Optimisation de code et parallélisme

Une introduction

Georges-André Silber

2024/2025

École des mines de Paris

« *Il faut toujours plus de temps que prévu, même en tenant compte de la loi de Hofstadter.* »

— Douglas Hofstadter, *Gödel, Escher, Bach*, 1979.

« *Software expands to fill the available memory. (Parkinson)* »

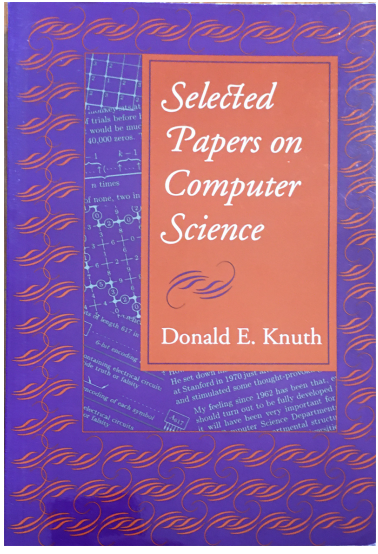
« *Software is getting slower more rapidly than hardware is becoming faster. (Reiser)* »

— Niklaus Wirth, *A Plea for Lean Software*, 1995.

« *Software : it's a gas !* »

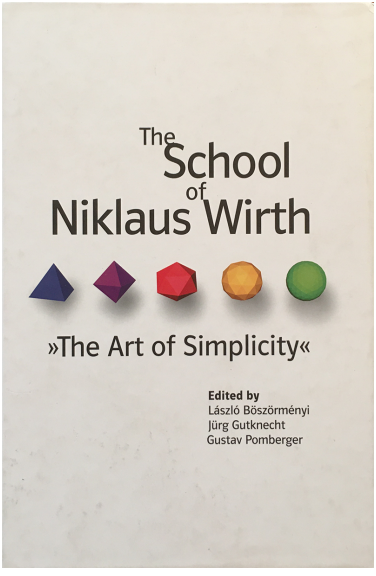
— Nathan P. Myhrvold, *The Next Fifty Years of Software*, 1997.

Donald E. Knuth : *theory and practice* (A.M. Turing Award 1974)



Challenge de Knuth : « *make a thorough analysis of everything your computer does during one second of computation.* »

Niklaus Wirth : *the art of simplicity* (A.M. Turing Award 1984)

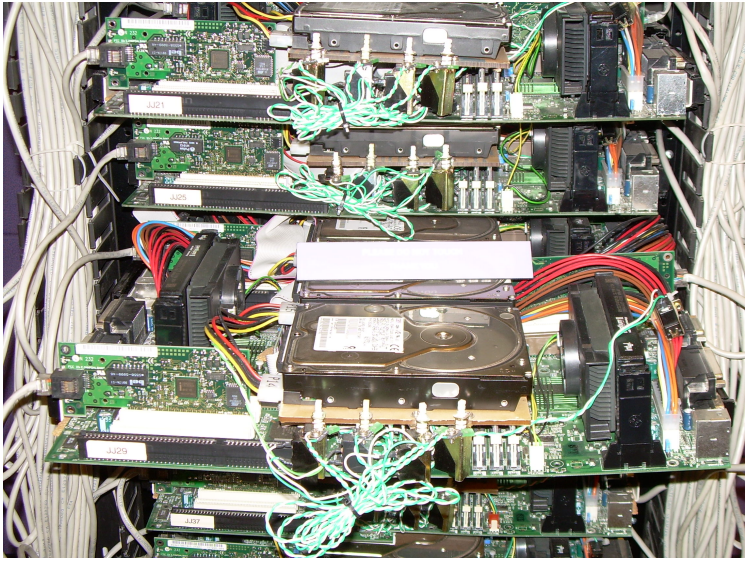


Simple vs complex. Essential vs Ephemeral. A tools is counterproductive when a large part of the entire project is taken up by mastering the tool. One learns best when inventing : every single project as a *learning experiment*.

High Performance Computing et TOP500

Qu'est-ce qu'un supercalculateur ?

- Des *chips* du commerce (avec plusieurs cœurs);
- En ajoutant un ou des GPU on obtient un *nœud* (serveur);
- Les serveurs sont rassemblés dans des armoires (*racks*);
- Puis interconnectés avec des *switchs*.


J325
J326
J329
J330

Exaflop ?

- 1 flop : une addition ou multiplication d'un flottant sur 64 bits
- 1 Eflop : 10^{18} flop (un milliard de milliards)
- 1 Eflop/s : un milliard de milliards d'opérations flottantes par seconde

Frontier : champion du Top 500 (novembre 2022)

- Oak Ridge National Laboratory (USA)
- Performance crête théorique de 1,6 Eflop/s
- Chaque noeud a :
  - 1 CPU AMD EPYC 7A53 (x64) avec 64 cœurs : 2 Tflop/s (< 1% de la perf.)
  - 4 GPU AMD Instinct MI250X avec 220 cœurs : 4 × 53 Tflop/s (99% de la perf.)
  - 730 Go de RAM
  - 2 To de disque NVMe
- 9 408 nœuds (602 112 cœurs CPU, 8 279 040 cœurs GPU)
- Performance mesurée (benchmark HPL) : 1,1 Eflop/s

Benchmark HPL

- High Performance Linpack
- https://netlib.org/benchmark/hpl/
- C, MPI
- $Ax = b$ où A est une matrice carrée de taille $N \times N$
- Résolution d'un système linéaire dense (LU)
- Flottants double précision (64 bits)
- On ne compte que les calculs, données matricielles générées "en place"

Top 500 : HPL (novembre 2022)

PRESENTED BY

ICL INNOVATIVE COMPUTING LABORATORY

BERKELEY LAB Lawrence Berkeley National Laboratory

ISC GROUP Moving Forward.

FIND OUT MORE AT top500.org

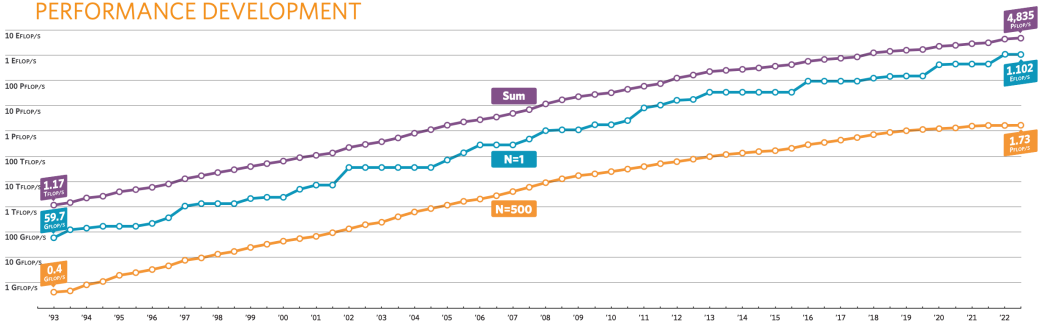
NOVEMBER 2022

			SITE	COUNTRY	CORES	RMAX PFLOP/S	POWER MW
1	**Frontier**	HPE Cray EX235a, AMD Opt 3rd Gen EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-10	DOE/SC/ORNL	USA	8,730,112	1,102.0	21.1
2	**Fugaku**	Fujitsu A64FX (48C, 2.2GHz), Tofu Interconnect D	RIKEN R-CCS	Japan	7,630,848	442.0	29.9
3	**LUMI**	HPE Cray EX235a, AMD Opt 3rd Gen EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-10	EuroHPC/CSC	Finland	2,174,976	304.2	5.82
4	**Leonardo**	Atos Bullsequana intelXeon (32C, 2.6 GHz), NVIDIA A100 quad-rail NVIDIA HDR100 Infiniband	EuroHPC/CINEC	Italy	1,463,616	174.7	5.61
5	**Summit**	IBM POWER9 (22C, 3.07GHz), NVIDIA Volta GV100 (80C), Dual-Rail Mellanox EDR Infiniband	DOE/SC/ORNL	USA	2,414,592	148.6	10.1

Top 11 du Top 500 : HPL (novembre 2022)

#	Machine	Pays	Rpeak (Tflop/s)	Rmax (Tflop/s)	% Rpeak	MW	Eff. (Gflop/s/W)
1	Frontier	USA	1685	1102	65	21.1	52.23
2	Fugaku	Japan	537	442	82	29.9	14.78
3	LUMI	Finland	428	309	72	6.0	51.38
4	Leonardo	Italy	255	174	68	5.6	31.14
5	Summit	USA	200	148	74	10.1	14.72
6	Sierra	USA	125	94	75	7.4	12.72
7	Sunway	China	125	93	74	15.4	6.05
8	Perlmutter	USA	93	70	75	2.6	27.37
9	Selene	USA	79	63	80	2.6	23.98
10	Tianhe-2A	China	100	61	61	18.5	3.32
11	Adastra	France	61	46	74	0.9	50.03

Top 500 : évolution de la performance



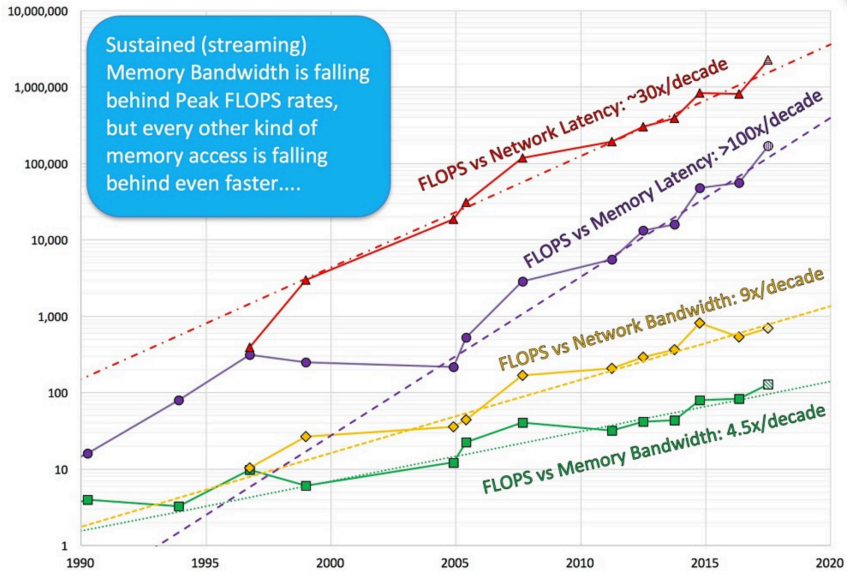
Benchmark HPCG

- High Performance Conjugate Gradients
- Multigrid preconditioned conjugate gradient
- `https://www.hpcg-benchmark.org/`
- C++, OpenMP, MPI, CUDA
- Algèbre linéaire ”creuse”

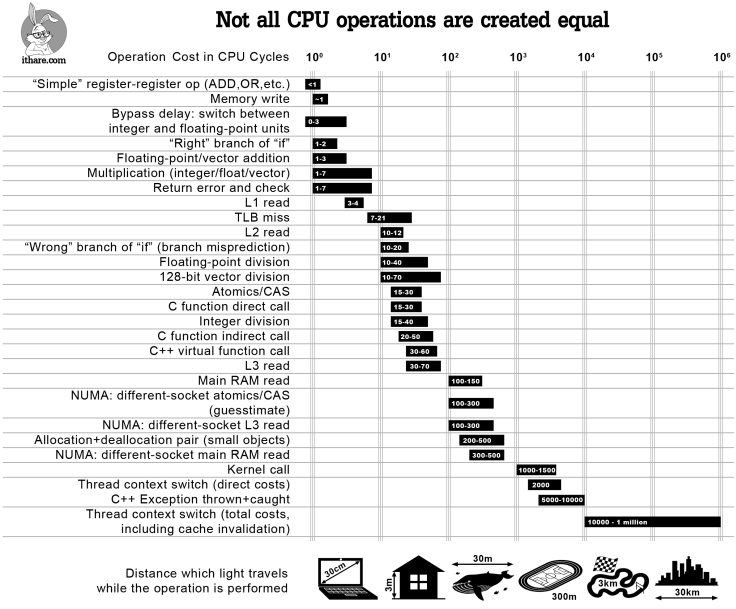
Top 500 : HPCG (novembre 2022)

#	Machine	Pays	Rpeak (Tflop/s)	Rmax (Tflop/s)	% Rpeak	MW	Eff. (Gflop/s/W)
1	Fugaku	Japan	537	16.0	3.0	29.9	0.54
2	Frontier	USA	1685	14.05	0.8	21.1	0.67
3	LUMI	Finland	428	3.41	0.8	6.0	0.57
4	Summit	USA	200	2.93	1.5	10.1	0.29
5	Leonardo	Italy	255	2.57	1.0	5.6	0.46
6	Perlmutter	USA	93	1.91	2.0	2.6	0.74
7	Sierra	USA	125	1.8	1.4	7.4	0.24
8	Selene	USA	79	1.62	2.0	2.6	0.61
9	JUWELS	Germany	70	1.28	1.8	1.8	0.72
10	HPC5	Italy	51	0.86	1.7	2.3	0.38
11	Wisteria	Japan	25	0.82	3.2	1.5	0.56

Flop vs data access



Coûts en cycles des opérations CPU



Top 500 : Green 500 (novembre 2022)

#	# T500	Machine	Pays	Rpeak (Tflop/s)	Rmax (Tflop/s)	% Rpeak	MW	Eff. (Gflop/s/W)
1	405	Henri	USA	5	2	37	0.03	65
2	32	Frontier TDS	USA	23	19	83	0.31	63
3	11	Adastra	France	61	46	74	0.92	58
4	15	Setonix	Australia	34	27	77	0.48	57
5	68	Dardel	Sweden	10	8	80	0.15	56
6	1	Frontier	USA	1685	1102	65	21.1	52
7	3	LUMI	Finland	428	309	72	6.02	51
8	159	ATOS THX	France	4	3	70	0.09	41
9	359	MN-3	Japan	3	2	65	0.05	41
10	331	Champollion	France	2	2	92	0.06	39
11	349	SSC-21	S. Korea	2	2	87	0.1	34

Jack Dongarra (UTK) : A.M. Turing Award 2021

DR. JACK DONGARRA

United States – 2021

CITATION

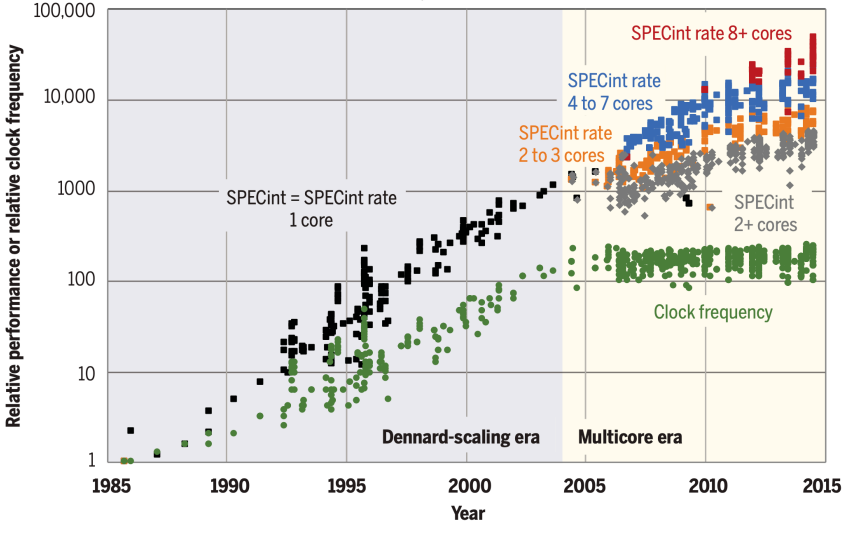
For his pioneering contributions to numerical algorithms and libraries that enabled high performance computational software to keep pace with exponential hardware improvements for over four decades

There is plenty of room (for improvements)

1959 : *There's Plenty of Room at the Bottom*

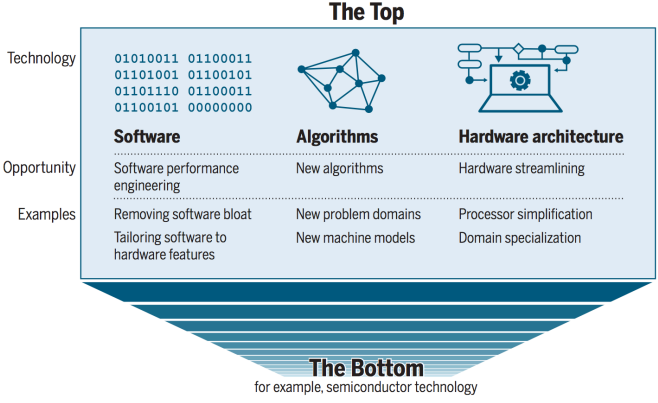
- Phrase de Richard Feynman
- **Miniaturisation** des **semi-conducteurs** : moteur de l'accroissement des performances pendant 50 ans
- Loi de **Moore** : **doublement** du nombre de transistors par puce tous les **deux ans**
- Loi de **Dennard** (MOSFET) : **même niveau** de consommation d'**énergie** à **surface constante**
- Ces deux **lois empiriques** sont aujourd'hui **fausses**

Fin de la loi de Dennard (2004)

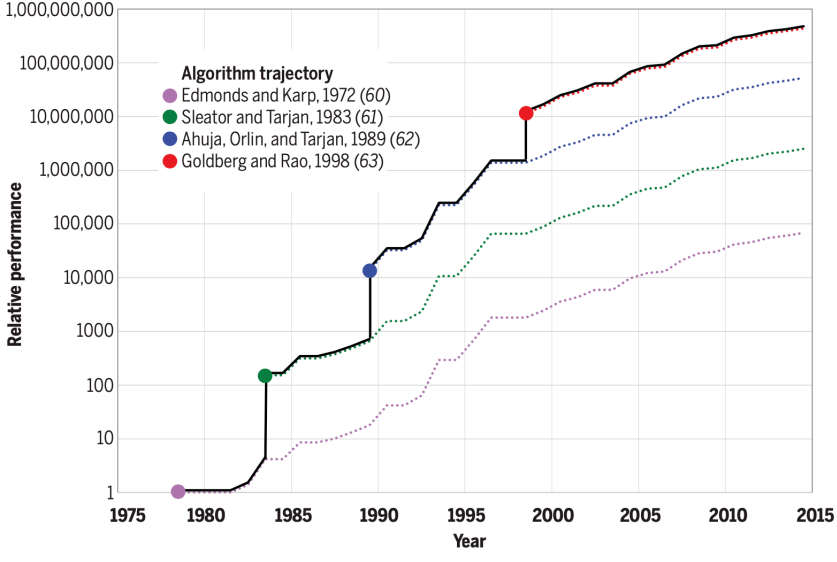


Leiserson (2020) : *There's Plenty of Room at the Top*

There's plenty of room at the Top : What will drive computer's performance after Moore's law ?
C.E. Leiserson et al., Science 2020.



Leiserson (2020) : avancées algorithmiques (ex. Max. Flow Problem)



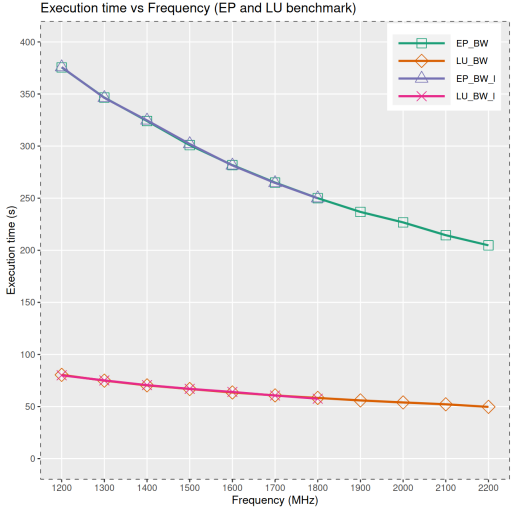
Leiserson (2020) : optimisation de code et parallélisation

```
for i in range(4096):
  for j in range(4096):
    for k in range(4096):
      C[i][j] += A[i][k] * B[k][j]
```

#	Méthode	Temps (s)	Gflop/s	Acc.	Acc. rel.	% peak
1	Python	25 552,48	0,005	1	—	0,00
2	Java	2 372,68	0,058	11	10,8	0,01
3	C	542,67	0,253	47	4,4	0,03
4	Parallel loops	69,80	1,969	366	7,8	0,24
5	Parallel divide and conquer	3,80	36,180	6 727	18,4	4,33
6	plus vectorization	1,10	124,914	23 224	3,5	14,96
7	plus AVX intrinsics	0,41	337,812	62 806	2,7	40,45

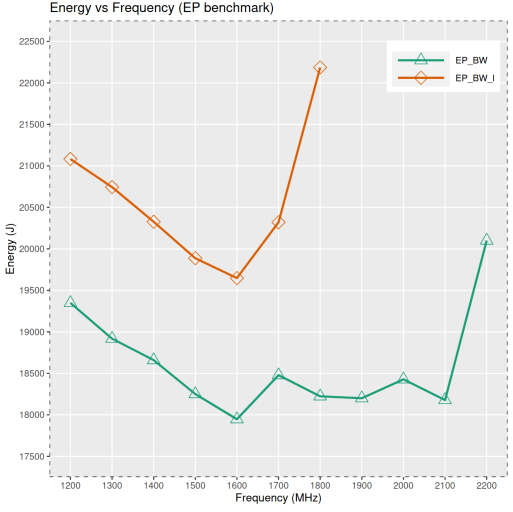
Modèles de coût en temps et en énergie

Temps d'exécution / fréquence du CPU



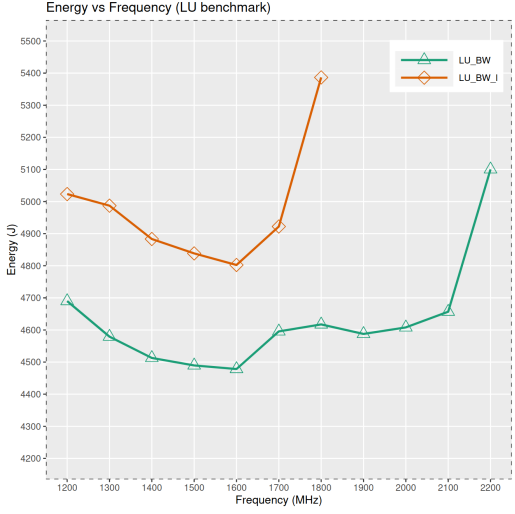
Source : *Experimental Workflow for Energy and Temperature Profiling on HPC Systems*, K. R. Vaddina et al., 2021.

Énergie / fréquence du CPU (PE)



Source : *Experimental Workflow for Energy and Temperature Profiling on HPC Systems*, K. R. Vaddina et al., 2021.

Énergie / fréquence du CPU (LU)



Source : *Experimental Workflow for Energy and Temperature Profiling on HPC Systems*, K. R. Vaddina et al., 2021.

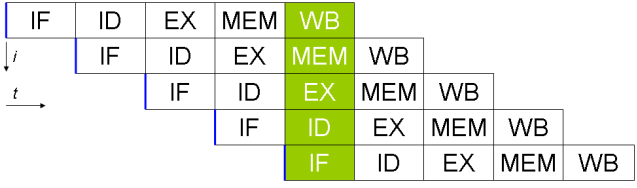
Instruction Level Parallelism (ILP)

Éxécution d'une instruction RISC

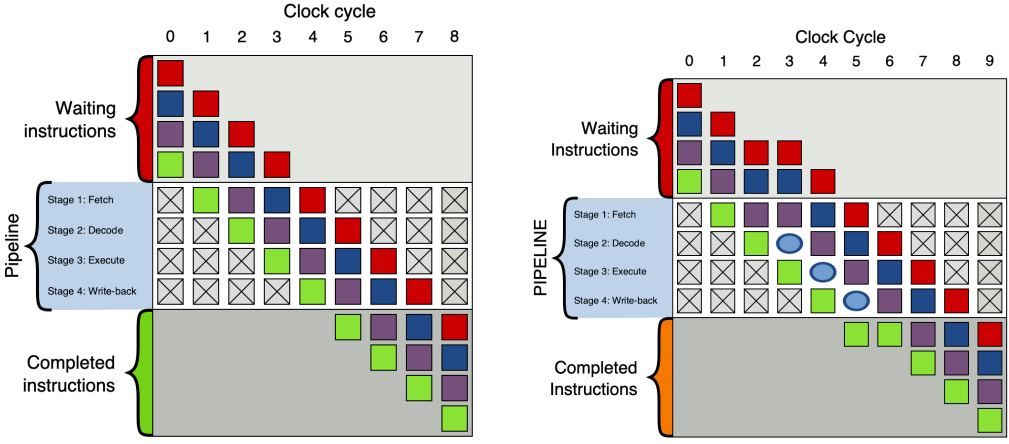
- **Découpage** de l'exécution d'une **instruction** en *n* étapes
  - `IF` : *Instruction fetch*, `pc++`
  - `ID` : *Instruction decode* / *Register fetch* (ou branchement)
  - `EX` : *Execute* / *Effective address*
  - `MEM` : *Memory access* (load ou store)
  - `WB` : *Write back* (dans un registre)

Pipeline

- Étapes exécutées en un **temps identique**
- Sur des éléments fonctionnels **distincts**
- Exécution en **parallèle**



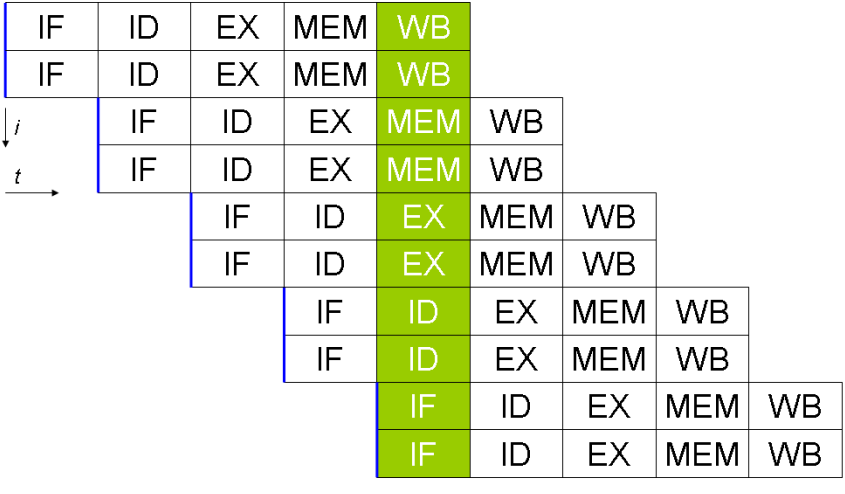
Exemple : pipeline à 4 étages



1

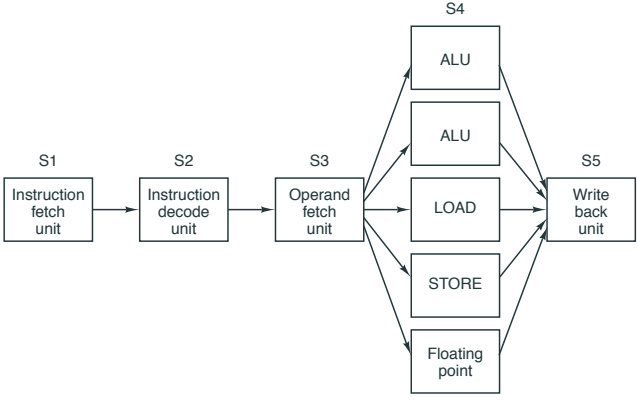
1. Source : https://commons.wikimedia.org/w/index.php?curid=1499754 (CC BY-SA 3.0)

Pipelines de processeur *superscalaire*



Very Large Instruction Word

- Intel Itanium (IA-64) [*stoppé en 2021*]
- MPPA de Kalray

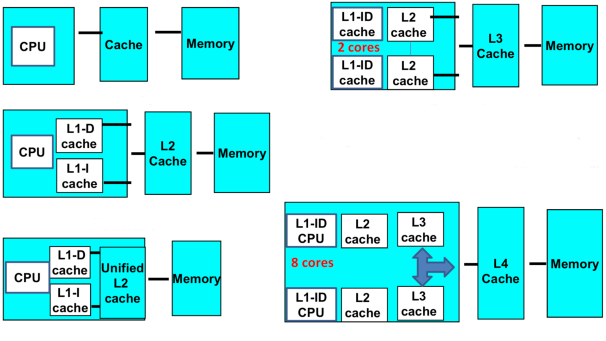


Le mur de la mémoire

CPU vs DRAM

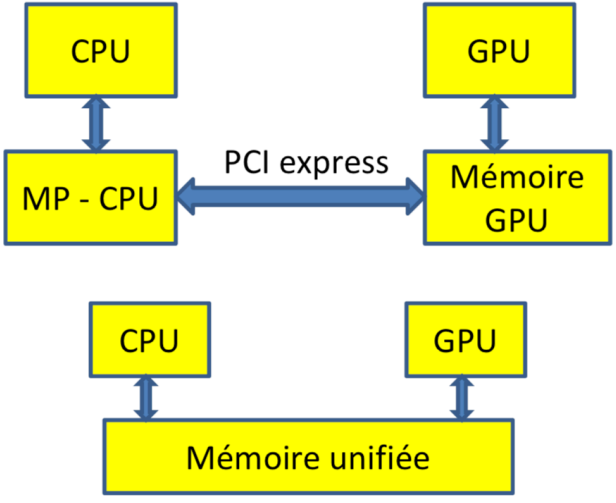
- La vitesse des **CPU** a cru beaucoup plus vite que celle des **DRAM**
- Les **caches** permettent d'atténuer le problème
- → répartir le travail sur plusieurs **processeurs**

Caches



On the complexity of cache analysis for different replacement policies, D. Monniaux et V. Touzeau, 2019.

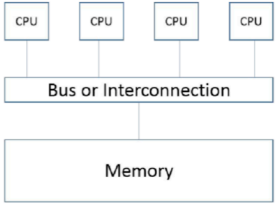
Liaison CPU / GPU



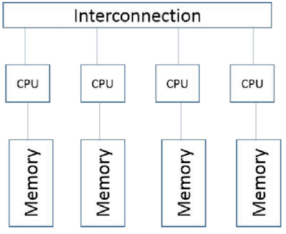
Thread / Process Level Parallelism

Interconnexions

OpenMP
Pthreads



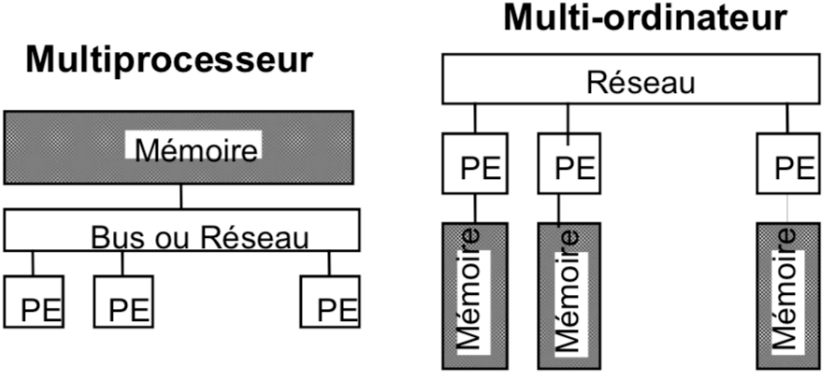
Multiprocesseurs



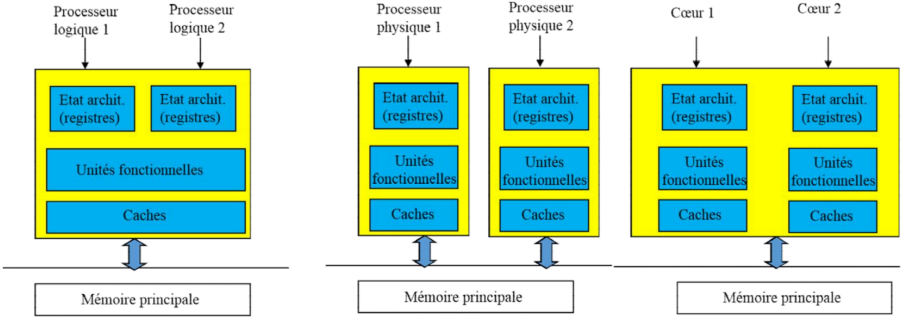
Multi-ordinateurs

MPI

Multiprocesseurs vs Cluster



Synthèse


Processeur logique 1
Processeur logique 2
Etat archit. (registres)
Etat archit. (registres)
Unités fonctionnelles
Caches
Mémoire principale
Multithread
Processeur physique 1
Processeur physique 2
Etat archit. (registres)
Etat archit. (registres)
Unités fonctionnelles
Unités fonctionnelles
Caches
Caches
Mémoire principale
Multiprocesseur
Cœur 1
Cœur 2
Etat archit. (registres)
Etat archit. (registres)
Unités fonctionnelles
Unités fonctionnelles
Caches
Caches
Mémoire principale
Multi-cœur


Multithreaded programming
Theory
Actual
the9gag.com