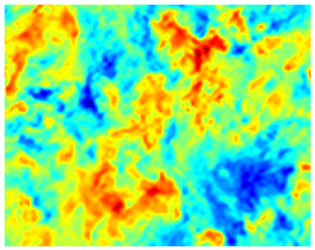
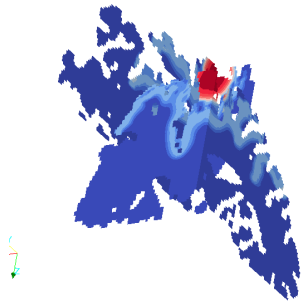
Architecture des calculateurs

Violaine Louvet [1]

[1] Institut Camille jordan - CNRS

Ecole Doctorale 2014-2015





OBJECTIFS DE CE MODULE

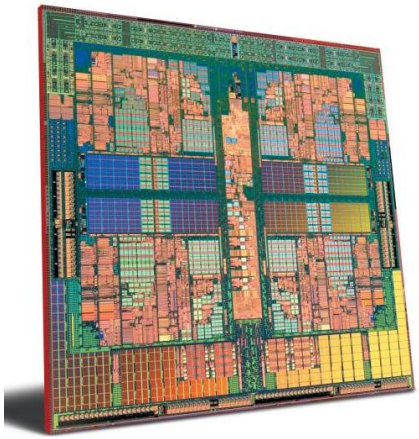
Modéliser et simuler un problème scientifique issu de la physique, biologie, chimie, ...

- Comprendre l'architecture
- Adapter les méthodes numériques, les algorithmes les structures de données et la programmation
- Comprendre et expliquer le comportement d'un programme
- Optimiser les codes de calcul
- Réaliser des codes portables et réutilisables
- Paralléliser

Importance des aspects matériels

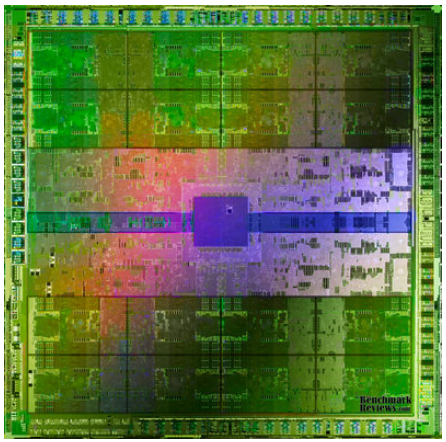
La connaissance des architectures de calcul permet :

- de choisir et d'adapter son matériel en fonction de ses besoins
- de le dimensionner correctement en retenant le meilleur compromis : architecture équilibrée
- de comprendre le comportement d'un programme
- d'adapter les méthodes numériques, les algorithmes, la programmation



Multi-cœurs

Blue-gene P



GPU

Sommaire

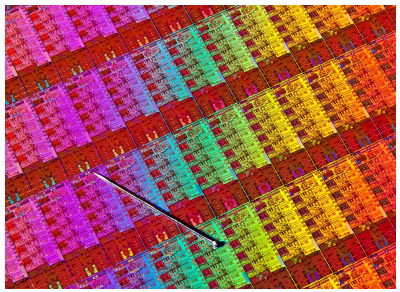
1. Architecture générale
   - Processeur
   - Mémoire
   - Communications internes et réseaux
   - Stockage
2. Tendances
   - Evolutions technologiques et problématiques associées
   - GPU
   - Many-cœurs
3. Moyens de calcul disponibles
   - Micro éclairage sur le HPC mondial
   - Pyramide des moyens de calcul : paysages européen et français
   - Les grilles de calcul
   - Comment accéder à ces ressources ?
4. Concepts du parallélisme
   - Introduction
   - Classification et terminologie
   - Efficacité du parallélisme
   - Modèles de programmation parallèle
5. Conclusions

Sommaire

1. Architecture générale
   - Processeur
   - Mémoire
   - Communications internes et réseaux
   - Stockage
2. Tendances
   - Evolutions technologiques et problématiques associées
   - GPU
   - Many-cœurs
3. Moyens de calcul disponibles
   - Micro éclairage sur le HPC mondial
   - Pyramide des moyens de calcul : paysages européen et français
   - Les grilles de calcul
   - Comment accéder à ces ressources ?
4. Concepts du parallélisme
   - Introduction
   - Classification et terminologie
   - Efficacité du parallélisme
   - Modèles de programmation parallèle
5. Conclusions

Qu'y a-t-il dans un ordinateur ?

- Des transistors connectés sur des circuits intégrés (puces ou chip)
  Selon la tension qu'on leur applique, laissent ou non passer le courant : information de type 1 ou 0



Qu'y a-t-il dans un ordinateur ?

- Des transistors connectés sur des circuits intégrés (puces ou chip)
  Selon la tension qu'on leur applique, laissent ou non passer le courant : information de type 1 ou 0
- Quels composants sur cette carte mère ?

Bits, octets & co

- bit (b) : binary digit, plus petite unité d'information d'un composant en informatique. Vaut 0 ou 1
  - 1 bit permet donc d'avoir 2 états (0 ou 1), 2 bits permettent d'avoir 4 (2^2) états (00,01,10,11),... , n bits permettent ainsi d'avoir 2^n états
- octet (o) ou byte (B) = composé de 8 bits

Normalisation 1998

Depuis 1998 les préfixes binaires associés aux octets ou bytes ont été normalisés (kibi = kilo binaire, mébi = méga binaire ...) :

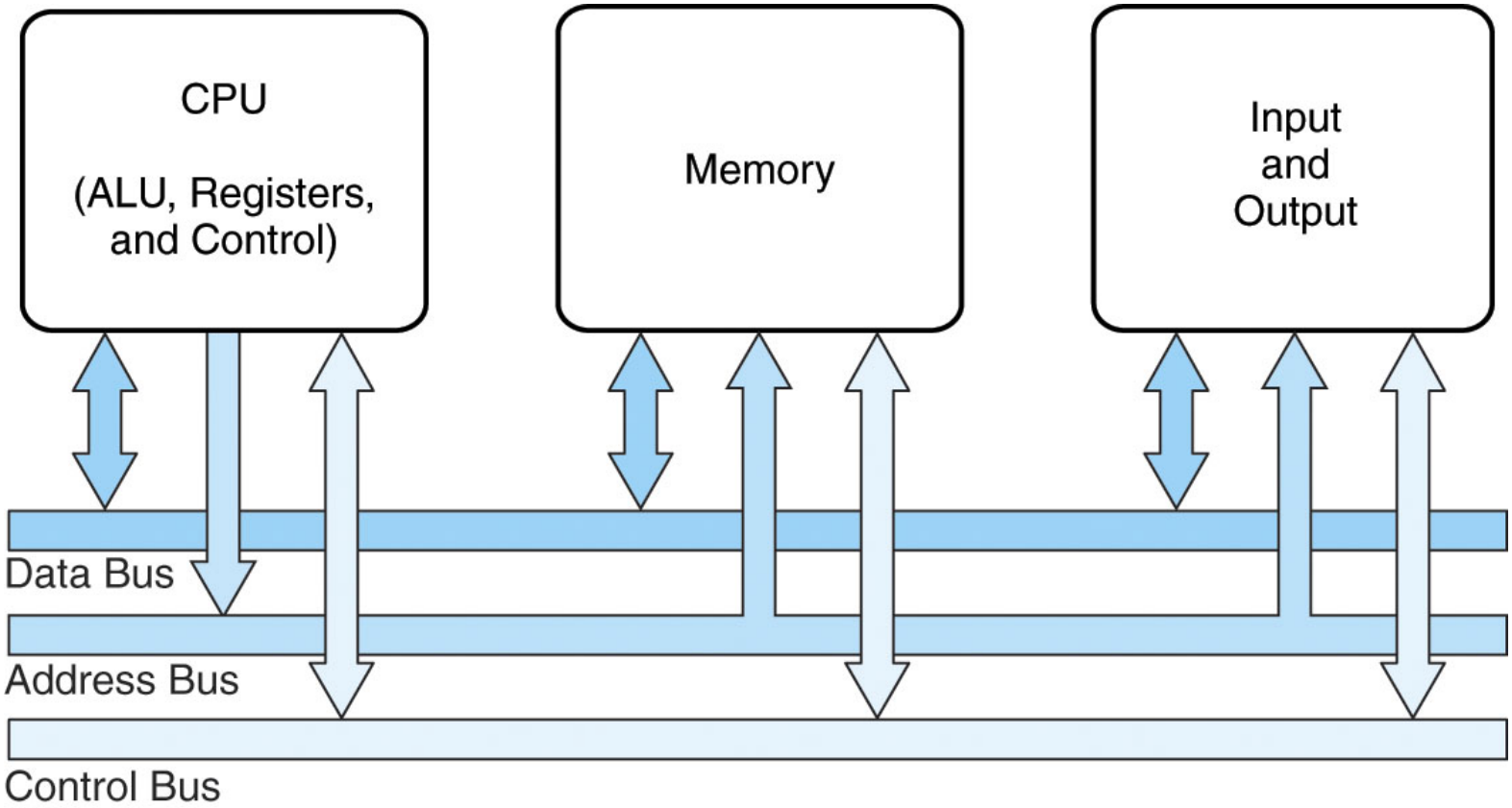
- $1 kibio = 2^{10}$ octets $= 1024$ octets $= 8192$ bits
- $1 ko = 1000$ octets

Mais attention, de façon courante : 1 kilo-octet $= 1 Ko = 1024$ octets

Usage

- Processeurs et mémoires opèrent sur des octets
- Débits des réseaux et des bus exprimés en bits par seconde

Architecture générale



Architecture de Von Neumann

Sommaire

1. Architecture générale
   - Processeur
   - Mémoire
   - Communications internes et réseaux
   - Stockage
2. Tendances
   - Evolutions technologiques et problématiques associées
   - GPU
   - Many-cœurs
3. Moyens de calcul disponibles
   - Micro éclairage sur le HPC mondial
   - Pyramide des moyens de calcul : paysages européen et français
   - Les grilles de calcul
   - Comment accéder à ces ressources ?
4. Concepts du parallélisme
   - Introduction
   - Classification et terminologie
   - Efficacité du parallélisme
   - Modèles de programmation parallèle
5. Conclusions

Caractéristiques d'un processeur

- Quels critères pour le choix d'un processeur ?

Caractéristiques d'un processeur

- Nombre de cœurs : le processeur peut intégrer plusieurs noyaux de calcul appelés cœurs

Caractéristiques d'un processeur

- Nombre de cœurs : le processeur peut intégrer plusieurs noyaux de calcul appelés cœurs
- Fréquence d'horloge (MHz) : vitesse de fonctionnement du processeur = nombre de millions de cycles que le processeur est capable d'effectuer par seconde
  - Cycle = plus petite unité de temps au niveau du processeur. Chaque opération/instruction nécessite au minimum un cycle, et plus souvent plusieurs
  - $1 GHz = 10^9 Hz = 10^9 \text{cycle/s}$

Caractéristiques d'un processeur

- Nombre de cœurs : le processeur peut intégrer plusieurs noyaux de calcul appelés cœurs
- Fréquence d'horloge (MHz) : vitesse de fonctionnement du processeur = nombre de millions de cycles que le processeur est capable d'effectuer par seconde
- Jeu d'instructions : ensemble des opérations qu'un processeur peut exécuter, plus ou moins complexes

Caractéristiques d'un processeur

- Nombre de cœurs : le processeur peut intégrer plusieurs noyaux de calcul appelés cœurs
- Fréquence d'horloge (MHz) : vitesse de fonctionnement du processeur = nombre de millions de cycles que le processeur est capable d'effectuer par seconde
- Jeu d'instructions : ensemble des opérations qu'un processeur peut exécuter, plus ou moins complexes
- Largeur (32 ou 64 bits) : notamment du bus de données et des registres internes. Bon indicateur de la quantité d'information que celui-ci peut gérer en un temps donné

Adressage mémoire

Les processeurs 32 bits ne peuvent pas adresser plus de 4 gigaoctets (2^{32}) de mémoire centrale, tandis que les processeurs 64 bits peuvent adresser 16 exaoctets (2^{64}) de mémoire.

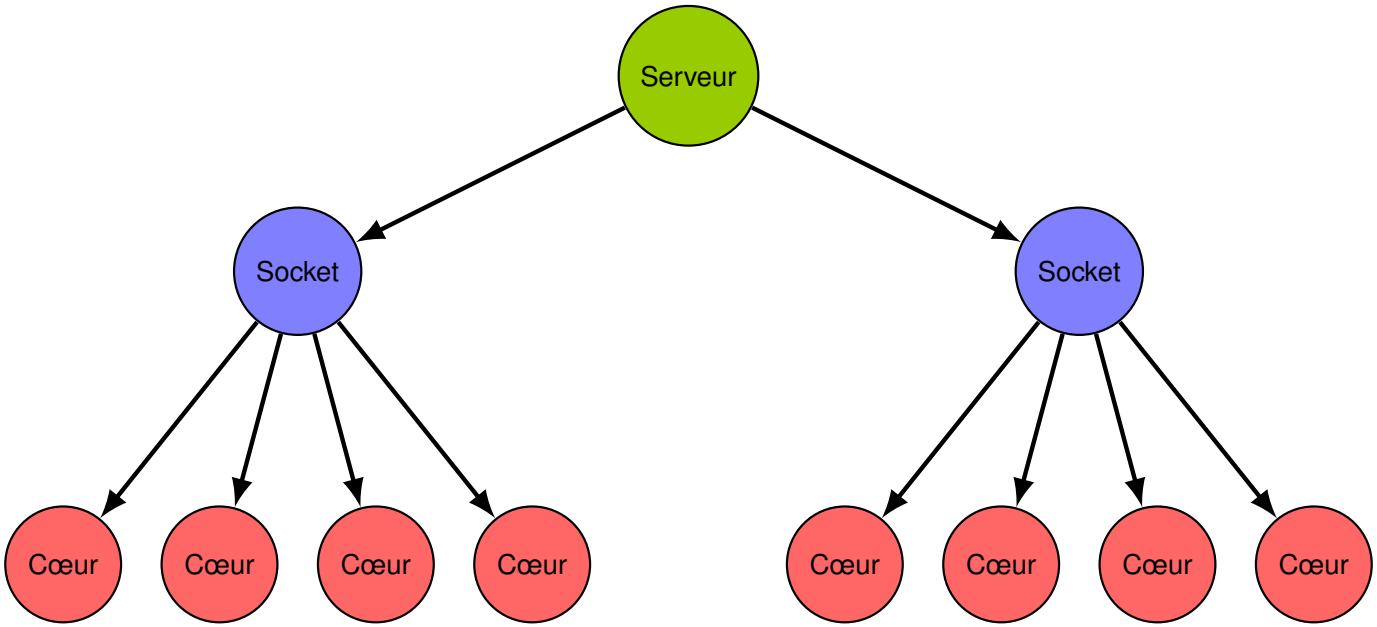
CPU, processeur, cœur, socket ?

CPU, processeur, cœur, socket ?

CPU, Processeur : Central Processing Unit, partie de l'ordinateur en charge du traitement et de l'exécution des instructions des programmes

Cœur : élément du processeur capable d'exécuter des programmes de façon autonome

Socket : réceptacle du processeur sur la carte mère

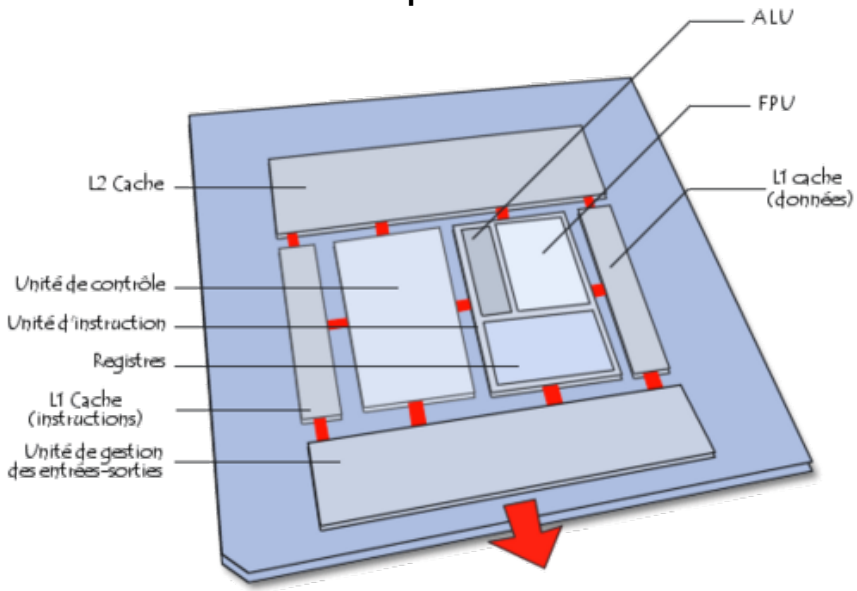


Serveur bi-sockets quadri-cœurs

Composition d'un cœur

Eléments importants du processeur :

- Unité de commande : permet de séquencer le déroulement des instructions
- Unité Arithmétique et Logique (UAL), calculs arithmétiques élémentaires sur des nombres entiers et opérations logiques
- Unité de calcul en virgule flottante (en anglais Floating Point Unit - FPU) : traitement des calculs sur des nombres à virgule
- Registres : mémoires de petite taille (quelques octets) très rapides
- Mémoires caches : diminuent les temps d'accès à la mémoire



FONCTIONNEMENT : LES POINTS IMPORTANTS

- Il existe des mécanismes complexes pour augmenter les performances : pipelining, processeur superscalaire, exécution spéculative, hyperthreading ...

- Instructions vectorielles : le processeur, en fonction de son jeu d'instructions, est capable d'appliquer la même instruction simultanément à plusieurs données (SSE, AVX).

$\rightarrow$ Parallélisme multi-niveaux

Pipeline

- L'exécution d'une instruction est décomposée en une succession d'étapes.
- Chaque étape correspond à l'utilisation d'une des fonctions du micro-processeur.
- Lorsqu'une instruction se trouve dans l'une des étapes, les composants associés aux autres étapes ne sont pas utilisés.

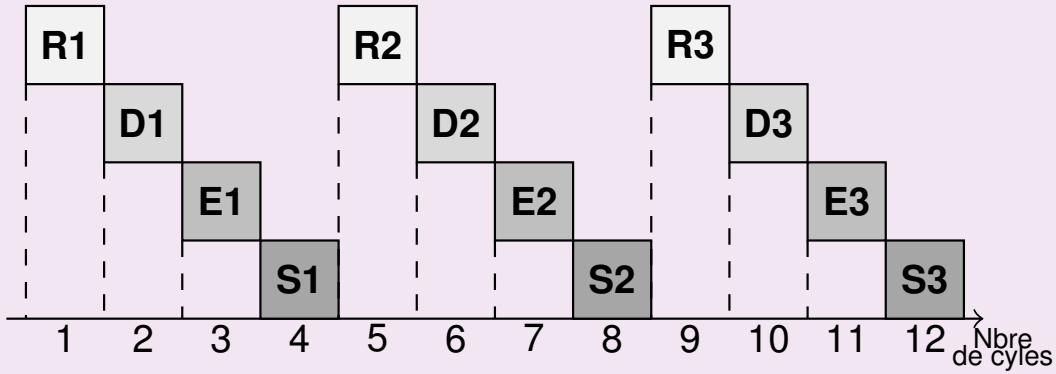
Problèmes potentiels

- Le temps de traitement de chaque étape doit être à peu près égal sinon les unités rapides doivent attendre les plus lentes.
- Des « aléas » si une instruction produit un résultat utilisé par l'instruction suivante par exemple, ou quand deux instructions ont besoin d'utiliser la même ressource.
- Prédiction de branchement : devoir attendre l'adresse de destination du branchement ou faire une prédiction.

Exemple de pipeline à 4 étages

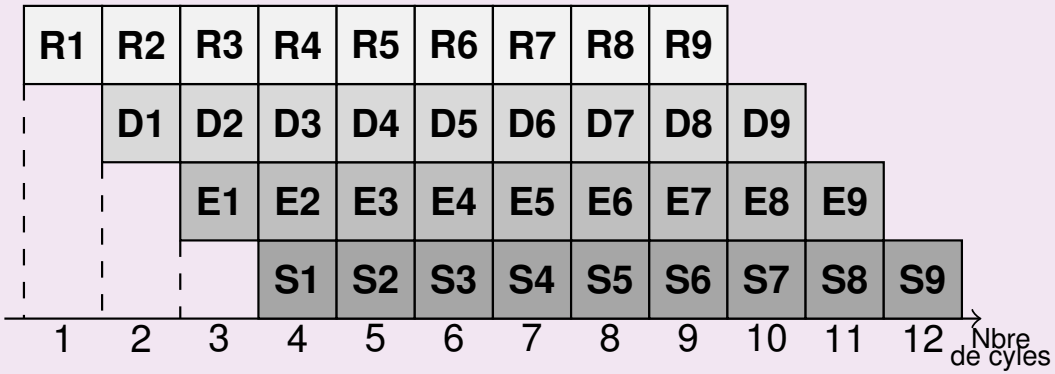
Recherche	Décodage	Exécution	Sauv. résultat

Modèle classique



Exemple de pipeline à 4 étages

Modèle pipeliné à 4 étages



- Chaque « travailleur » effectue le même travail sur des objets différents
- Si il y a 4 étapes dans la chaîne, 4 objets peuvent être traités simultanément
- Et si ces étapes prennent le même temps, la chaîne est continuellement occupée

Jeux d'instructions (ISA, Instructions Set Architecture)

- Ensemble des opérations élémentaires qu'un processeur peut accomplir
- Plusieurs familles de processeurs possédant chacune un jeu d'instructions propre :
  - 80x86 : le « x »représente la famille. On parle ainsi de 386, 486, 586, 686, etc.
  - ARM
  - IA-64
  - PowerPC ...

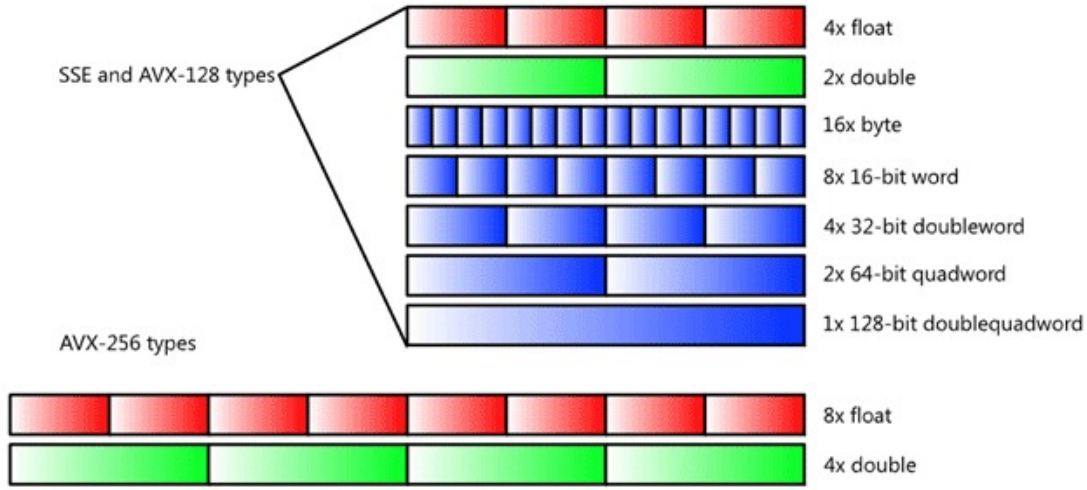
Attention

Un programme réalisé pour un type de processeur ne peut pas fonctionner directement sur un système possédant un autre type de processeur.

Parallélisation de données

SIMD (Single Instruction Multiple Data) : appliquer la même instruction simultanément à plusieurs données

- Jeux d'instructions :
  - SSE (Streaming SIMD Extensions) chez Intel
  - AVX (Advanced Vector Extensions) chez Intel et AMD (prise en charge de registres vectoriels plus grands)
- Capacités vectorielles des processeurs : possibilité d'exécuter une opération arithmétique sur un registre vectoriel (contenant deux DP ou quatre SP en général).



Parallélisation de données

Exemple SSE

Algorithme :

```
for i = 1,n
    x[i] = sqrt(x[i])
```

Travail du processeur :

```
for i = 1,n
    load x[i] to the floating-point register
    calculate the square root
    write the result from the register to memory
```

Avec SSE :

```
for {i1,i2,i3,i4} in {1:n}
    load x[i1],x[i2],x[i3],x[i4] to the SSE register
    calculate 4 square roots in one operation
    write the result from the register to memory
```

Hyperthreading

- SMT = Simultaneous MultiThreading, plus connu sous le nom d'Hyperthreading chez Intel.
- Principe : créer deux cœurs logiques sur une seule puce, chacun doté de ses propres registres. Ces deux unités partagent les éléments du cœur physique comme le cache et le bus système.
- Dégradation des performances individuelles des threads mais amélioration des performances de l'ensemble.

Attention

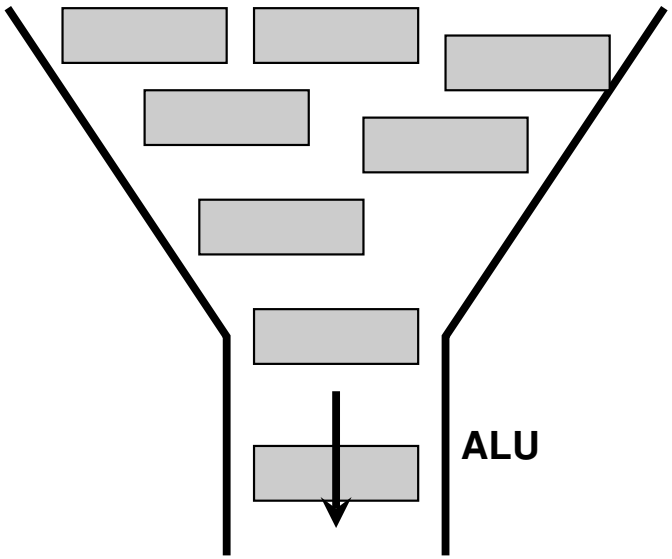
Ce fonctionnement est généralement activé par défaut.

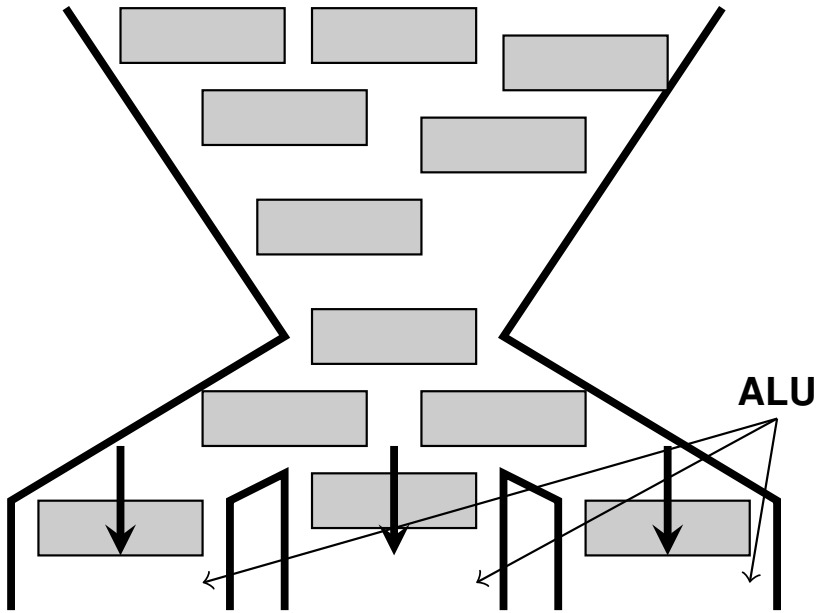
Démo : cpuinfo

Architecture superscalaire

Principe

- Exécuter plusieurs instructions en même temps.
- Doter le micro-processeur de plusieurs unités de traitement travaillant en parallèle
- Nécessite de disposer d'un flot de données/instructions important





Flops et performance du processeur

- La performance s'exprime en opérations (additions ou multiplications) à virgule flottante par seconde = FLoating point Operations Per Second
- Puissance crête (point de vue théorique) : mesure les performance des unités de calcul en virgule flottante (FPU) contenues dans le cœur.
  - Exemple sur un processeur Intel Sandybridge
    - ⋆ fréquence d'un coeur : 3.2GHz
    - ⋆ nombre de coeurs : 6
    - ⋆ registres vectoriels : 16 en simple précision (c'est à dire quand les flottants sont stockés sur 32 bits), 8 en double précision (c'est à dire quand les flottants sont stockés sur 64 bits)

  La performance crête d'un processeur de ce type sera donc :

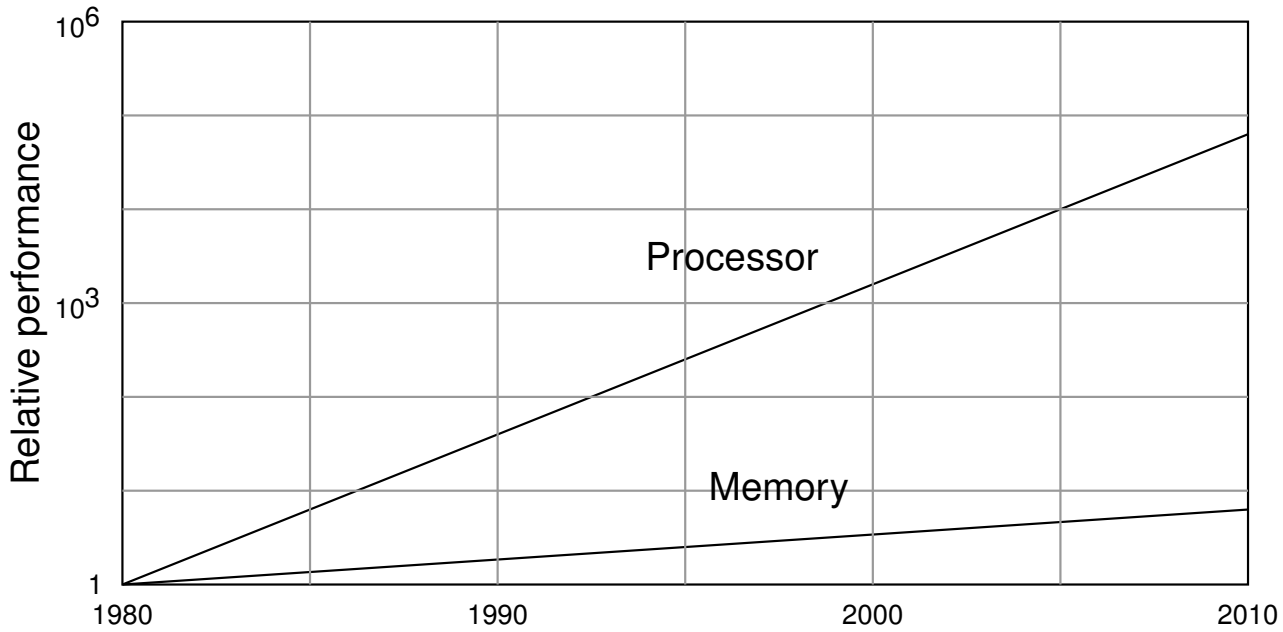
  6 x 3.2 x 16 = 307.2 GFLOPS

- D'un point de vue pratique, la puissance d'une machine dépend de l'ensemble de ses composants : fréquence du processeur, accès mémoire, vitesse des bus, complexité de l'architecture ... mais aussi charge de la machine, système d'exploitation ... On est souvent loin de la puissance théorique ...

Sommaire

1. Architecture générale
   - Processeur
   - Mémoire
   - Communications internes et réseaux
   - Stockage
2. Tendances
   - Evolutions technologiques et problématiques associées
   - GPU
   - Many-cœurs
3. Moyens de calcul disponibles
   - Micro éclairage sur le HPC mondial
   - Pyramide des moyens de calcul : paysages européen et français
   - Les grilles de calcul
   - Comment accéder à ces ressources ?
4. Concepts du parallélisme
   - Introduction
   - Classification et terminologie
   - Efficacité du parallélisme
   - Modèles de programmation parallèle
5. Conclusions

« Memory Wall » : $Vitesse_{CPU} \gg Vitesse_{Memory}$



Définitions

Bande passante

Débit d'informations.

- Mesurée généralement en octets (byte) par seconde (o/s, ou B/s) ou en bits par seconde (bit/s ou bps)

Latence

Temps minimum d'établissement de la connexion : indépendant de la quantité de données à transporter.

- Mesurée généralement en secondes

Problématique

« Memory wall »

- L'accès aux données est un des principaux facteurs limitant la performance
- Les processeurs sont déséquilibrés en ce qui concerne le rapport performance crête / bande passante mémoire

Equilibre d'une machine

- Ratio entre la bande passante mémoire et la performance crête

$$B_m = \frac{\text{memory bandwith (GWords/sec)}}{\text{peak performance (GFlops/sec)}}$$

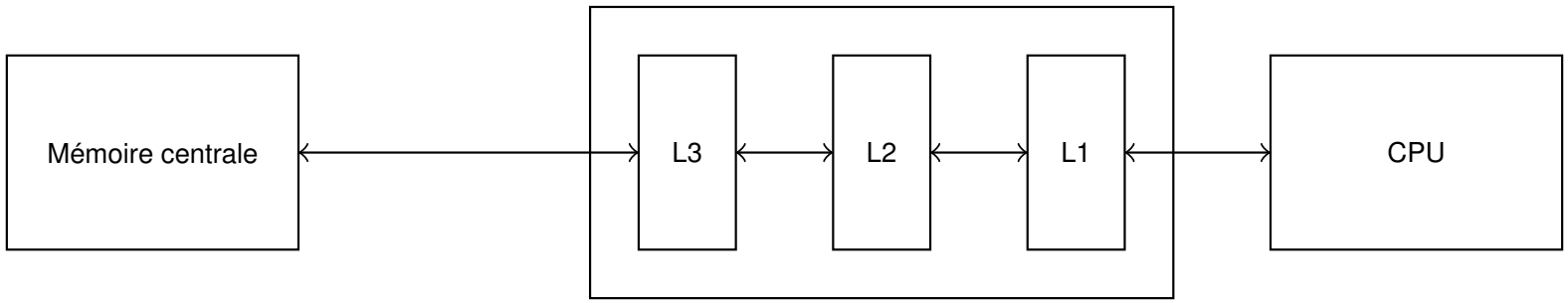
data path	**balance** (W/F)
cache	0.5 - 1.0
machine (RAM)	0.03 - 0.5
réseau rapide	0.001 - 0.02
GBit ethernet	0.0001 - 0.0007
disque	0.0001 - 0.01

Accès mémoire : problématique

Quels facteurs influencent les performances des accès mémoire ?

- Localisation : cache, RAM, disque ?
- Manière dont elles sont accédées :
  - au travers d'un chipset (jeu de composants électroniques qui permet le contrôle des échanges d'information : Northbridge par exemple)
  - directement par le processeur
  - via le processeur voisin ...

Plus une donnée est proche du processeur, plus elle est accédée rapidement.



Principes de localité

Localité spatiale

Lorsqu'un programme accède à une donnée ou à une instruction, il est probable qu'il accédera ensuite aux données ou instructions voisines

Localité temporelle

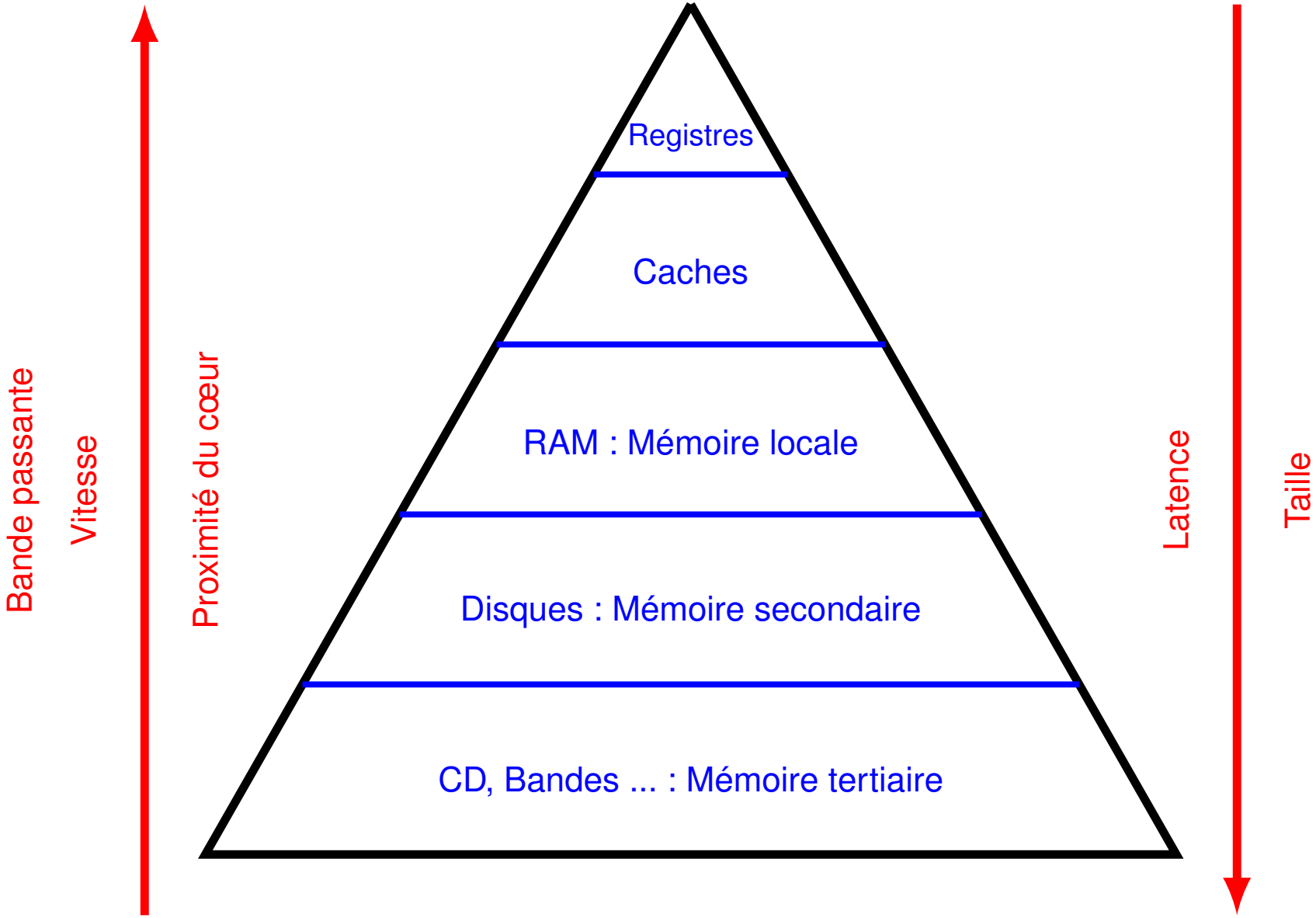
Lorsqu'un programme accède à une donnée ou à une instruction, il est probable qu'il y accédera à nouveau dans un futur proche

Exemple

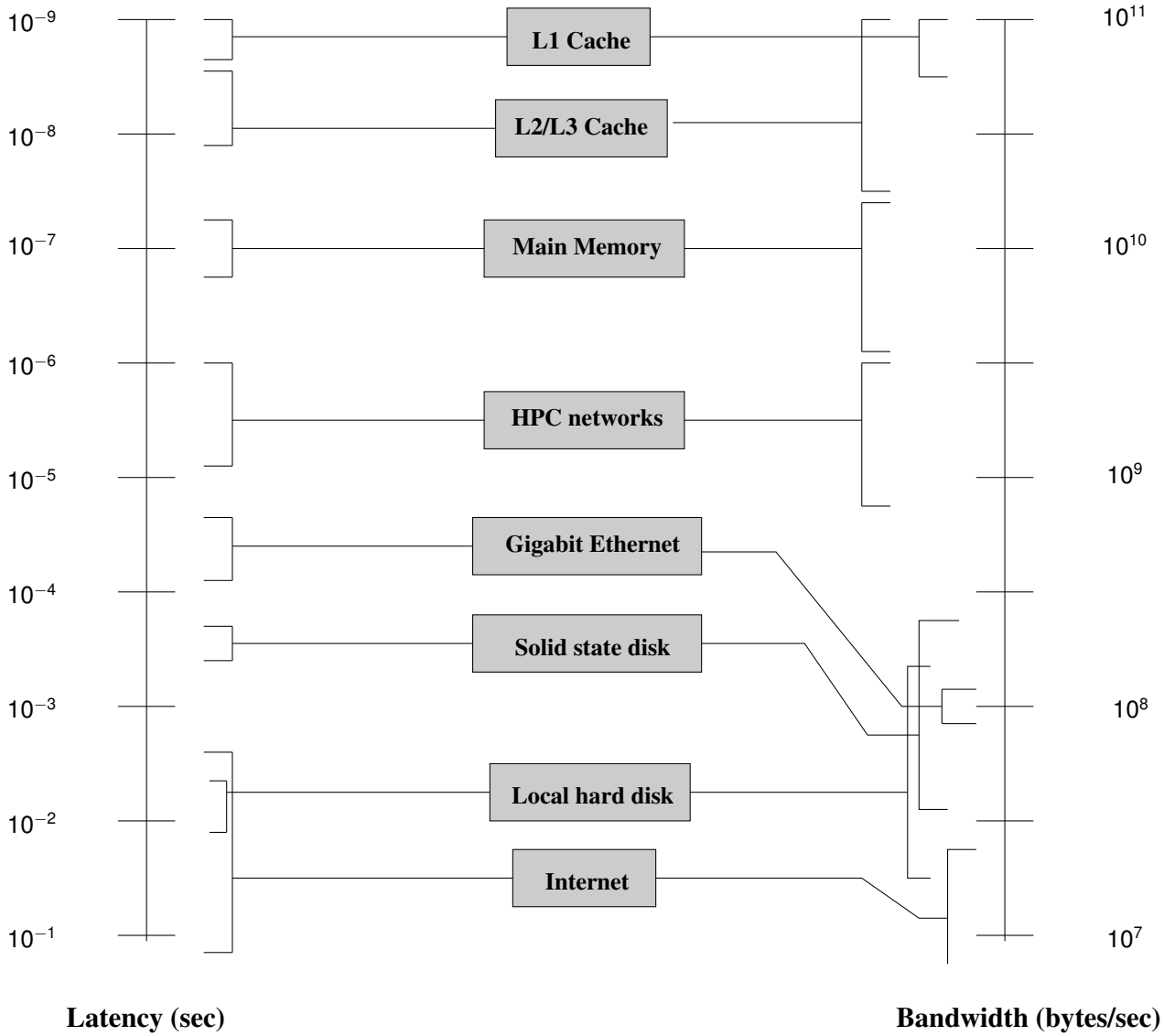
```
subroutine sumVec(vec,n)
    integer :: n
    integer :: vec(n)
    integer :: i,sum=0
    do i = 1,n
        sum = sum + vec(i)
    end do
end subroutine
```

- Bonne localité spatiale des données du tableau `vec` : accès en séquence
- Bonne localité temporelle de la donnée `sum` : accès fréquent
- Bonne localité spatiale et temporelle des instructions : accès en séquence et fréquemment

Hiérarchie Mémoire



Hiérarchie Mémoire



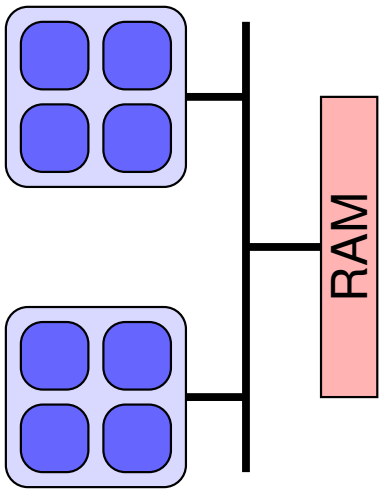
Quelques ordres de grandeur

Type	Taille	Vitesse	Coût/bit
Registre	< 1 KB	< 1 ns	$$$$
SRAM On-chip	8 KB - 6 MB	< 10 ns	$$$
SRAM Off-chip	1 MB - 16 MB	< 20 ns	$$
DRAM	64 MB - 1 TB	< 100 ns	$
Flash	64 MB - 32 GB	< 100 μs	c
Disk	40 GB - 1 PB	< 20 ms	$\sim$0

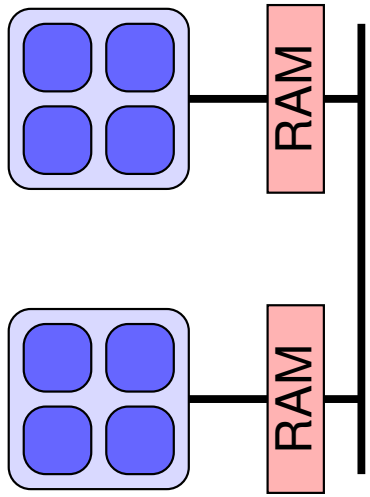
UMA/NUMA

- Modèle SMP : relie plusieurs processeurs à une seule mémoire centralisée. Architecture UMA (Uniform Memory Access) : le temps nécessaire pour accéder à un emplacement quelconque en mémoire est le même pour tous les processeurs.
- Modèle NUMA (Non Uniform Memory Access) : les processeurs sont capables d'accéder à la totalité de la mémoire du système mais les temps d'accès mémoire sont non uniformes en fonction de la distance mémoire/cœur.

Architecture SMP



Architecture NUMA



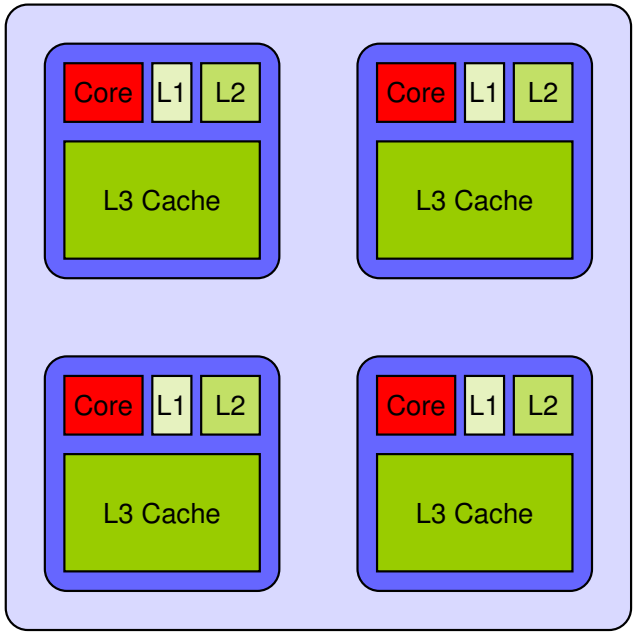
Fonctionnement des caches

- Le cache est divisé en lignes (ou blocs) de mots
- 2 niveaux de granularité :
  - le CPU travaille sur des mots (par ex 32 ou 64 bits)
  - les transferts mémoire se font par ligne (ou bloc, par ex 256 octets)
- Les lignes de caches sont organisées en ensembles à l'intérieur du cache, la taille de ces ensembles est constante et appelée le degré d'associativité.
- Exploitation de la localité spatiale : le cache contient des copies des mots par lignes de cache
- Exploitation de la localité temporelle : choix judicieux des lignes de cache à retirer lorsqu'il faut rajouter une ligne à un cache déjà plein

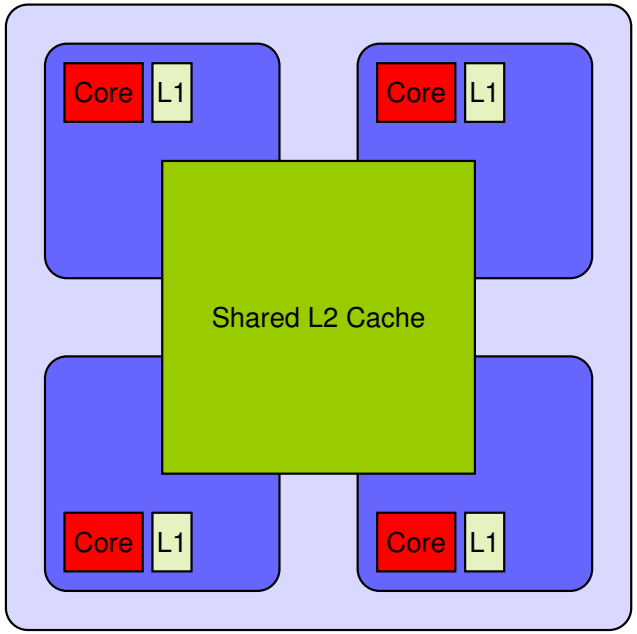
Lorsque le processeur tente d'accéder à une information (instruction ou donnée)

Si l'information se trouve dans le cache (hit), le processeur y accède sans état d'attente, sinon (miss) le cache est chargé avec un bloc d'informations de la mémoire

Organisation des caches



Caches indépendants



Caches partagés

- Problème de cohérence
- Problème d'accès concurrent

Quelles conséquences pour nous ?

Exemple du parcours d'une matrice

- On considère un tableau 2D
- Allocation mémoire : les données sont stockées dans un bloc mémoire contigu sous la forme d'un vecteur
- En C/C++ : stockage des lignes les unes derrière les autres
- En Fortran : stockage des colonnes les unes derrière les autres

for $i \in 1, n$ **do**
  for $j \in 1, n$ **do**
    $a_{i,j} x_j = 0.$
  end for
end for

for $j \in 1, n$ **do**
  for $i \in 1, n$ **do**
    $a_{i,j} = 0$
  end for
end for

Une version est « bonne », l'autre pas ! Démonstration par l'exemple

Page faults / Caches miss

- Défaut de cache (cache miss) : lorsque le processeur essaie d'accéder une donnée qui n'est pas dans le cache
  - Raté obligatoire (compulsory miss) : premier accès au bloc pendant l'exécution du programme. Inévitable.
  - Raté par capacité insuffisante (capacity miss) : accès à un bloc qui a été remplacé par un autre dans le cache car la capacité du cache est insuffisante.
- Défaut de page (page fault) : lorsqu'un processus tente d'accéder à une adresse logique correspondante à une page non résidente en mémoire centrale $\longrightarrow$ interruption du processus le temps de charger la page.
  - Temps de traitement d'un défaut de page : $\sim 25ms$

Sommaire

1. Architecture générale
   - Processeur
   - Mémoire
   - Communications internes et réseaux
   - Stockage
2. Tendances
   - Evolutions technologiques et problématiques associées
   - GPU
   - Many-cœurs
3. Moyens de calcul disponibles
   - Micro éclairage sur le HPC mondial
   - Pyramide des moyens de calcul : paysages européen et français
   - Les grilles de calcul
   - Comment accéder à ces ressources ?
4. Concepts du parallélisme
   - Introduction
   - Classification et terminologie
   - Efficacité du parallélisme
   - Modèles de programmation parallèle
5. Conclusions

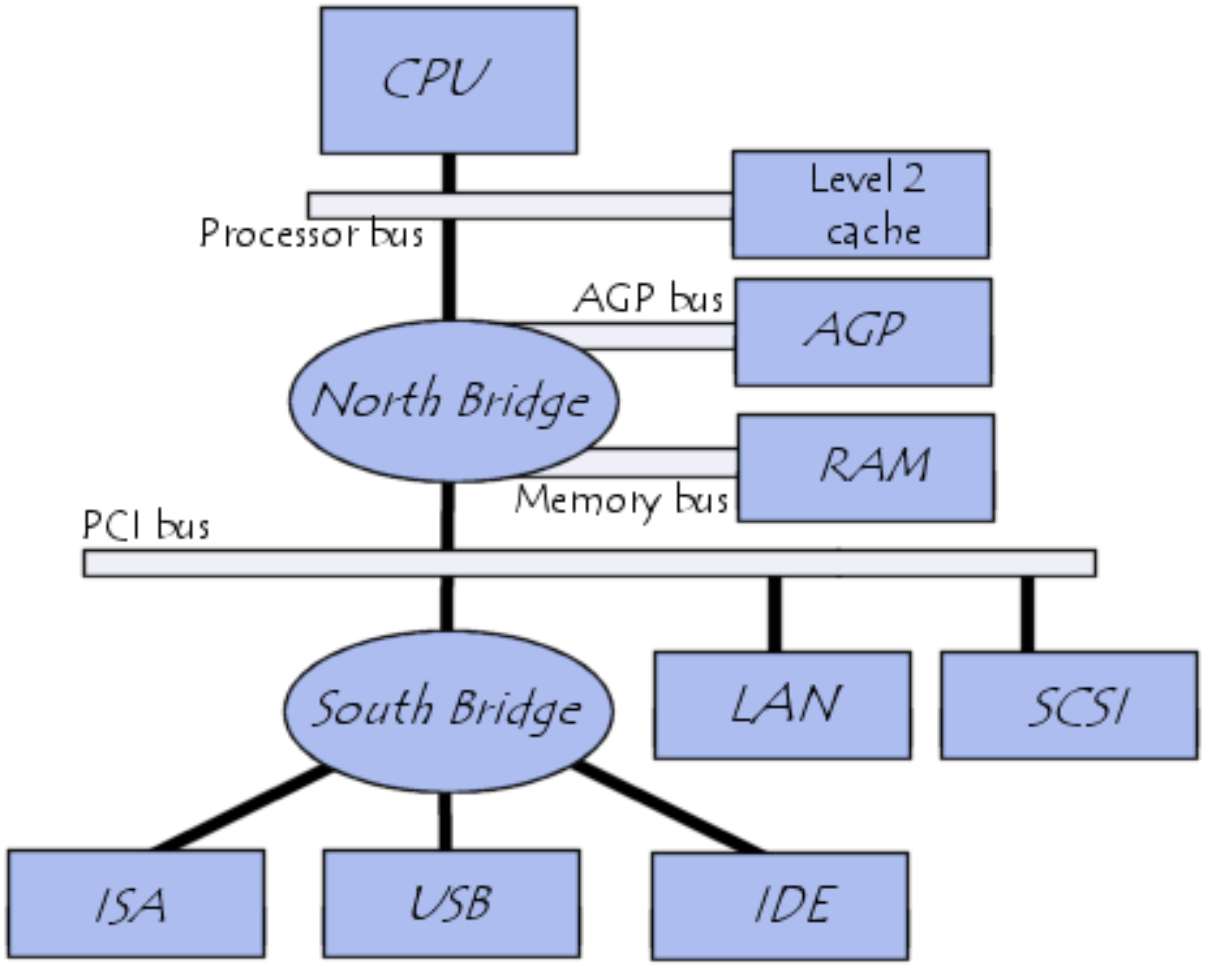
Quelques définitions

- Bus = ensemble de liaisons physiques pouvant être exploitées en commun par plusieurs éléments matériels afin de communiquer.
- Bande passante = débit d'informations ; quantité d'information échangée par unité de temps
- Latence = temps de réponse du bus à une requête de transfert, temps minimum d'établissement de la connexion : indépendant de la quantité de données à transporter

Ordres de grandeur

Protocole	Usage	Débit théorique maximal
SATA 3.0	Disque dur	750 Mo/s
USB 2.0	Périphériques externes	60 Mo/s
DDR3	RAM	17000 Mo/s
ADSL	Réseau domestique	8 Mb/s

Communications internes



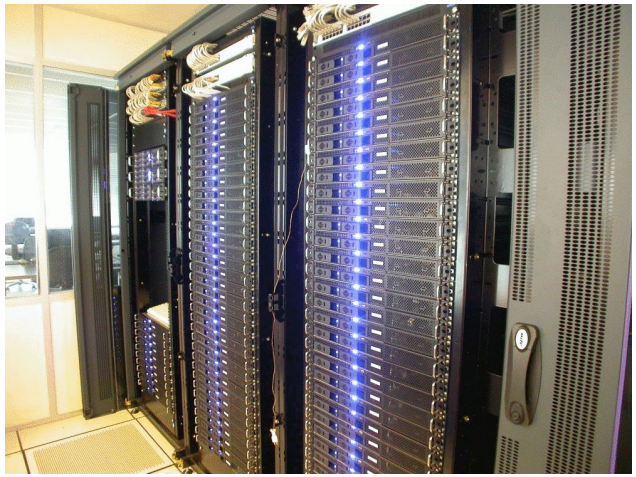
Réseaux

En Calcul Haute Performance (High Performance Computing, HPC), l'utilisation d'une seule machine ne suffit plus :

- durée de calcul trop longue
- domaine de calcul trop grand

Cluster de calcul

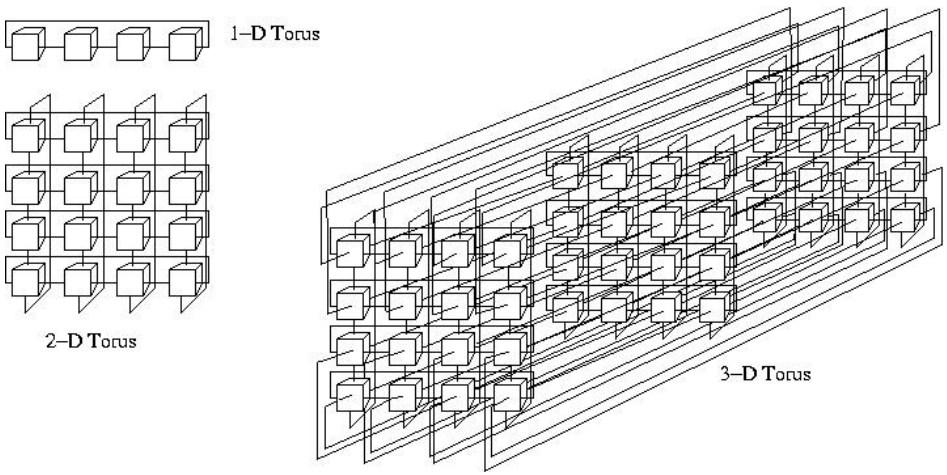
Regrouper plusieurs ordinateurs indépendants appelés nœuds (node en anglais), afin de dépasser les limitations d'une seule machine.

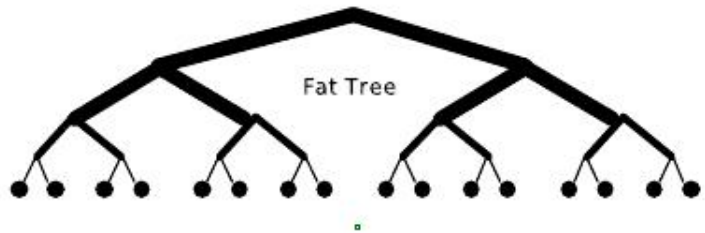


Le réseau est un élément essentiel des calculateurs

Caractéristiques des réseaux

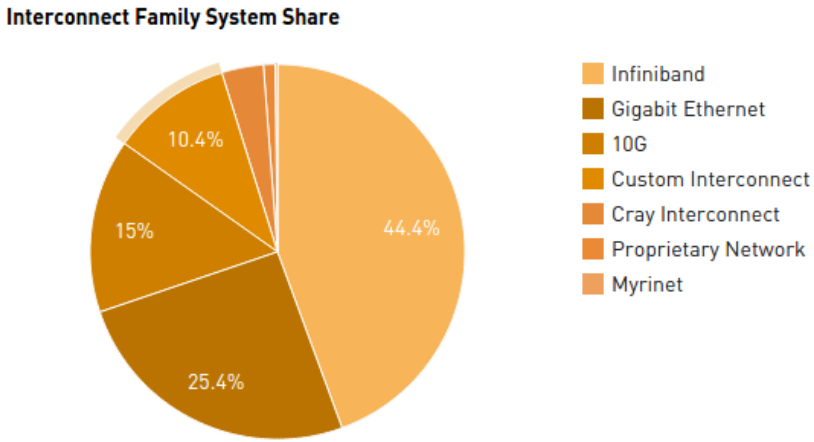
- La bande passante (bandwidth) est le débit maximal.
  - La bande passante effective est généralement inférieure à la bande passante physique (souvent à cause de l'overhead du protocole)
  - Ce qui intéresse l'utilisateur est la bande passante MPI
  - Les performances bas niveaux sont intéressantes pour le stockage
- La latence : temps d'initialisation de la communication.
- La distance maximum entre deux nœuds.
- La topologie : tore, fat-tree ...





Technologies réseaux

Technologie	Latence	Bande passante
1 Gb Ethernet	-	1 Gb/sec
10 Gb Ethernet	$\sim$10 μs	10 Gb/sec
Infiniband	< 2 μs	10, 20 & 40 Gb/sec
Myrinet	2.5 - 5.5 μs	10 Gb/sec



Répartition des familles d'interconnexion, top 500, juin 2014

Sommaire

1. Architecture générale
   - Processeur
   - Mémoire
   - Communications internes et réseaux
   - Stockage
2. Tendances
   - Evolutions technologiques et problématiques associées
   - GPU
   - Many-cœurs
3. Moyens de calcul disponibles
   - Micro éclairage sur le HPC mondial
   - Pyramide des moyens de calcul : paysages européen et français
   - Les grilles de calcul
   - Comment accéder à ces ressources ?
4. Concepts du parallélisme
   - Introduction
   - Classification et terminologie
   - Efficacité du parallélisme
   - Modèles de programmation parallèle
5. Conclusions

Goulot d'étranglement

Le débit global est dépendant de toute la chaîne d'I/Os (baie, serveur, interconnexion ...).

Bottleneck

Loi d'Amdahl sur l'équilibre des machines de calcul : une instruction par seconde requiert un octet de mémoire et un bit/seconde de capacité d'entrée-sortie.

- La capacité de stockage augmente beaucoup plus vite que la vitesse d'accès
- Le débit vers les disques augmente moins vite que la puissance de calcul
- La quantité de données générées augmente avec la puissance de calcul
- L'approche classique un fichier/processus génère de plus en plus de fichiers
- Trop de fichiers = saturation des systèmes de fichiers (nombre maximum de fichiers et espace gaspillé à cause des tailles de bloc fixes)
- Le temps passé dans les I/O croît (surtout pour les applications massivement parallèles)

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Evolution des performances

Top 500 Bench Linpack	Puissance soutenue	Puissance crête	Nombre de processeurs ou de cœurs
Juin 1993	59.7 GFlop/s	131 GFlop/s	1024
Juin 2014	33 862 TFlop/s	54 902 TFlop/s	3 120 000

1997 : année du Teraflops (10^{12} Flops)

↓

2008 : année du Petaflops (10^{15} Flops)

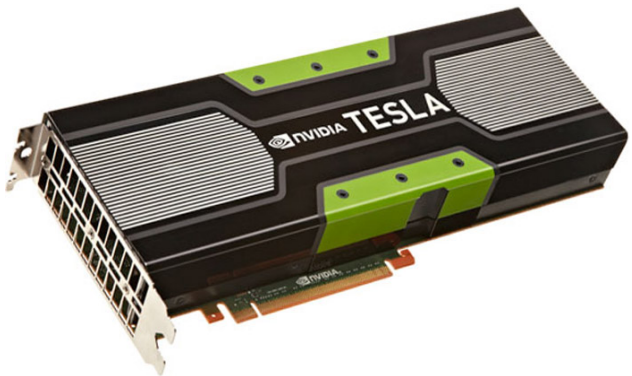
↓

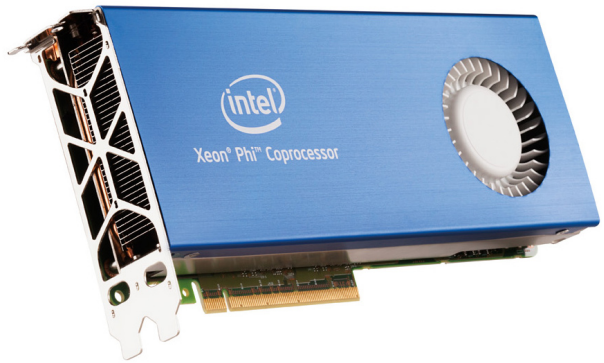
2022 : année de l'Exaflops (10^{18} Flops) ? ? ?

Architectures actuelles

Les machines de calcul actuelles sont constituées de :

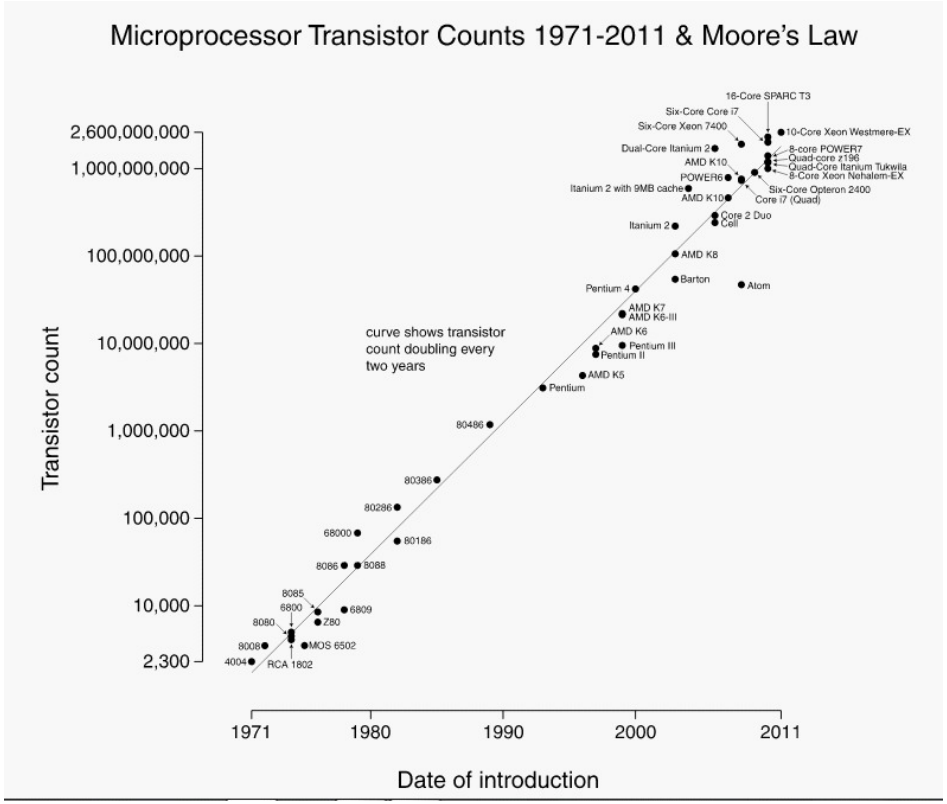
- Serveur multi-coeurs
- Hybride : contenant des accélérateurs comme des cartes graphiques (GPU) ou des cartes many-coeurs (Xéon Phi)
- Reliés entre eux pour constituer des clusters de calcul





Comment atteindre l'exascale ?

1. Améliorer la vitesse du processeur : augmenter la fréquence d'horloge.
2. Augmenter le nombre de transistors, donc augmenter le nombre de cœurs : diminuer la finesse de gravure.
3. Utiliser des processeurs spécialisés ou des cartes accélératrices.



Les principales limitations

Consommation électrique

- Augmente de façon exponentielle en fonction de la fréquence d'horloge :

$$P \propto f^3$$

  - P : puissance (Watt)
  - f : fréquence
- Mais pas que ! ! tous les éléments sont consommateurs : mémoires, carte mère, alimentation inefficace ...
- *CURIE (machine européenne hébergée en France) consomme environ 3.5 MW, autant que la commune voisine d'Arpajon (9000 habitants).*

Les principales limitations

Finesse de gravure

- Prochaine génération de processeurs Intel gravée à 14 nm, soit quelques centaines de couches d'atomes ... Feuille de route Intel : 4 nm en 2022. A quels coûts ?
- Plus de transistors donc plus de cœurs, et moins de dissipation thermique (plus c'est petit moins ça dissipe)
- Mais des défis technologiques pour les fondeurs et des technologies de plus en plus coûteuses !
- Jusqu'à quand ? ?

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

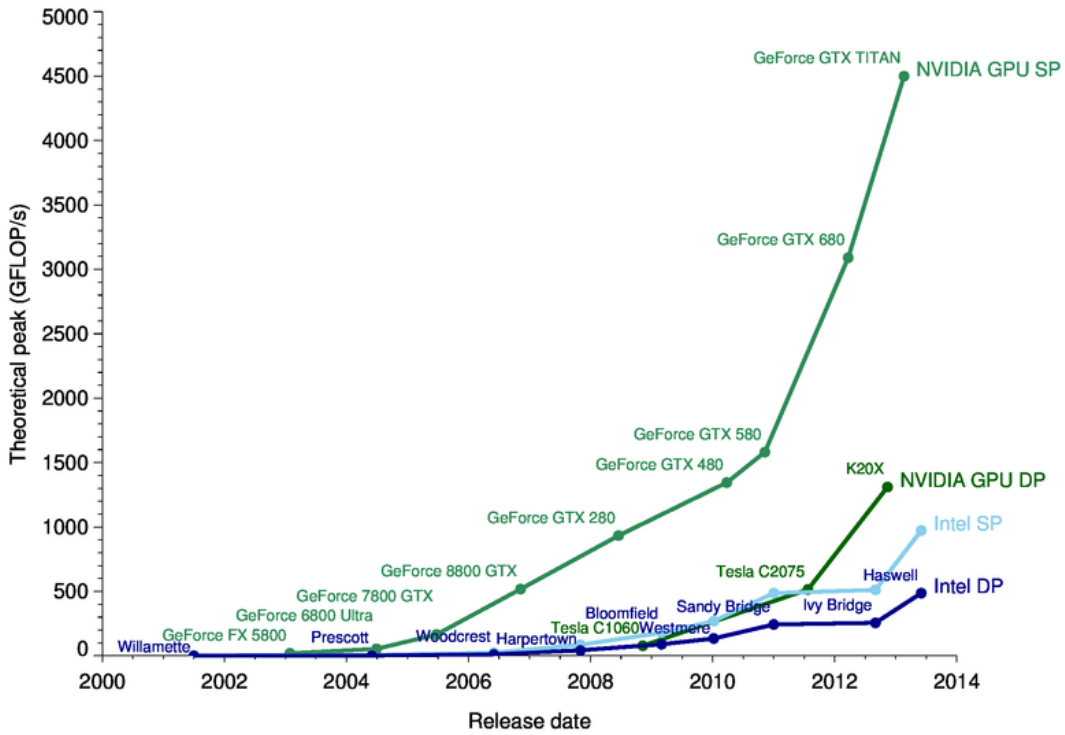
2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

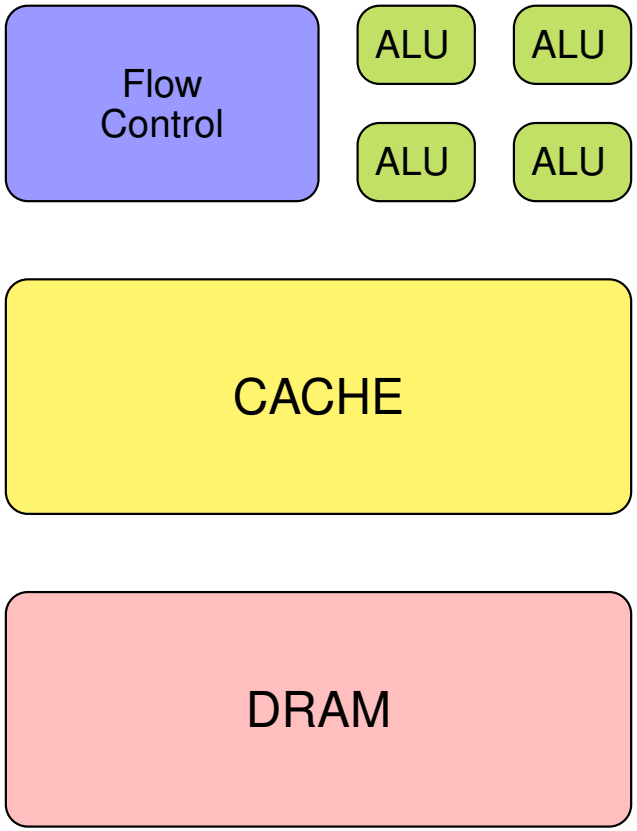
Pourquoi les GPU ?



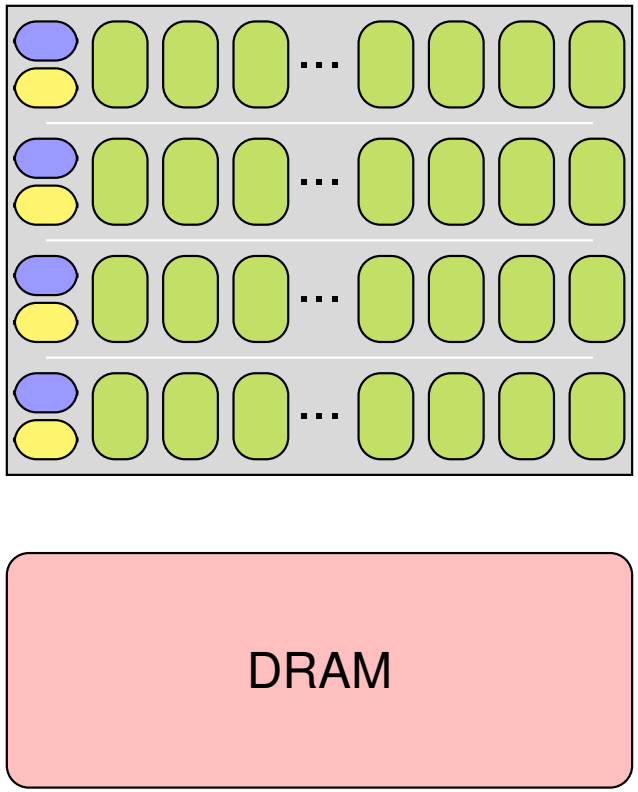
A performance théorique égale, les plateformes à base de GPU :

- occupent moins de place
- sont moins chères
- sont moins consommatrices d'électricité

Architecture comparée d'un CPU et d'un GPU



CPU

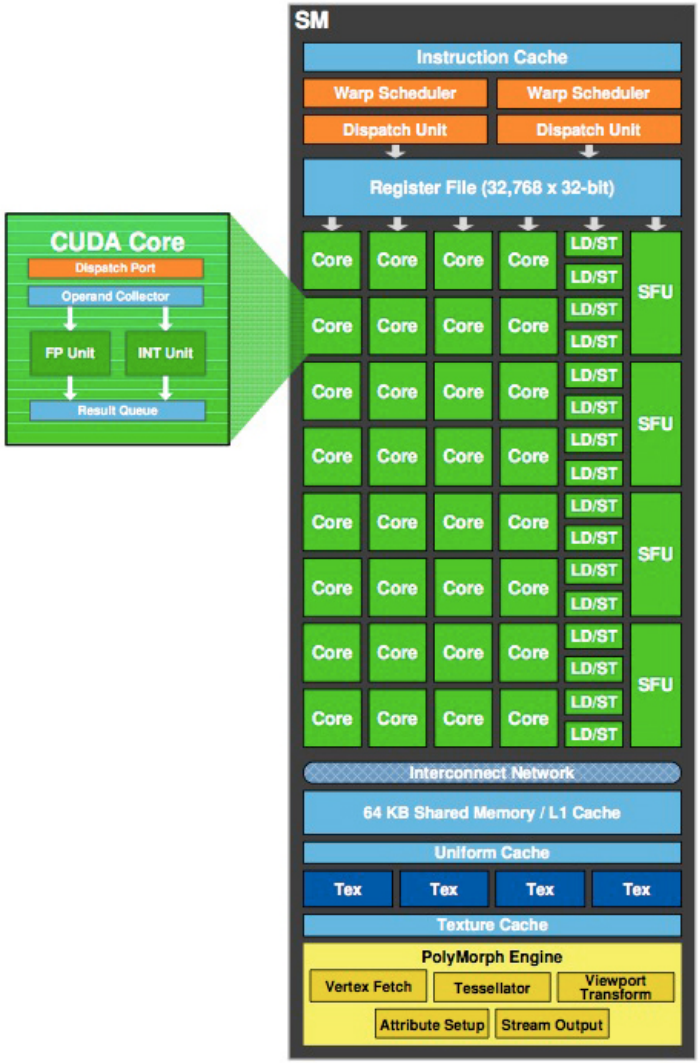


GPU

Caractéristiques d'un GPU

Deux composants principaux :

- **Mémoire globale** : similaire à la RAM du CPU, accessible à la fois par le GPU et le CPU.
  - Actuellement, jusqu'à 6 GB par GPU.
  - Bande passante jusqu'à 250 GB/s
- **Streaming Multiprocessors (SMs)** : en charge des calculs. Chaque SM possède ses propres :
  - unités de contrôle
  - registres
  - pipelines d'exécution
  - caches



Quelles applications pour le GPU ?

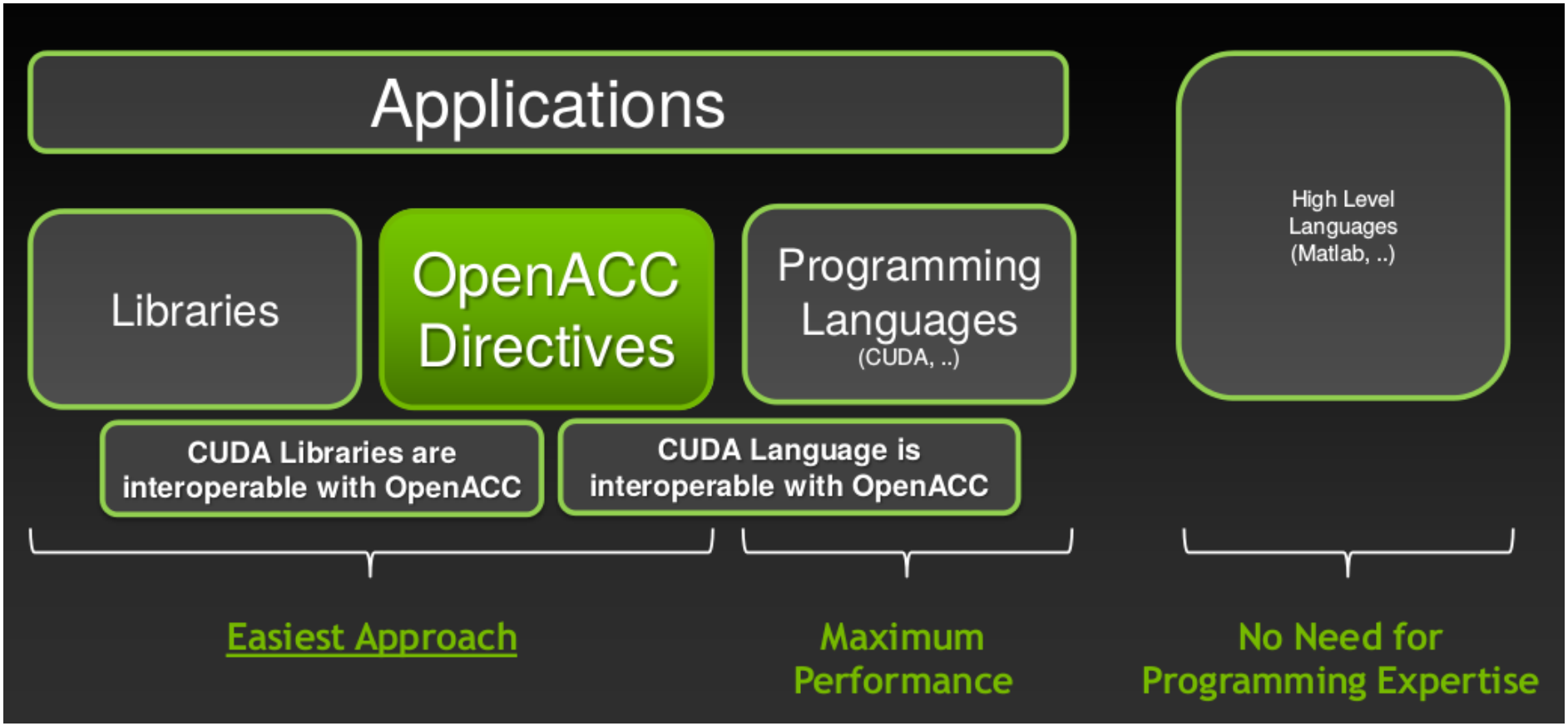
Exploitation d'un GPU

- Programmation de type SIMD, gestion de milliers de thread
- Transfert CPU-GPU encore très couteux
- De grosses évolutions en ce qui concerne la programmation

$\rightarrow$ Reste relativement complexe à exploiter pour en tirer des performances raisonnables, toutes les applications ne s'y prêtent pas :

- Problématique avec beaucoup d'algèbre linéaire (utilisation des bibliothèques CUDA)
- Applications avec de nombreuses tâches indépendantes
- Il faut pouvoir exploiter des milliers/millions de threads

Accélérer les applications grâce au GPU



Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

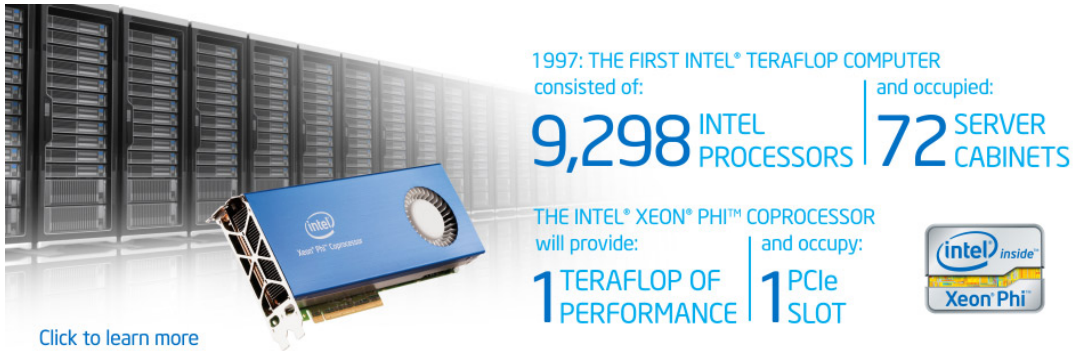
3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Evolution actuelle

- Apparition sur le marché de cartes Many Integrated Core : les Xeon Phi©
- Intel Xeon Phi 5110P : 60 coeurs à 1053 MHz, 8 Go de mémoire



Spécificités

Une architecture x86 native - en fait à la fois un complément idéal à Xeon et un concurrent crédible aux accélérateurs GPU.

Programmer sur le Phi

- La carte exécute son propre OS, dispose de sa propre adresse IP et passe par le bus PCIe pour communiquer avec les autres éléments du système
- On peut donc se connecter dessus par ssh et travailler presque comme sur un serveur classique. Deux modes d'exploitation :
  - déporter les parties parallèles sur le Phi (offload) : utilisation de directives openMP
  - exécuter nativement tout le code sur le Phi : compilation sur l'host en indiquant le MIC comme architecture cible
- 60 cœurs, 240 threads
- Vecteurs de 512 bits

Pour exploiter efficacement un Phi

- Application massivement parallèle
- Boucles vectorisées (alignement des données)
- Localité des données optimisée

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Micro éclairage sur le HPC mondial

Top 500

- Projet de classification des 500 premiers supercalculateurs connus au monde.
- Depuis juin 1993, liste mise à jour tous les six mois
- LinPack : créé par Jack Dongarra, il mesure le temps mis par un ordinateur pour résoudre un système de n équations à n inconnues dense, la solution étant obtenue par une utilisation partielle du pivot de Gauss, par $2/3.n^3 + n^2$ opérations à virgule flottantes. La performance est ensuite calculée en divisant le nombre d'opérations par le temps mis, donc en FLOPS.

Top 500, juin 2014

Petit aperçu du classement

Pays	Système	Nombre de cœurs	RPeak (TFlops/s)
Chine	Tianhe-2 Intel Xeon E5-2692, Intel Xeon Phi 31S1P	3,120,000	54,902.4
USA	Titan - Opteron 6274 1, NVIDIA K20x	560,640	27,112.5
USA	Sequoia - BlueGene/Q	1,572,864	20,132.7
Japon	K computer - SPARC64	705,024	11,280.4
USA	Mira - BlueGene/Q	786,432	10,066.3

Quelques statistiques

- Cartes accélératrices : 12.8% avec, 8.6% Nvidia, 3.2% Xéon Phi.
- Pays : 46.4% USA, 15.2% Chine, 6% Japon, 6% UK, 5.4% France, 4.4% Allemagne

IBM BlueGene

- Augmenter le nombre de coeurs en réduisant leur fréquence.
- Diminution de l'écart entre les débits et les latences mémoires, diminution de l'écart entre les débits et les latences réseaux $\longrightarrow$ machine plus équilibrée, une puissance de calcul élevée à consommation électrique équivalente
- Mais chaque coeur est lent, un très grand nombre de coeurs, peu de mémoire par coeur

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

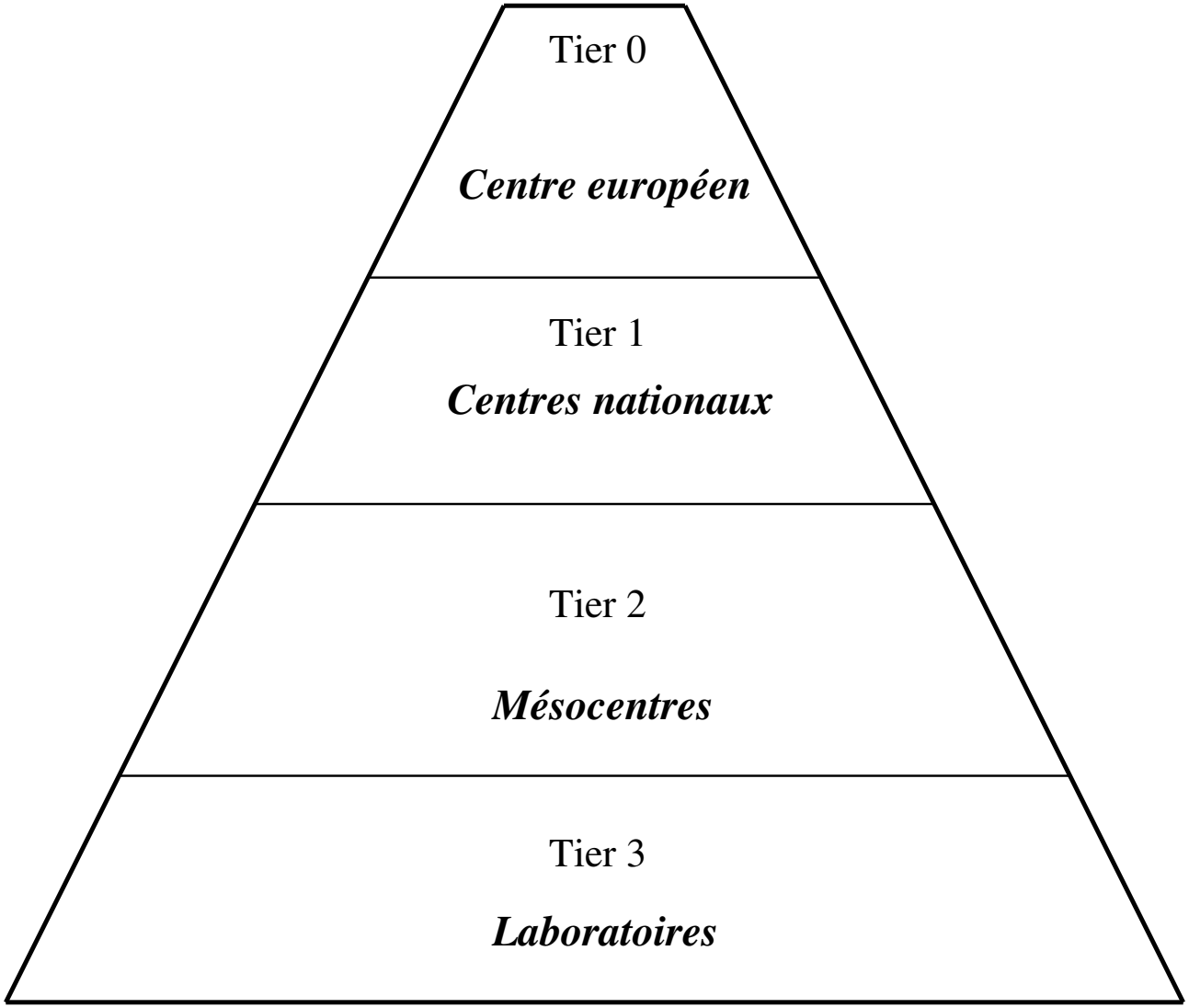
2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

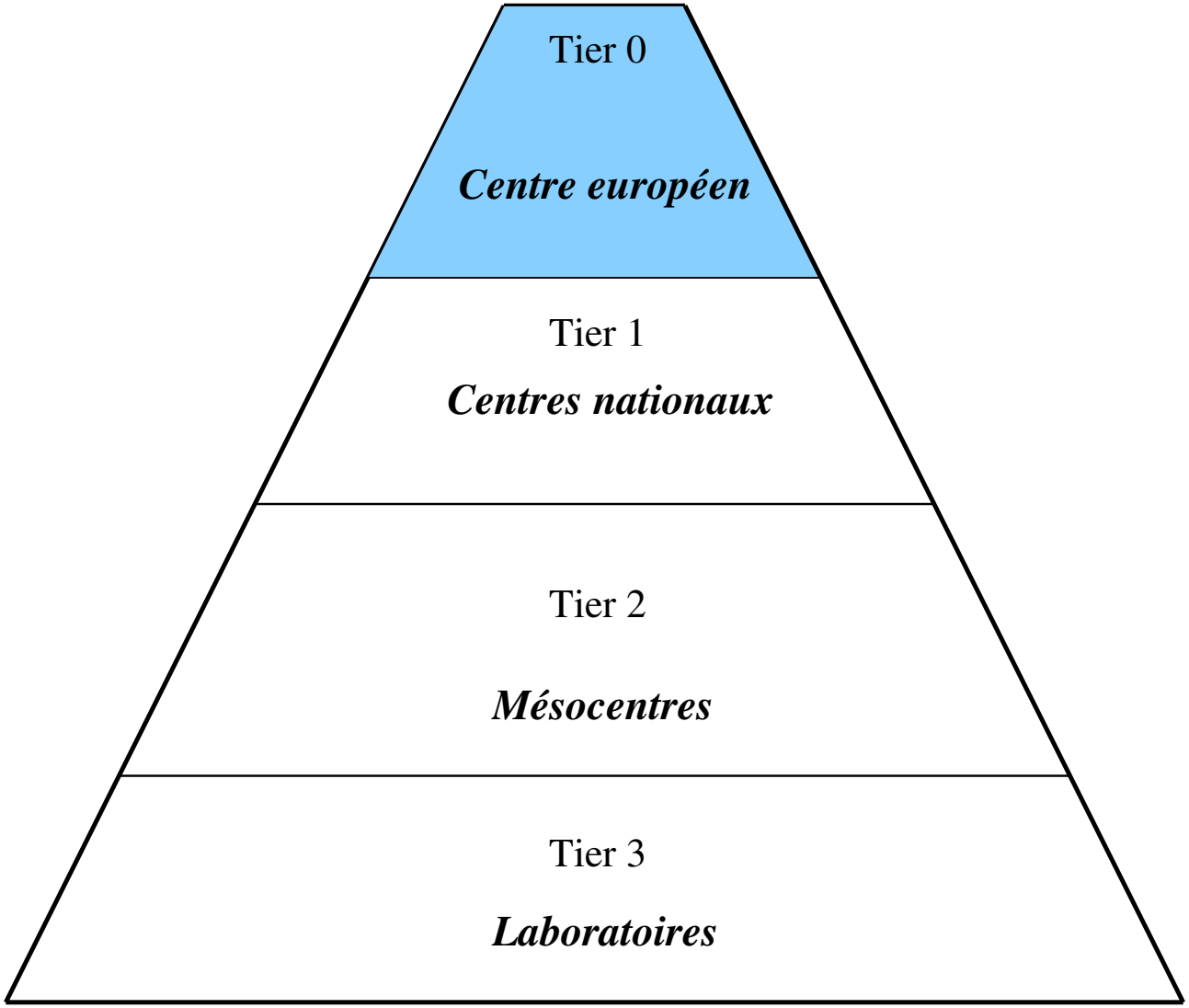
4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Paysages européen et français



Paysages européen et français



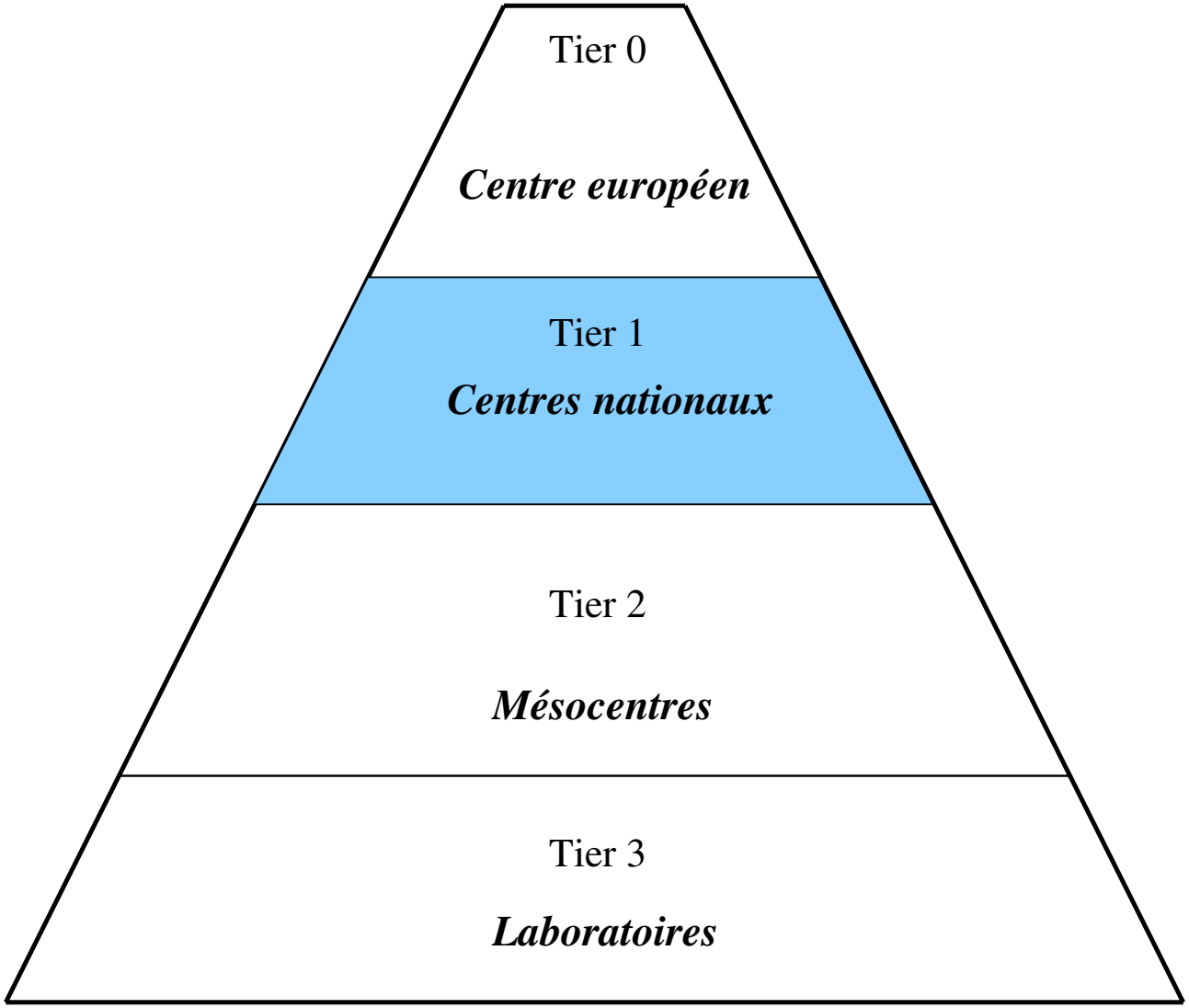
PRACE et l'échelle européenne

PRACE : Partnership for Advanced Computing in Europe

- Regroupement de 25 pays (dont Allemagne, Espagne, Grande-Bretagne, France ...)
- Accès à 6 supercalculateurs aux architectures complémentaires, d'une puissance crête globale de près de 15 petaflops, localisés en Allemagne, en Espagne, en France (CURIE) et en Italie.

http://www.prace-ri.eu/

Paysages européen et français



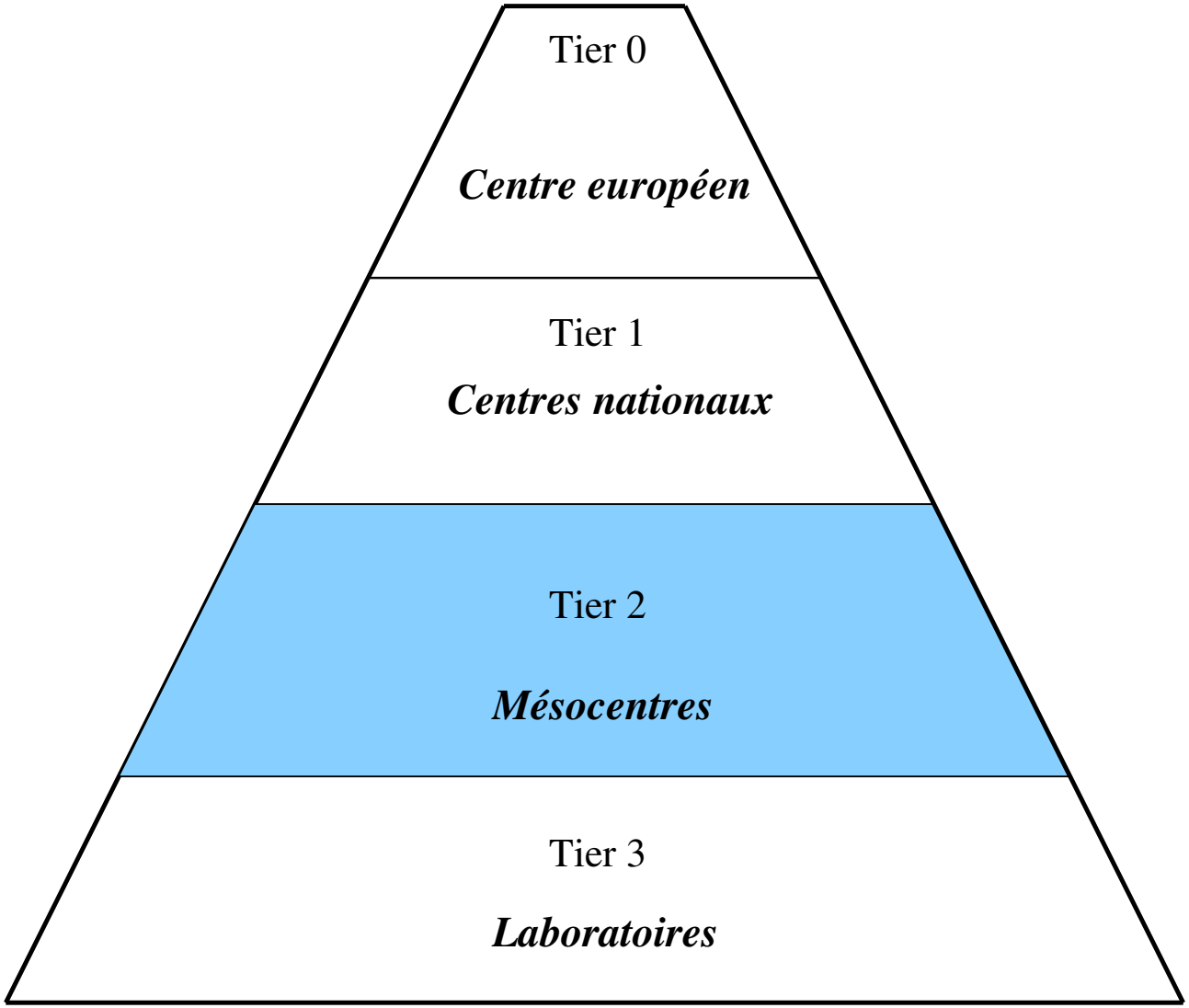
GENCI et l'échelle nationale

GENCI, Grand Equipement National de Calcul Intensif

Société de droit civil détenue à 49% par l'Etat représenté par le Ministère de la Recherche et l'Enseignement Supérieur, 20% par le CEA, 20% par le CNRS , 10% par les Universités et 1% par l'INRIA.

- TGCC (CEA)
- CINES
- IDRIS (CNRS)

http://www.genci.fr


Tier 0
Centre européen
Tier 1
Centres nationaux
Tier 2
Mésocentres
Tier 3
Laboratoires

Les mésocentres et l'échelle régionale

Définition d'un mésocentre

- Un ensemble de moyens humains, de ressources matérielles et logicielles à destination d'une ou plusieurs communautés scientifiques, issus de plusieurs entités (EPST, Universités, Industriels) en général d'une même région, doté de sources de financement propres, destiné à fournir un environnement scientifique et technique propice au calcul haute performance.

FLMSN

- Fédération Lyonnaise de Modélisation et des Sciences Numériques
- Regroupe le P2CHPD (Lyon 1), le PSMN (ENS), le PMS2I (ECL), le Centre Blaise Pascal et l'IXXI.

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Grille de calcul

Grille

- Agrégat de réseaux autonomes (l'autonomie s'exprime au niveau de chaque réseau) et hétérogènes(l'hétérogénéité se manifeste aussi bien au sein d'un réseau qu'entre deux réseaux) d'unités informatiques ou apparentées.
- Une grille de calcul intègre un intergiciel (terme anglais : middleware) : ensemble des couches réseaux et services logiciels qui permettent le dialogue entre les différents composants d'une application répartie. L'intergiciel masque la complexité des échanges inter-applications.

Différences entre calcul HPC et calcul sur grille :

- Grille : parallélisme non couplé, adapté aux traitements de masse de données, ou aux études paramétriques par exemple
- Supercalculateur : couplage massif entre les processeurs, adapté aux calculs massifs induisant des tâches non autonomes avec des communications, par exemple mécanique des fluides ...

Les grilles de production et de recherche

France Grille

- Infrastructure de grille de production française
- Contribue à l'infrastructure de grille européenne EGI
- Favorise les échanges entre grilles de production et grilles de recherche

http://www.france-grilles.fr

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Accès aux ressources de calcul

PRACE

- Appels d'offre réguliers, ouvert aux chercheurs européens
- `http://www.prace-ri.eu/call-announcements/`

GENCI

- DARI : Demande d'Attribution de Ressources Informatiques
- Regroupe les 3 centres nationaux
- Ouvert aux chercheurs français
- `https://www.edari.fr/`

France-Grille

- Nécessité de se rattacher à une VO (Virtual Oganization)
- `http://www.france-grilles.fr/Pour-les-chercheurs-ou-ingenieurs`
- Il existe une VO « générique » France-Grille à laquelle on peut se rattacher si notre communauté n'est pas utilisatrice traditionnelle des grilles.

Accès aux ressources de calcul

Mésocentres

- Politique différente selon les sites
- A Lyon, contacter directement les ingénieurs ou via les formulaires web :
  - Lyon 1 : `https://p2chpd-cluster.univ-lyon1.fr/P2CHPD/doku.php?id=p2chpd:support:demande_de_ressources`
  - ENS Lyon : `http://www.ens-lyon.fr/PSMN/doku.php?id=contact:forms:inscription`
  - ECL : `http://pmcs2i.ec-lyon.fr/tiki-index.php?page=DemandeRessources`

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

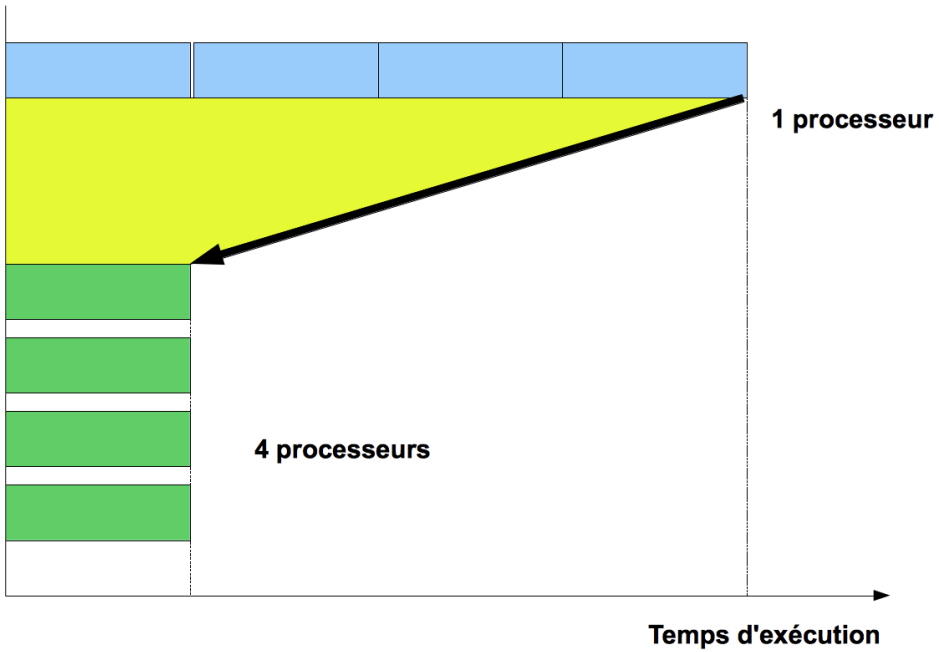
3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Introduction

- Exécutions séquentielles des programmes : le processeur n'exécute les instructions que l'une après l'autre, en passant éventuellement d'un processus à un autre via des interruptions en simulant le multitâche.
- Parallélisation : Ensemble de techniques logicielles et matérielles permettant l'exécution simultanée de séquences d'instructions indépendantes, sur des processeurs différents.



Pourquoi paralléliser ?

- Traitement de problèmes scientifiques constituant de grands challenges, tels que : météo et climat, biologie (génome), géophysique (activité sismique), réactions chimique et réactions nucléaires, ...
- Mais aujourd'hui également pour des applications commerciales : Bases de données parallèles, exploration pétrolière, moteurs de recherche web, réalité virtuelle, ...

✓ Traiter des problèmes plus grands et/ou plus complexes.
✓ Mais aussi exploiter les architectures actuelles !

La multiplication du nombre d'unités de calcul ne divise pas spontanément le temps d'exécution des programmes !

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Classification de Flynn

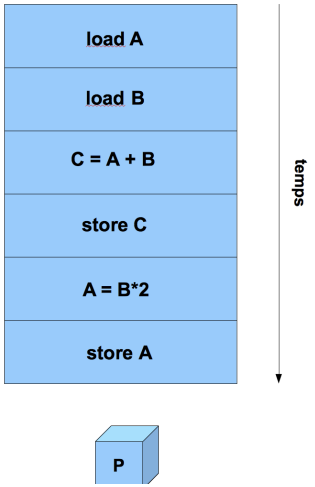
Les programmes et les architectures sont classés selon le type d'organisation du flux de données (« data ») et du flux d'instructions.

- 2 états possibles pour chacune : « single » ou « multiple »

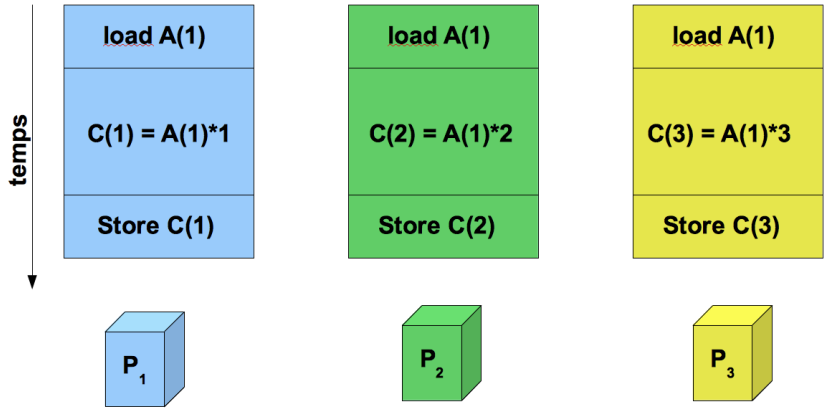
SISD Single Instruction, Single Data $\Longrightarrow$ machines séquentielles	**SIMD** Single Instruction, Multiple Data $\Longrightarrow$ processeurs vectoriels, GPU
MISD Multiple Instruction, Single Data $\Longrightarrow$ rare	**MIMD** Multiple Instruction, Multiple Data $\Longrightarrow$ multiproc, multicores

SISD, MISD

SISD architecture séquentielle avec un seul flot d'instructions, un seul flot de données, exécution déterministe.

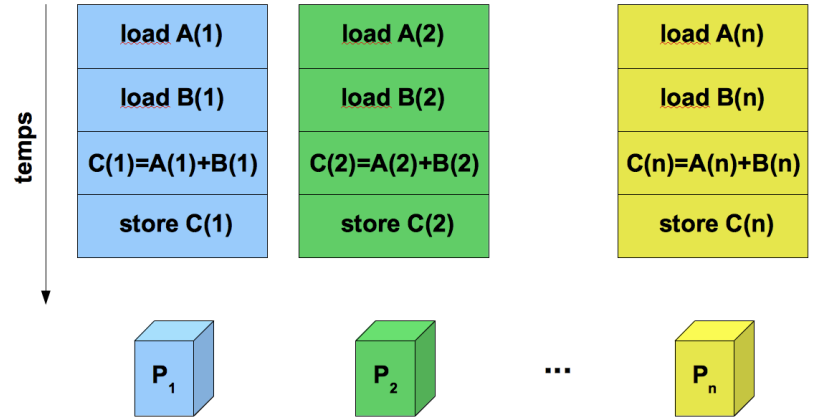


MISD un seul flot de données alimente plusieurs unités de traitement, chaque unité de traitement opère indépendamment des autres, sur des flots d'instructions indépendants.
Peu implémenté.
Ex d'utilisation : exécution en // de plusieurs algos de cryptographie pour le décodage d'1 même message.

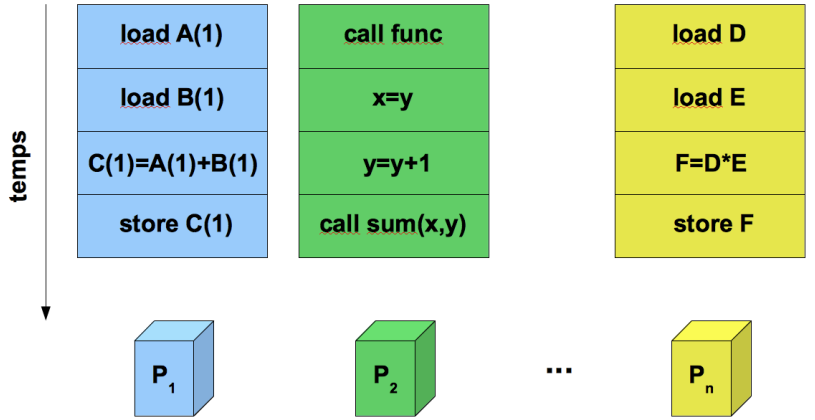


SIMD, MIMD

SIMD architecture // , toutes les unités de traitement exécutent la même instruction à un cycle d'horloge donnée, chaque unité peut opérer sur des données différentes, exécution déterministe.



MIMD architecture la plus courante. Chaque unité de traitement peut gérer un flot d'instructions différent. Chaque unité peut opérer sur un flot de données différent. L'exécution peut être synchrone ou asynchrone, déterministe ou non-déterministe.



Terminologie

- Tâche : une portion de travail à exécuter sur un ordinateur, du type un ensemble d'instructions d'un programme qui est exécuté sur un proc.
- Tâche parallèle : une tâche qui peut s'exécuter sur plusieurs processeurs, sans risque sur la validité des résultats.
- Exécution séquentielle : exécution d'un programme séquentiel, une étape à la fois.
- Exécution parallèle : exécution d'un programme par plusieurs tâches, chaque tâche pouvant exécuter la même instruction ou une instruction différente, à un instant donné.
- Mémoire partagée : D'un point de vue hard, réfère à une machine dont tous les proc. ont un accès direct à une mémoire commune (généralement via un bus). D'un point de vue modèle de programmation : toutes les tâches ont la même image mémoire et peuvent directement adresser et accéder au même emplacement mémoire logique, peu importe où il se trouve en mémoire physique.

Terminologie

- Mémoire distribuée : d'un point de vue physique, basée sur un accès mémoire réseau pour une mémoire physique non commune.
  D'un point de vue modèle de programmation, les tâches ne peuvent voir que la mémoire de la machine locale et doivent effectuer des communications pour accéder à la mémoire dune machine distante, sur laquelle d'autres tâches s'exécutent.
- Communications : les tâches parallèles échangent des données, par différents moyens physiques : via un bus à mémoire partagée, via un réseau. Quelque soit la méthode employée, on parle de « communication ».
- Synchronisation : la coordination des tâches en temps réel est souvent associée aux communications, elle est souvent implémentée en introduisant un point de synchronisation au-delà duquel la tâche ne peut continuer tant que une ou plusieurs autres tâches ne l'ont pas atteint.

Terminologie

- Granularité : mesure qualitative du rapport calcul / communications
  - Grain grossier (coarse) : relativement bcp de calculs entre différentes communications.
  - Grain fin (fine) : relativement peu de calcul entre différentes communications.
- speedup : mesure de l'amélioration des performances due au parallélisme. (wall clock time of serial execution) / (wall clock time of parallel execution)
- Scalabilité : réfère à la capacité d'un système parallèle à fournir une augmentation de l'accélération proportionelle à l'augmentation du nombre de processeurs.

Sommaire

1 Architecture générale
- Processeur
- Mémoire
- Communications internes et réseaux
- Stockage

2 Tendances
- Evolutions technologiques et problématiques associées
- GPU
- Many-cœurs

3 Moyens de calcul disponibles
- Micro éclairage sur le HPC mondial
- Pyramide des moyens de calcul : paysages européen et français
- Les grilles de calcul
- Comment accéder à ces ressources ?

4 Concepts du parallélisme
- Introduction
- Classification et terminologie
- Efficacité du parallélisme
- Modèles de programmation parallèle

5 Conclusions

Parallélisation efficace

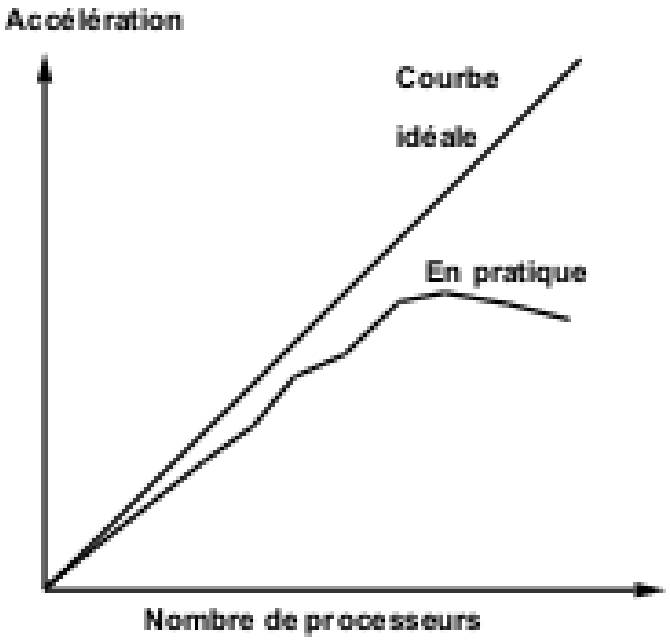
- Accélération et efficacité sont une mesure de la qualité de la parallélisation.
- Soit $T(p)$ le temps d'exécution sur p processeurs.
- L'accélération $A(p)$ et l'efficacité $E(p)$ sont définies comme étant :

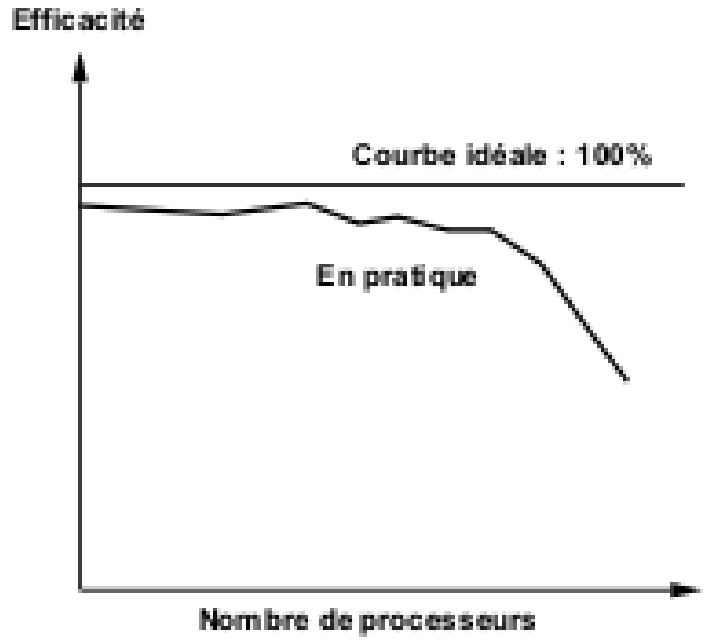
$$\begin{aligned} A(p) &= T(1)/T(p) \ \ (p = 1, 2, 3...) \\ E(p) &= A(p)/p \end{aligned}$$

- Pour une accélération parallèle parfaite, on obtient :

$$\begin{aligned} T(p) &= T(1)/p \\ A(p) &= T(1)/T(p) = T(1)/(T(1)/p) = p \\ E(p) &= A(p)/p = p/p = 100\% \end{aligned}$$

Accélération et efficacité





- Les programmes scalables demeurent efficaces pour un grand nombre de processeurs (scalables = passages à l'échelle).

Scalabilité, loi d'Amdahl

$\Longrightarrow$ Propriété d'une application à être exécutée efficacement sur un très grand nombre de processeurs.

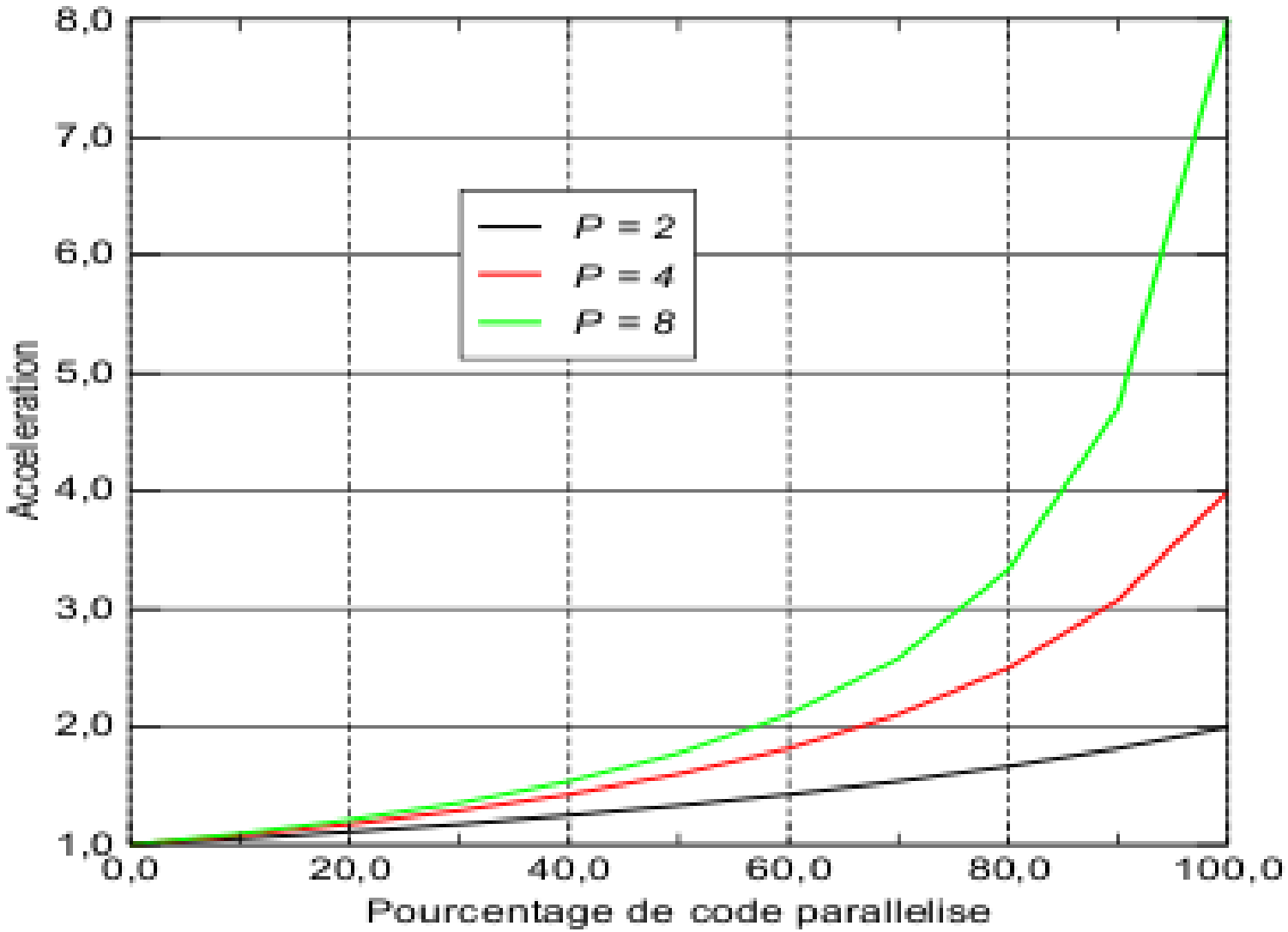
$\Longrightarrow$ Raison possibles d'une faible scalabilité :

- La machine parallèle employée : architecture inadaptée, charge ...
- Le programme parallélisé : analyser le comportement, améliorer les performances.
- un peu des deux ...

Loi d'Amdahl

- Plus une règle qu'un loi.
- Principe : **la partie non parallèle d'un programme limite les performances et fixe une borne supérieure à la scalabilité.**

Loi d'Amdahl

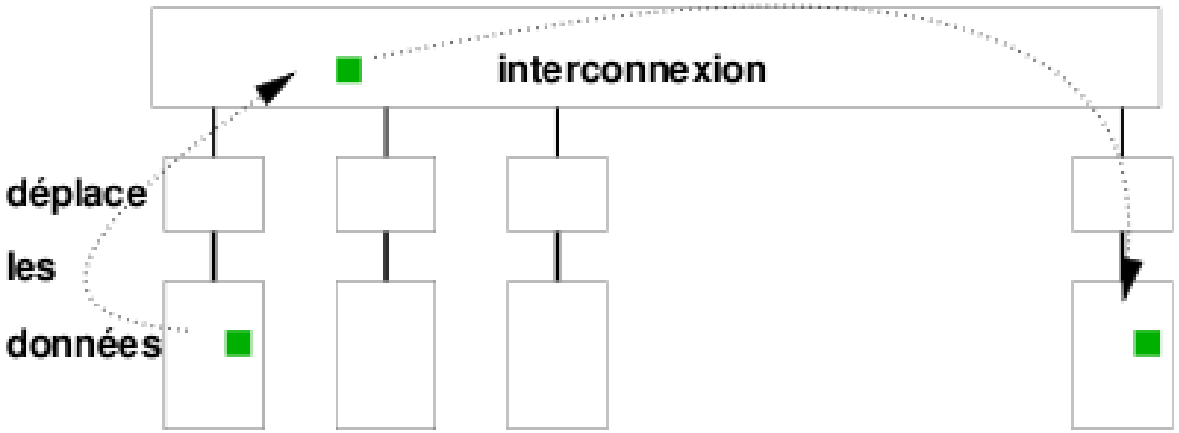

8,0
7,0
6,0
5,0
4,0
3,0
2,0
1,0
Accélération
P = 2
P = 4
P = 8
0,0
20,0
40,0
60,0
80,0
100,0
Pourcentage de code parallelise

Sommaire

1. Architecture générale
   - Processeur
   - Mémoire
   - Communications internes et réseaux
   - Stockage
2. Tendances
   - Evolutions technologiques et problématiques associées
   - GPU
   - Many-cœurs
3. Moyens de calcul disponibles
   - Micro éclairage sur le HPC mondial
   - Pyramide des moyens de calcul : paysages européen et français
   - Les grilles de calcul
   - Comment accéder à ces ressources ?
4. Concepts du parallélisme
   - Introduction
   - Classification et terminologie
   - Efficacité du parallélisme
   - Modèles de programmation parallèle
5. Conclusions

Mémoire Partagée

- Tous les processeurs accèdent à toute la mémoire globale avec un même espace d'adressage global.
- Chaque processeur travaille indépendamment des autres, mais les modifications effectuées par un processeur à un emplacement mémoire (en mémoire globale) sont visibles par tous les autres.
- Les processeurs ont leur propre mémoire locale (cache).



Mémoire Partagée

- UMA, Uniform Memory Access :
  - Des processeurs identiques, ayant tous le même temps d'accès à la mémoire. Plus couramment appelées SMP (Symmetric MultiProcessor).
  - Gestion de la cohérence des caches. Certains systèmes sont CC-NUMA : si un processeur modifie un emplacement mémoire partagé, la modification est visible par tous les autres processeurs.
- NUMA (Non Uniform Memory Access) :
  - Conçue pour pallier aux problèmes d'accès mémoire concurrents via un unique bus.
  - En général fabriquées à base de plusieurs blocs SMP interconnectés.
  - Des temps d'accès à la mémoire diffèrent suivant la zone accédée. Le temps d'accès via le lien d'interconnexion des blocs est plus lent.
  - Gestion de la cohérence des caches. CC-NUMA : Cache Coherent NUMA.

Mémoire Partagée

- Espace d'adresse globale $\longrightarrow$ facilite le travail du programmeur.
- Mémoire proche des CPUs $\longrightarrow$ le partage des données entre tâches est rapide.

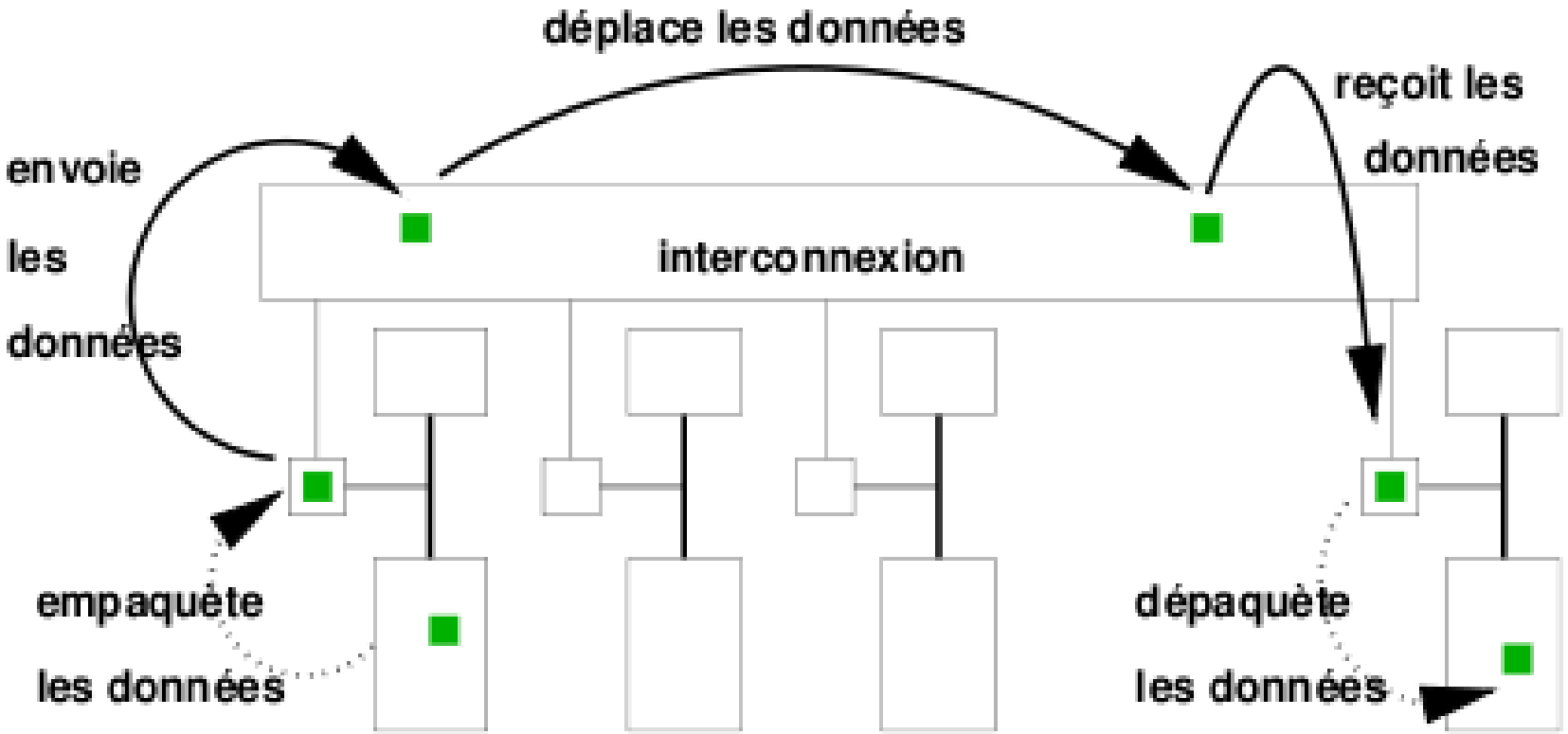
Mais :

- Manque de scalabilité : augmenter le nombre de CPUs accroît le trafic sur le chemin d'accès à la mémoire partagée.
- Le programmeur doit gérer la synchronisation pour un accès correct à la mémoire globale.
- Très coûteux de construire des architectures à mémoire partagée avec un grand nombre de CPUs.

Mémoire Distribuée

- Le point commun de ces architectures : elles possèdent un réseau d'interconnexion pour communiquer entre les mémoires des différents processeurs.
- Chaque processeur a sa propre mémoire locale, il n'y a pas de notion d'espace d'adressage global entre tous les procs.
- Les procs opèrent indépendamment les uns des autres, une modification en mémoire locale n'a pas d'effet sur la mémoire des autres procs.
- Si un processeur a besoin d'une donnée dans la mémoire d'un autre processeur, le programmeur doit définir explicitement la communication.
- Les réseaux d'interconnexion sont divers, avec des niveaux de performance très variables.

Mémoire Distribuée



Mémoire Distribuée

- On peut augmenter le nombre de processeurs et la mémoire proportionnellement.
- Accès rapide à la mémoire locale sur chaque proc, sans surcoût de gestion de cohérence de cache.
- Coût raisonnable, ex : PCs en réseau.

Mais :

- Le programmeur doit gérer toutes les communications entre processeurs.
- Peut être difficile de faire coïncider une structure de données basée sur une mémoire globale, à cette organisation physique de la mémoire.
- Temps d'accès mémoire non locaux élevés.

Architecture mémoire hybride

- La plupart des calculateurs actuels sont aujourd'hui un mixte de mémoire partagée et mémoire distribuée.
- La brique de base (nœud) est un multiprocesseur à mémoire partagée.
- Ces briques sont interconnectées par un réseau (type ethernet, Infiniband, ...).
- A noter qu'on trouve aussi de l'hybride entre CPU et cartes accélératrices.

Processus / Thread

Processus

- Un processus a son propre espace mémoire, sa propre pile qu'il ne partage pas avec les autres
- les processus sont plus lourds que les threads à créer : en général, 100 fois plus rapides à créer qu'un processus
- les processus sont réellement indépendants

Processus léger / Thread

- Les threads partagent le même espace mémoire
- les threads peuvent se partager des informations facilement
- les threads doivent eux-mêmes faire attention à ne pas se marcher sur les pieds : il n'y a pas de protection entre les threads d'un même processus
- Les threads ont peu d'informations propres

Sommaire

1. Architecture générale
   - Processeur
   - Mémoire
   - Communications internes et réseaux
   - Stockage
2. Tendances
   - Evolutions technologiques et problématiques associées
   - GPU
   - Many-cœurs
3. Moyens de calcul disponibles
   - Micro éclairage sur le HPC mondial
   - Pyramide des moyens de calcul : paysages européen et français
   - Les grilles de calcul
   - Comment accéder à ces ressources ?
4. Concepts du parallélisme
   - Introduction
   - Classification et terminologie
   - Efficacité du parallélisme
   - Modèles de programmation parallèle
5. Conclusions

Conclusions

Fréquence d'horloge, nombre de cœurs, nombre d'unités fonctionnelles, fréquence et capacité mémoire, caches, stockage, interconnexions internes et réseaux ...

Tous les éléments ont leur importance ; l'architecture doit être équilibrée pour que l'ensemble soit performant

Au niveau technologique

- Beaucoup de cœurs
- Des cœurs hybrides CPU / cartes accélératrices
- Une différence entre la vitesse de calcul et les transferts mémoire et I/O

Conclusions

Au niveau applicatif

- Nécessité d'une bonne connaissance des architectures, notamment au niveau de la mémoire.
- Plus de croissance des performances au niveau du cœur $\Longrightarrow$ parallélisation obligatoire pour un gain de performance.
- Nécessité d'élaborer des algorithmes et des programmes capables d'exploiter un grand nombre de processeurs.
- Nécessité d'exploiter le parallélisme aux différents niveaux du hardware.
- Programmation sur des architectures hybrides avec différents types de processeurs.