run on compute nodes



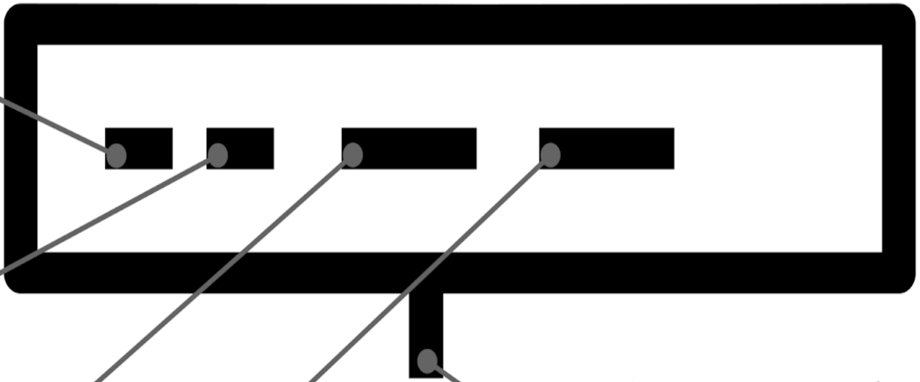
Abstract Cluster Diagram



Components of HPC cluster

- Just 3 things!
  - Headnode
  - Cluster
  - Filesystem

Compute Node



CPU Each CPU has discrete processor cores that you see as separate CPUs when using the cluster. On each node they are all the same, but between nodes they may be different.

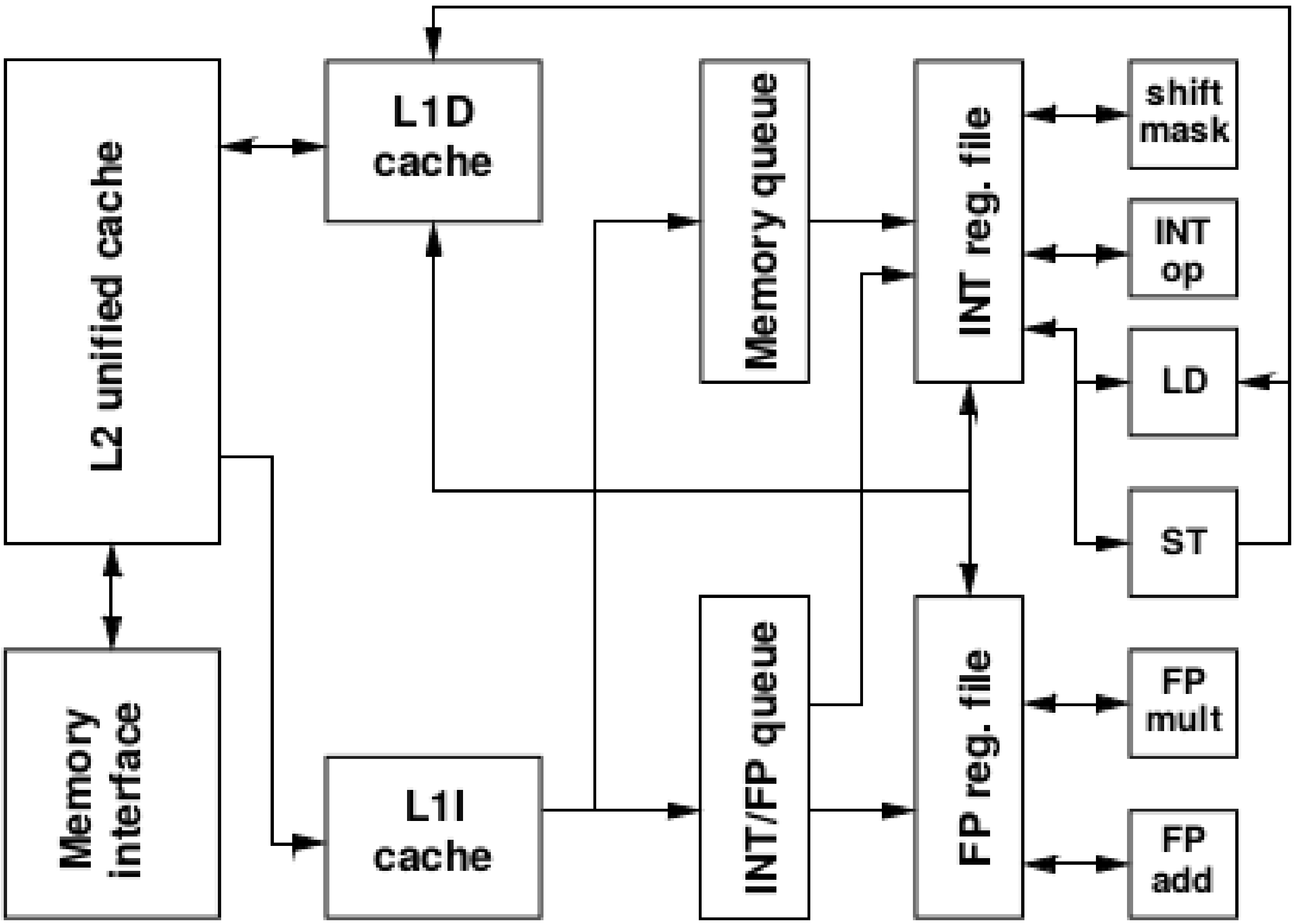
RAM Random Access Memory is where files and programs are loaded to run. Make sure you request enough for your job, otherwise you might get errors.

HDD The Hard Disk Drive on each node stores the operating system and some programs. It's also where /tmp is. Files on a node's HDD can only be seen by that node.

GPU Grapics Processing Unit. Some nodes also have graphics processors. Though historically used for gaming and image processing, they can also be used in some parallel computing.

NIC Network Interface Card. Connects the node to other nodes, shared storage, and other networks. Most nodes also use the NIC to connect to the internet.

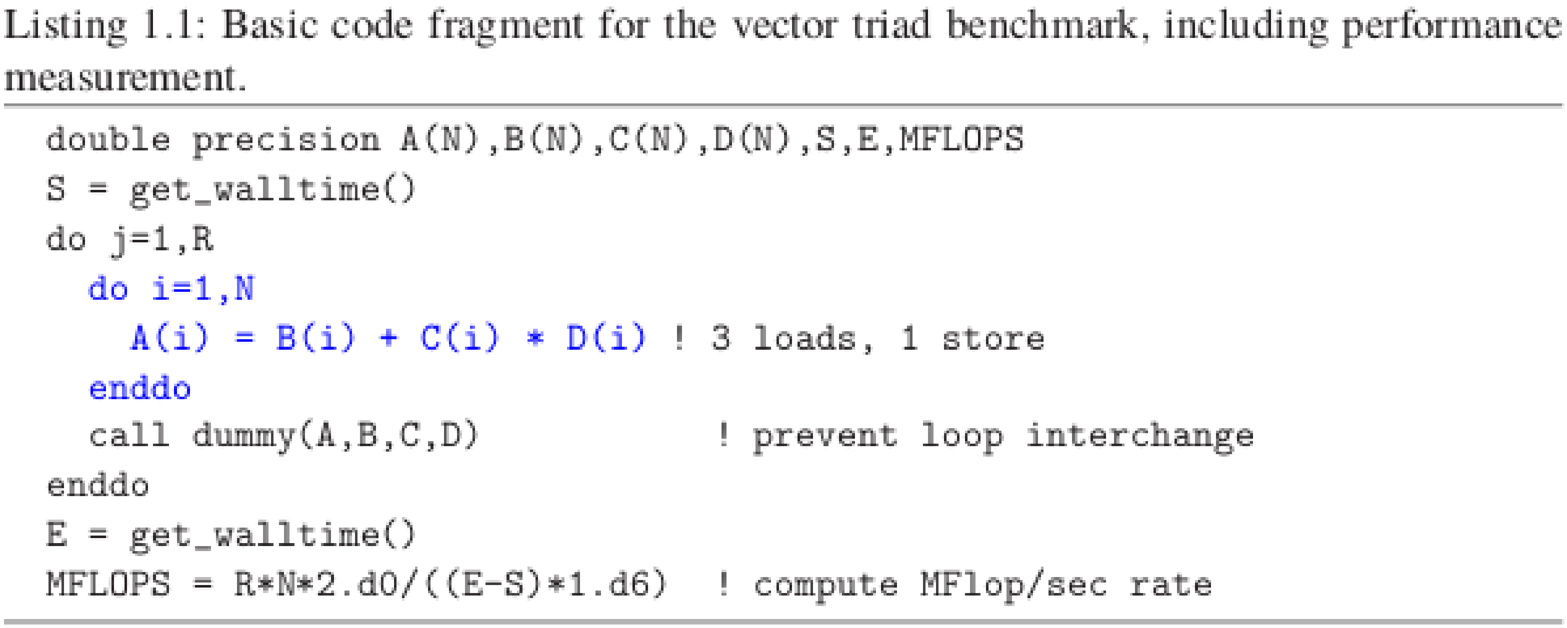
Microprocessor architecture



Performance metrics and benchmarks

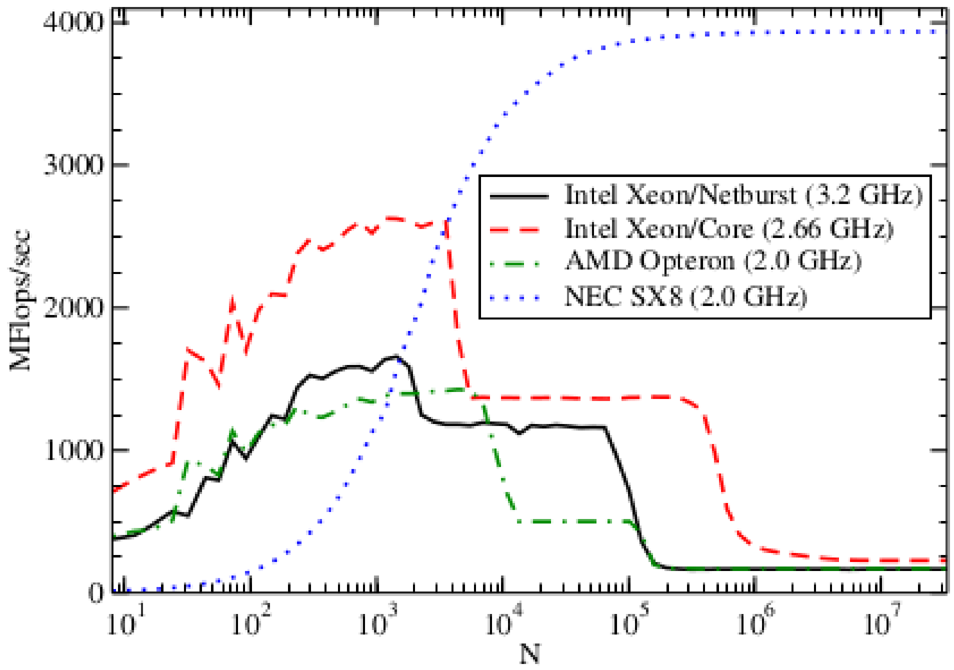
- maximum operating speed - ***peak performance***
- floating-point operations per second – **Flops/sec**
- measuring multiply and add operations execution time
- operation like square root or trigonometric function too slow – should be avoided
- *low-level benchmarking*: a program to test some specific feature of the architecture
- *application benchmarks*

Performance metrics and benchmarks

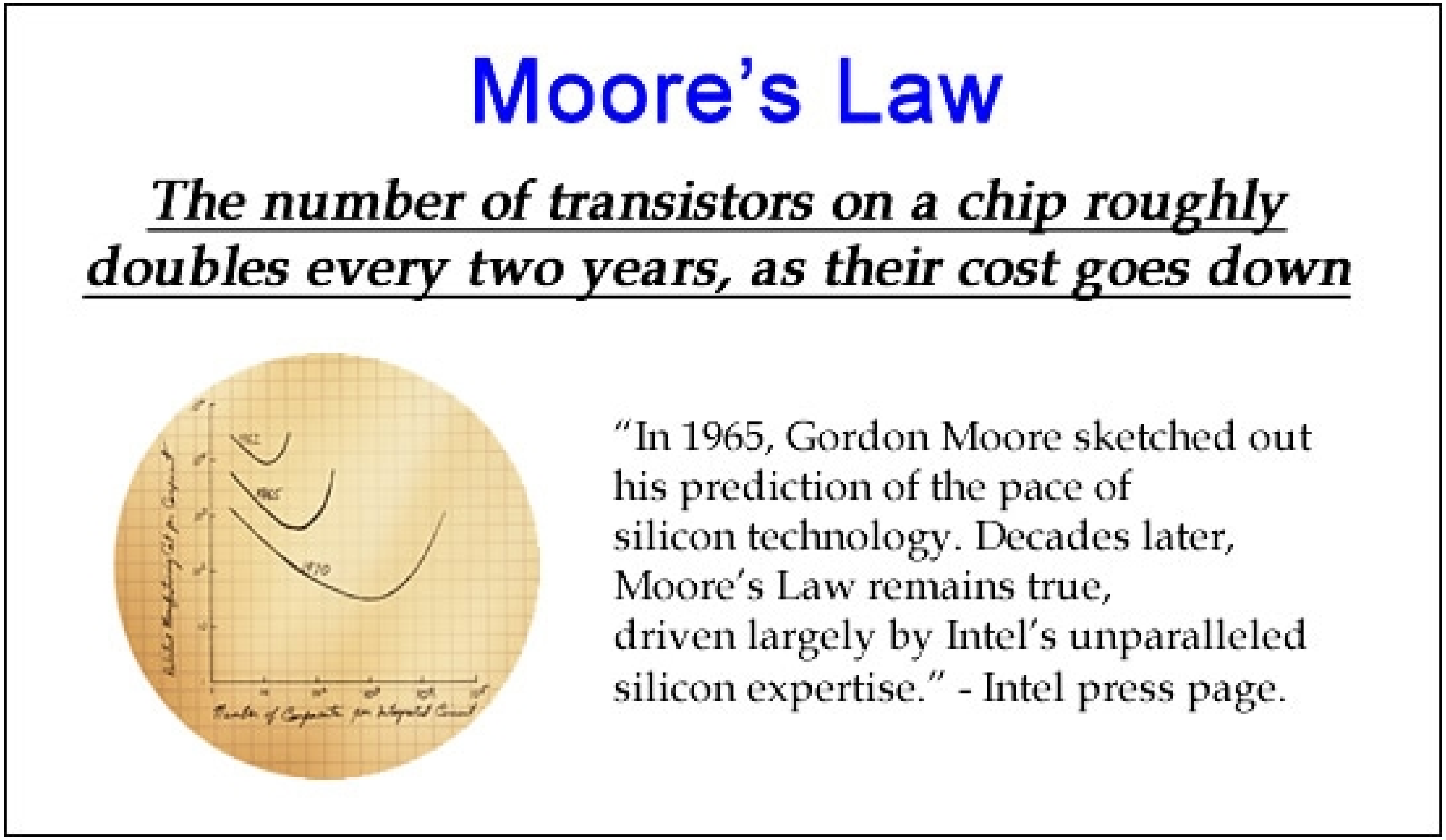


Listing 1.1: Basic code fragment for the vector triad benchmark, including performance measurement.

```
double precision A(N),B(N),C(N),D(N),S,E,MFLOPS
S = get_walltime()
do j=1,R
  do i=1,N
    A(i) = B(i) + C(i) * D(i) ! 3 loads, 1 store
  enddo
  call dummy(A,B,C,D)           ! prevent loop interchange
enddo
E = get_walltime()
MFLOPS = R*N*2.d0/((E-S)*1.d6)  ! compute MFlop/sec rate
```



Moore’s Law

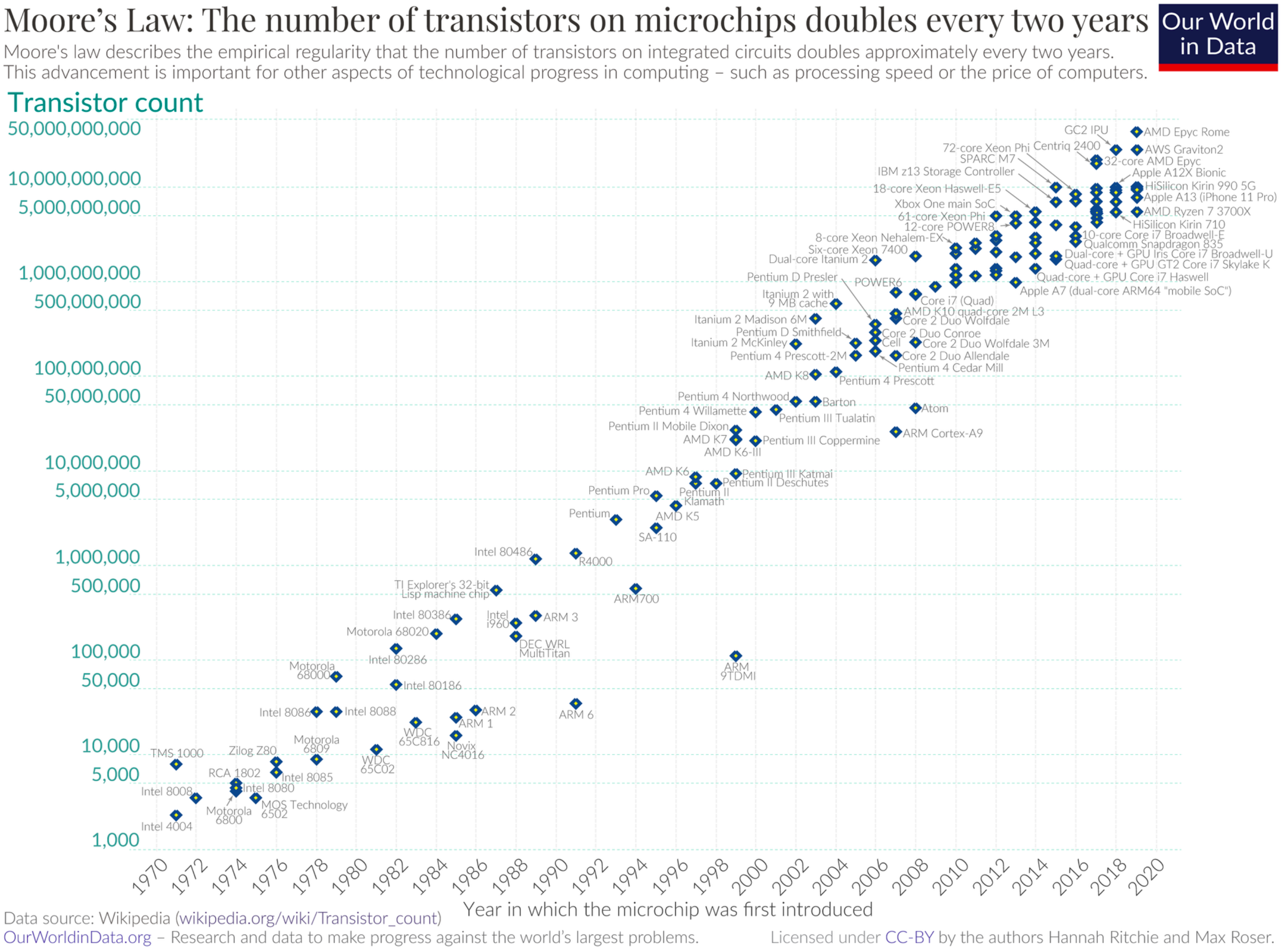


Moore’s Law

The number of transistors on a chip roughly doubles every two years, as their cost goes down

“In 1965, Gordon Moore sketched out his prediction of the pace of silicon technology. Decades later, Moore’s Law remains true, driven largely by Intel’s unparalleled silicon expertise.” - Intel press page.

Moore’s Law



Moore’s Law: The number of transistors on microchips doubles every two years

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.

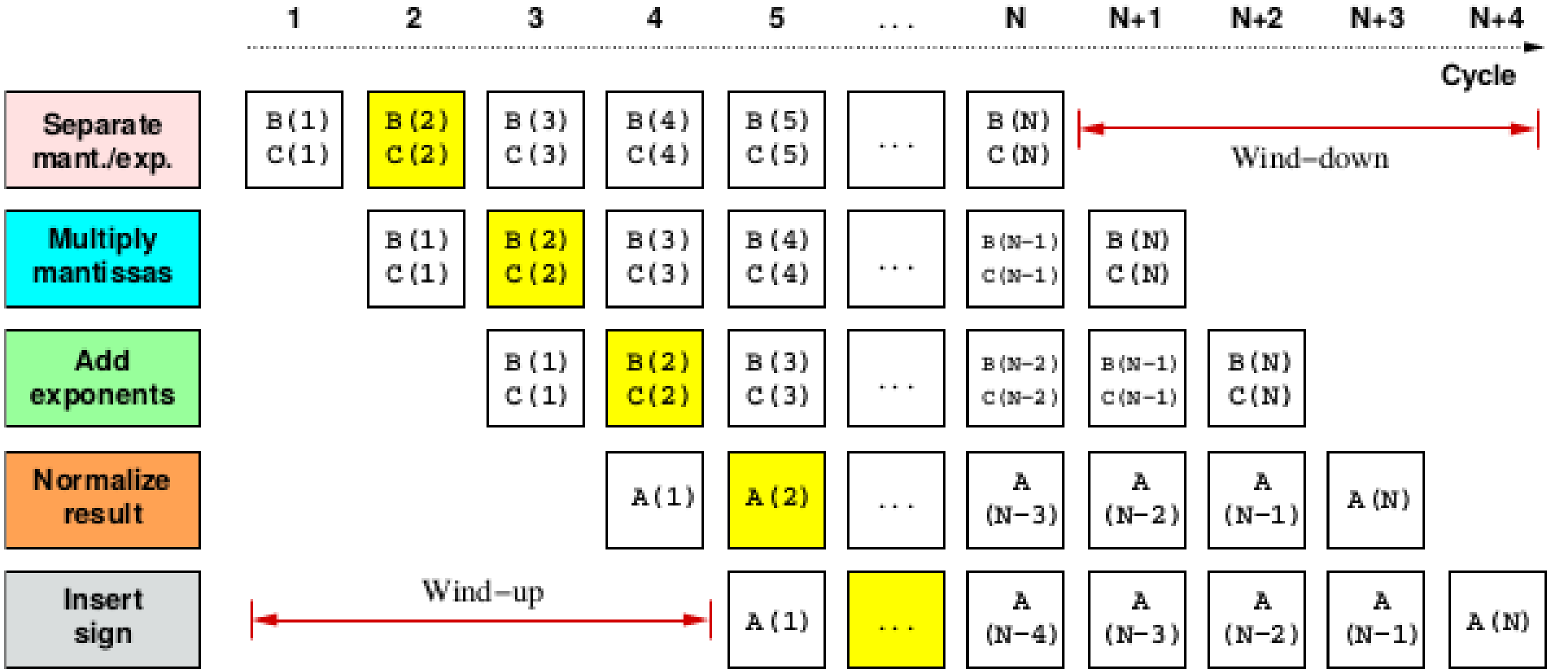
Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
OurWorldinData.org – Research and data to make progress against the world’s largest problems.

Advanced processors concepts

Advanced processors concepts have been developed to improve application performance:

- *Pipelined functional units*
- *Superscalar architecture*
- *Out-of-order execution*
- *Larger caches*
- *Advancement of instruction set design (CISC, RISC, EPIC)*

Pipelining



	1	2	3	4	5	...	N	N+1	N+2	N+3	N+4
Separate mant./exp.	B(1) C(1)	B(2) C(2)	B(3) C(3)	B(4) C(4)	B(5) C(5)	...	B(N) C(N)				
Multiply mantissas		B(1) C(1)	B(2) C(2)	B(3) C(3)	B(4) C(4)	...	B(N–1) C(N–1)	B(N) C(N)			
Add exponents			B(1) C(1)	B(2) C(2)	B(3) C(3)	...	B(N–2) C(N–2)	B(N–1) C(N–1)	B(N) C(N)		
Normalize result				A(1)	A(2)	...	A (N–3)	A (N–2)	A (N–1)	A(N)	
Insert sign					A(1)	...	A (N–4)	A (N–3)	A (N–2)	A (N–1)	A(N)

- a pipeline of depth (or latency) m, executing N independent, subsequent operations takes **N + m – 1 steps**
- a general-purpose unit needs **m-cycles** to generate a single result

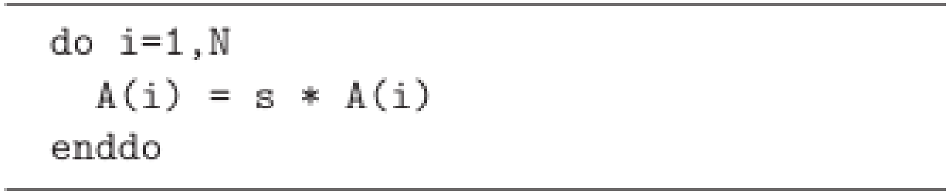
Pipelining

- the expected speedup: $$\frac{T_{\mathrm{seq}}}{T_{\mathrm{pipe}}} = \frac{mN}{N+m-1} \,,$$
- throughput: $$\frac{N}{T_{\mathrm{pipe}}} = \frac{1}{1+\frac{m-1}{N}} \,,$$
- the deeper the pipeline the larger the number of independent operations must be to achieve reasonable throughput because of the overhead incurred by *wind-up* and *wind-down* phases
- min ***N*** to get at least ***p*** results per cycle

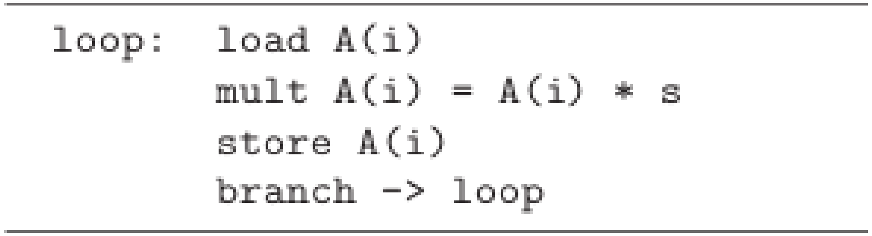
$$p = \frac{1}{1+\frac{m-1}{N_c}} \quad \Longrightarrow \quad N_c = \frac{(m-1)p}{1-p}$$

Software Pipelining

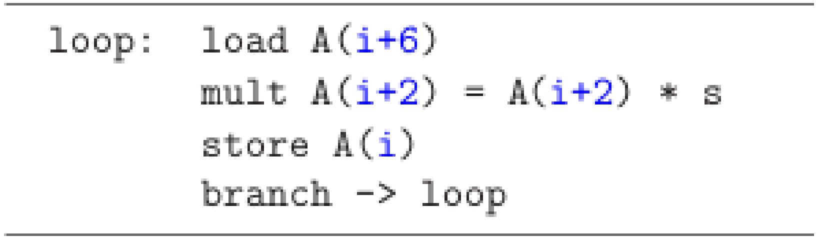
- Interleaving of loop iterations in order to meet latency requirements is called *software pipelining*



```
do i=1,N
  A(i) = s * A(i)
enddo
```

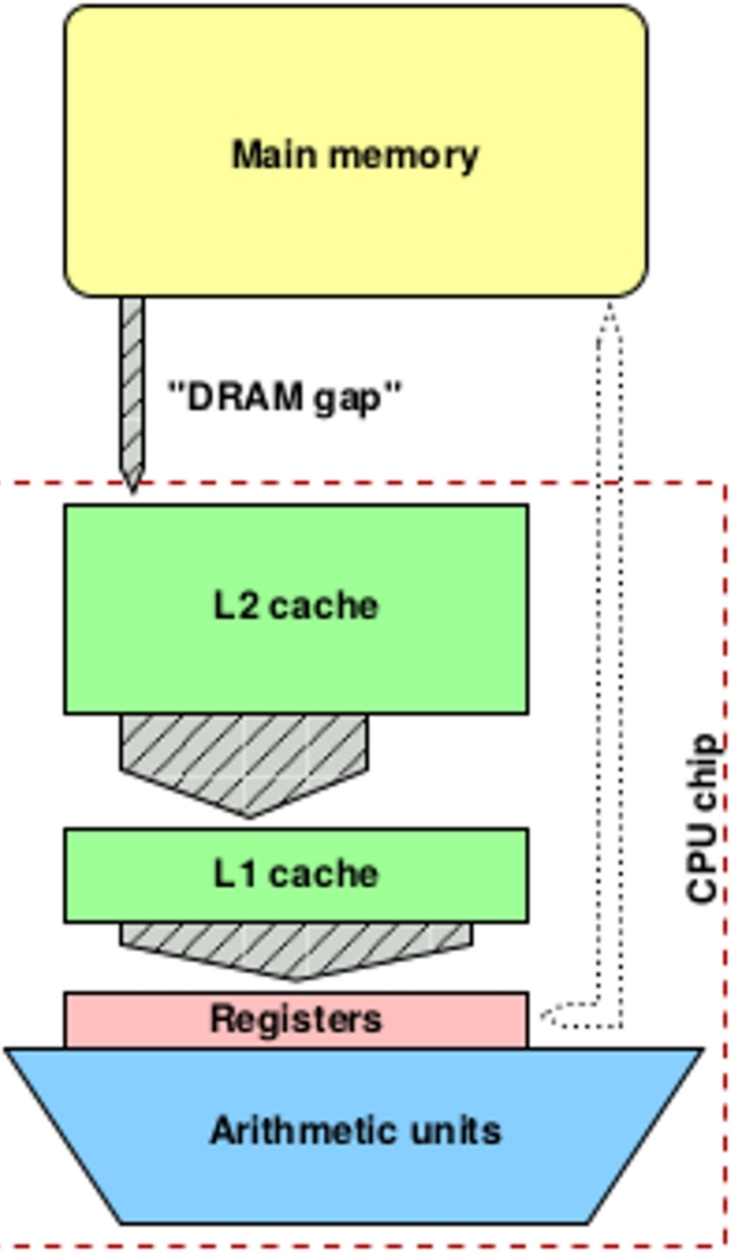


```
loop:   load A(i)
        mult A(i) = A(i) * s
        store A(i)
        branch -> loop
```



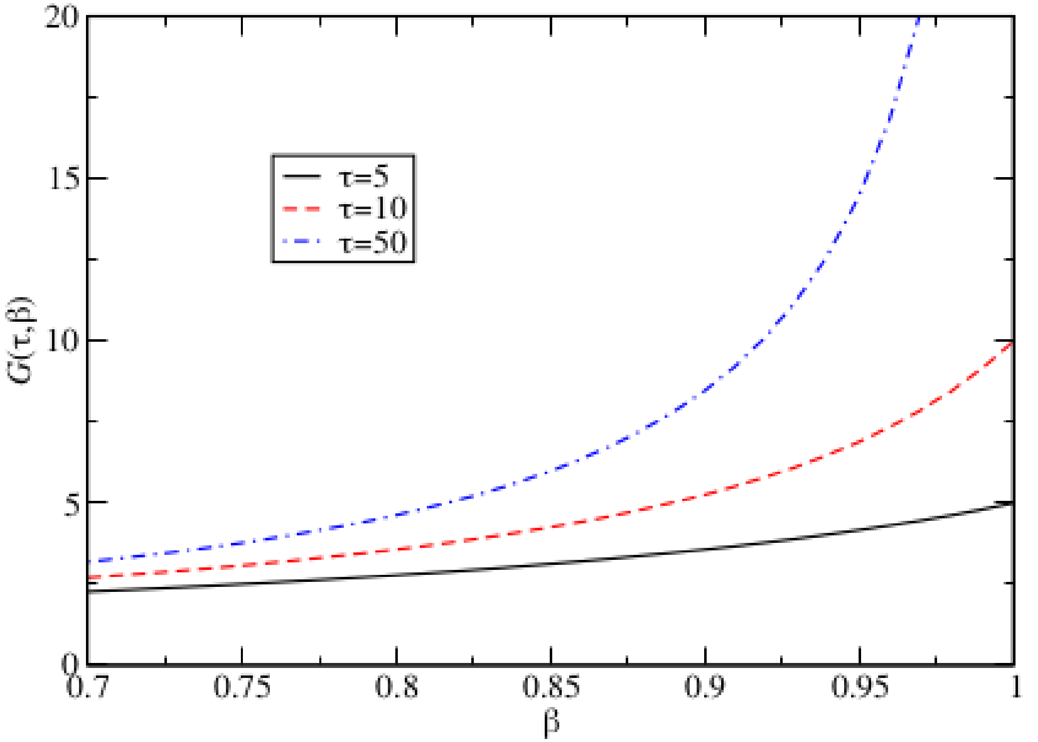
```
loop:   load A(i+6)
        mult A(i+2) = A(i+2) * s
        store A(i)
        branch -> loop
```

Memory Hierarchies

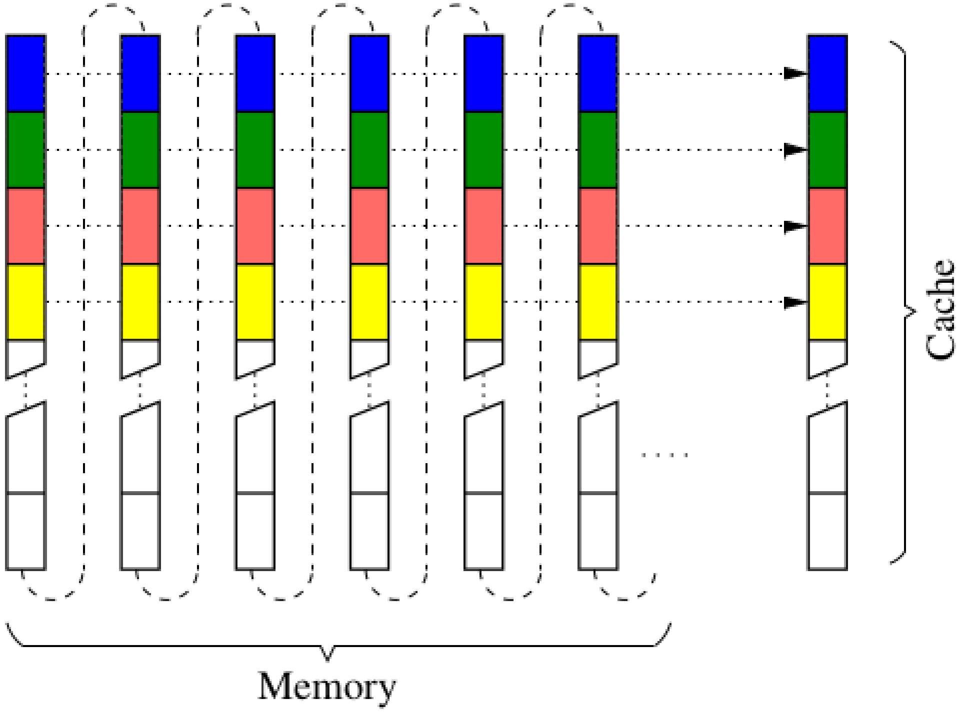


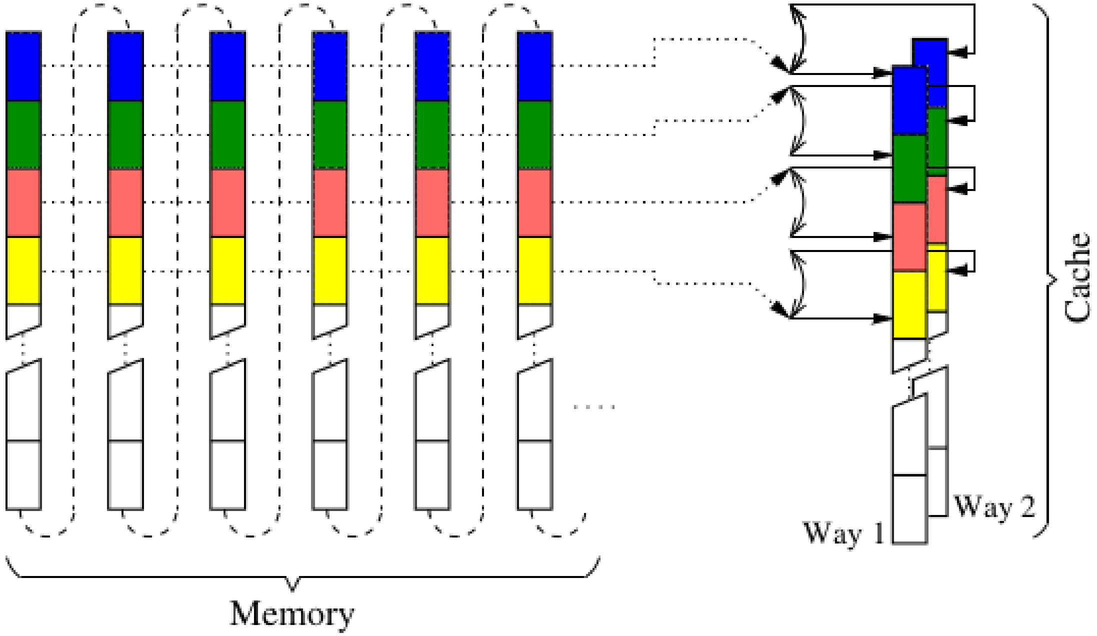
Cache Memory

- Caches are low-capacity, high-speed memories
- Main memory is much slower but also much larger than cache
- Caches can only have a positive effect on performance if the data access pattern of an application shows some locality of reference



Cache Memory Mapping





Prefetch

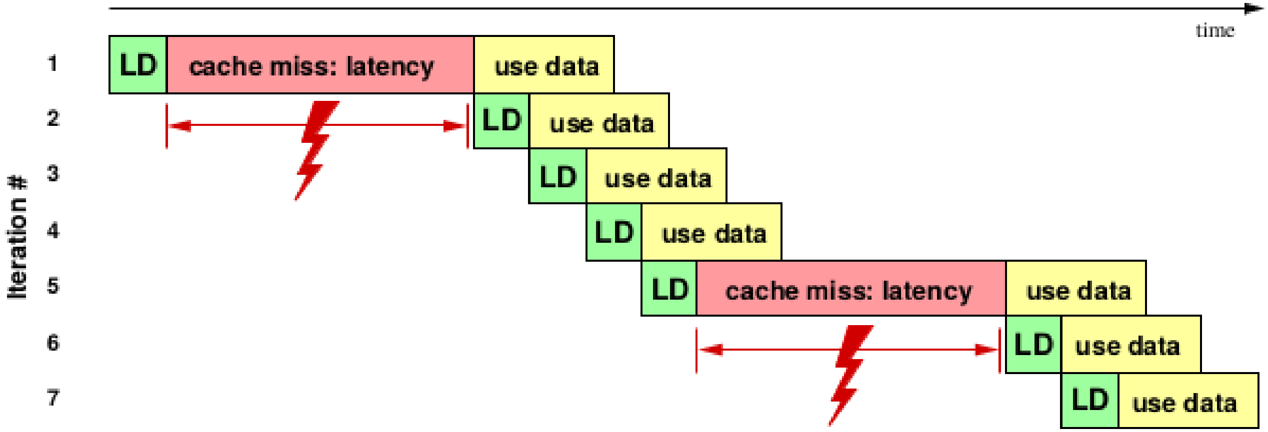
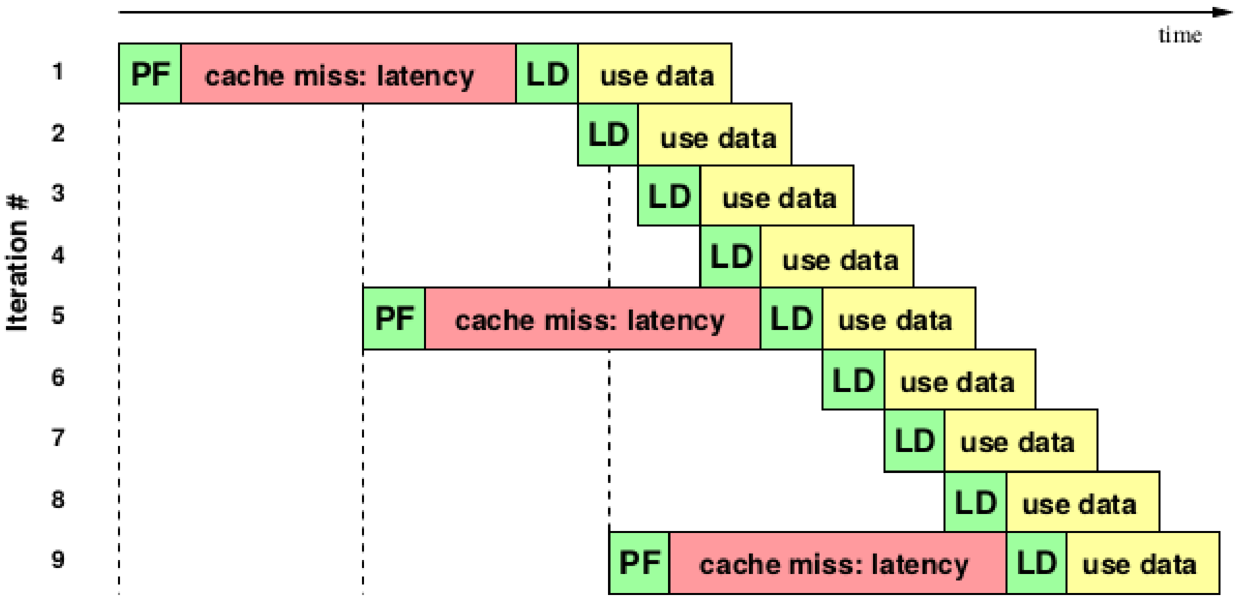


Figure 1.10: Timing diagram on the influence of cache misses and subsequent latency penalties for a vector norm loop. The penalty occurs on each new miss.



Multi-core processors