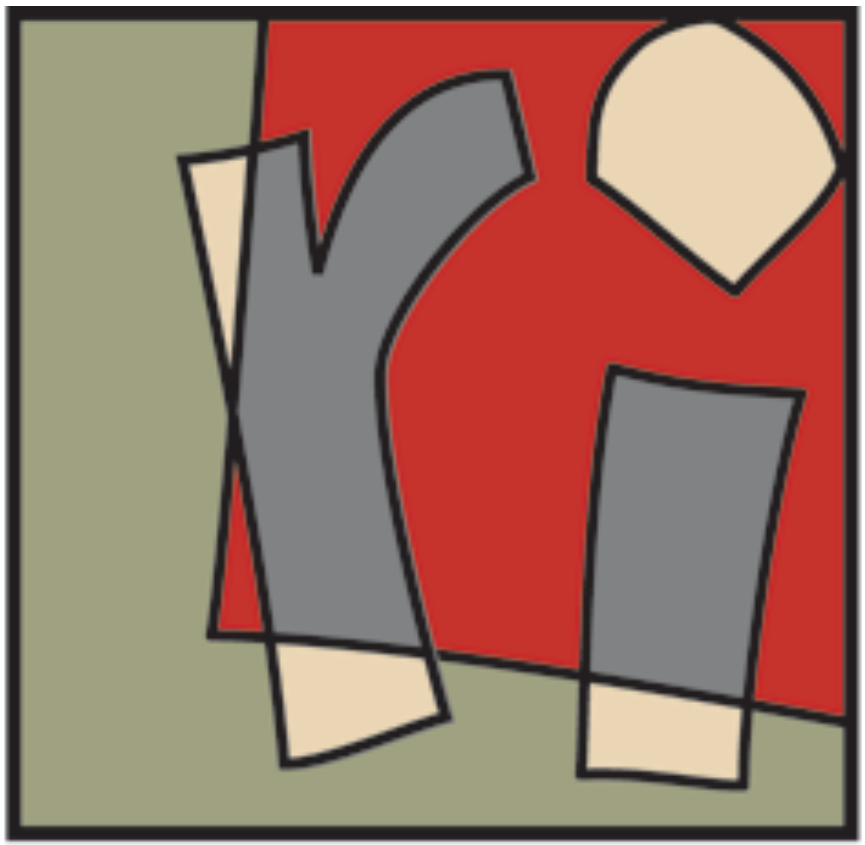
Introduction au Parallèlisme
Session 1 - Historique, Modèle et Architecture



Joel Falcou

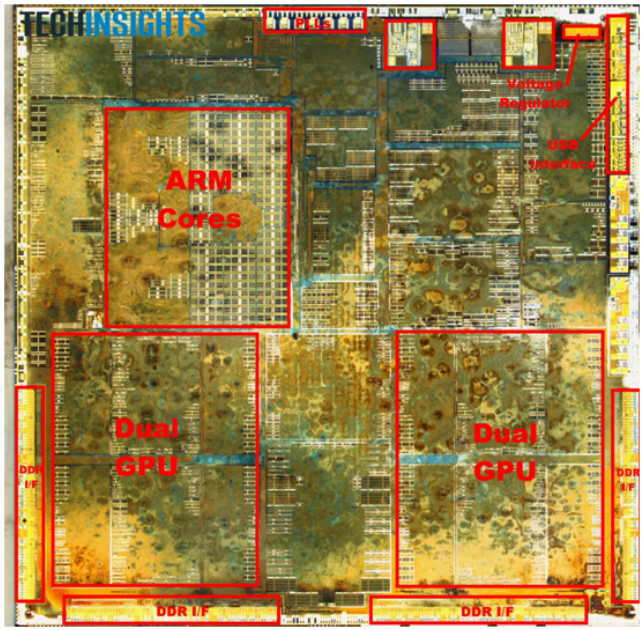
LRI - Universite Paris Sud XI - CNRS

February 4, 2014

Pourquoi un cours sur le parallélisme ?

Parce qu'il est partout

du super-calculateur à votre tablette.



Pourquoi un cours sur le parallélisme ?

Quels besoins ?

- Résoudre des problèmes plus rapidement
  - Traiter plus dans un temps fini.
  - Réduire le temps de traitement pour un travail donné.
- Obtenir des meilleurs résultats dans le même temps
  - Modèles plus précis
  - Modèles plus complexes
- Traiter des problèmes de plus grande taille
  - Données des réseaux sociaux
  - Google
  - LHC

Une introduction frileuse

- *"I think there is a world market for maybe five computers."*
  - Thomas Watson, chairman of IBM, 1943.

Une introduction frileuse

- *"I think there is a world market for maybe five computers."*
  - Thomas Watson, chairman of IBM, 1943.
- *"There is no reason for any individual to have a computer in their home."*
  - Ken Olson, president and founder of Digital Equipment Corporation, 1977.

Une introduction frileuse

- *"I think there is a world market for maybe five computers."*
  - Thomas Watson, chairman of IBM, 1943.
- *"There is no reason for any individual to have a computer in their home."*
  - Ken Olson, president and founder of Digital Equipment Corporation, 1977.
- *On several recent occasions, I have been asked whether parallel computing will soon be relegated to the trash heap reserved for promising technologies that never quite make it."*
  - Ken Kennedy, CRPC Directory, 1994.

Limite technologique

Accélérons donc les processeurs !

- Soit une machine séquentielle à 1 Tflop
  - Les données doivent aller de la mémoire au CPU (distance r)
  - Pour récupérer une donnée par cycle (10^{12} fois par seconde) à la vitesse de la lumière (c = 299 792 458 m/s = 3e8 m/s)
  - Donc r = c/10^{12} = .3mm
- Il faut mettre 1 Tera-octet de données dans 0.3 mm2
- Chaque mot occupe $\approx$ 3 Angstroms^2, soit la taille d'un petit atome
- Attention aussi à la chaleur dégagée par un tel processeur !

Limite technologique

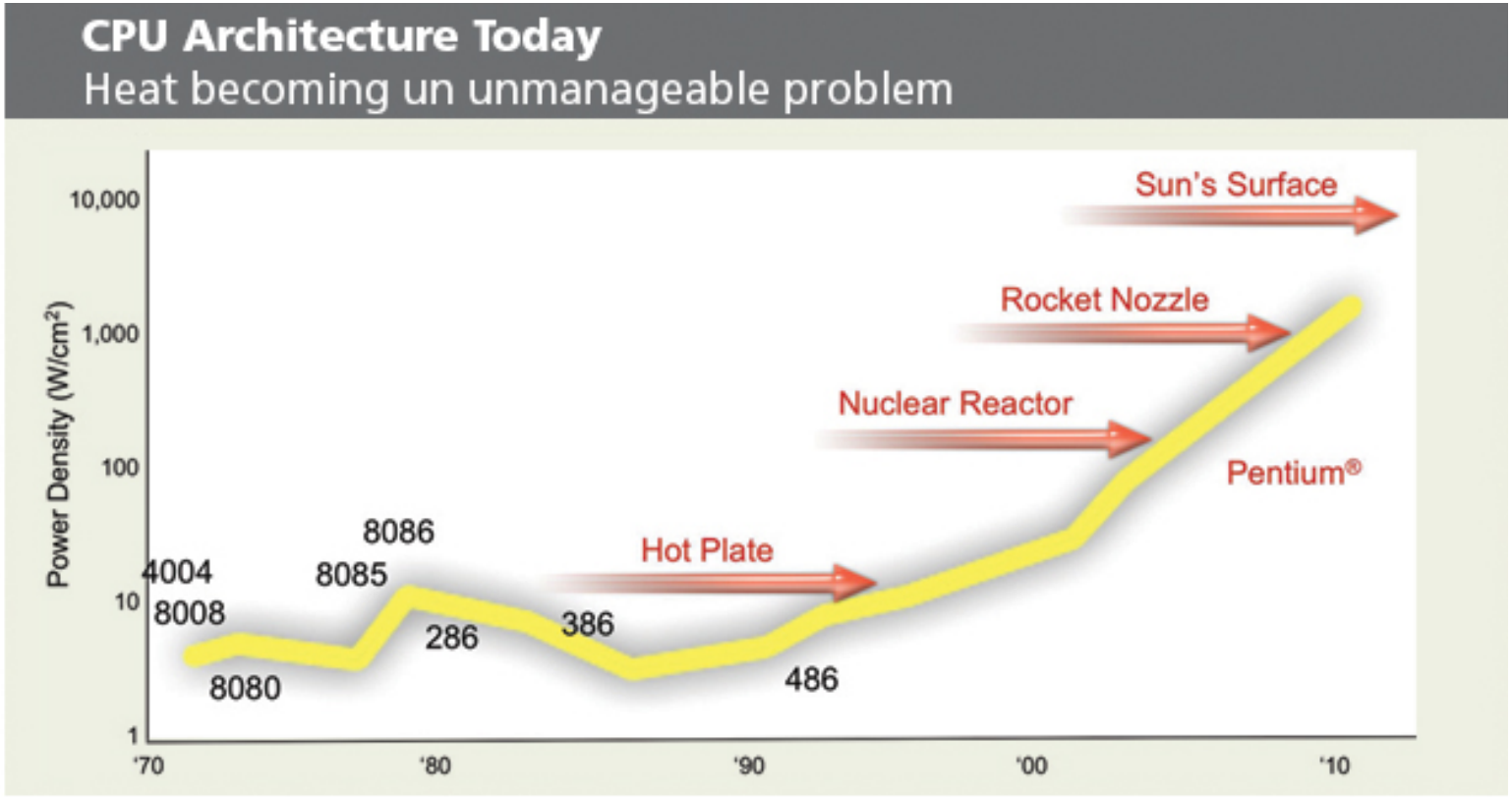
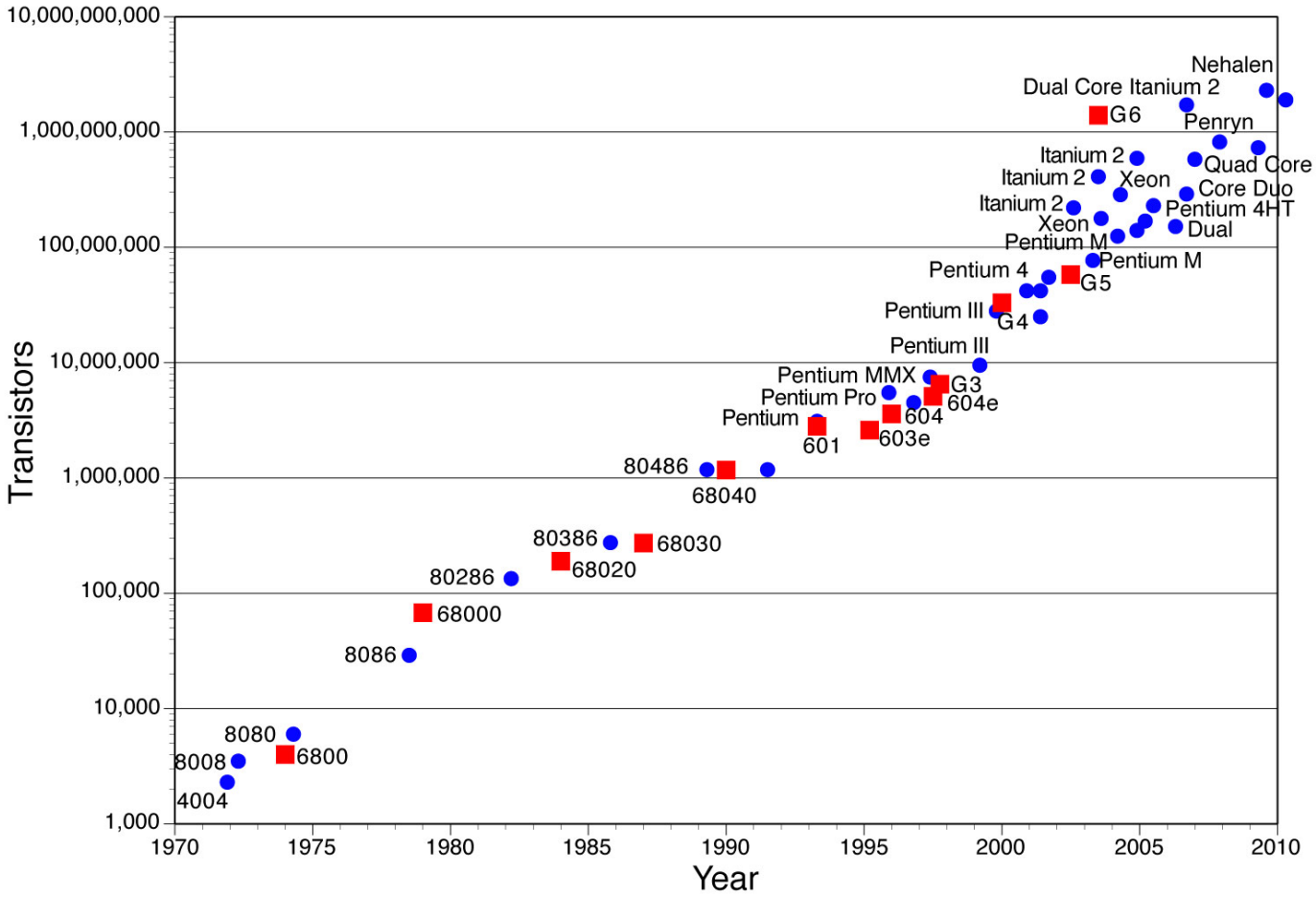
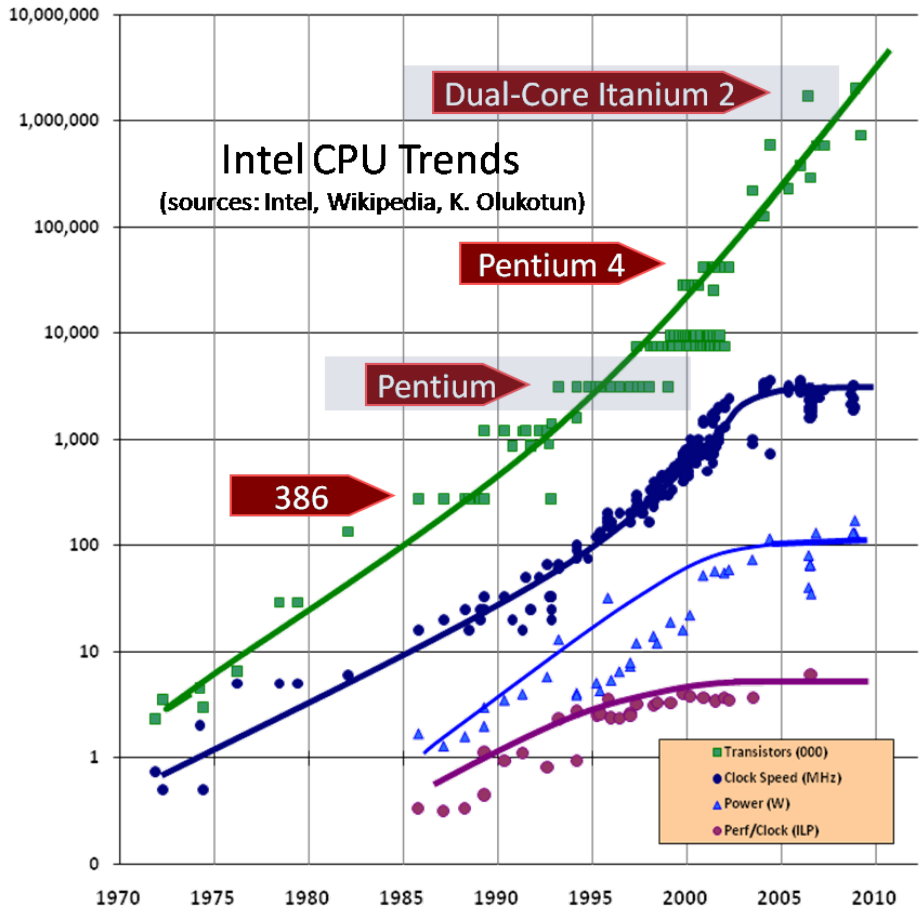


Figure 1. In CPU architecture today, heat is becoming an unmanageable problem. (Courtesy of Pat Gelsinger, Intel Developer Forum, Spring 2004)

Limite technologique



The Free Lunch is Over



Conséquences

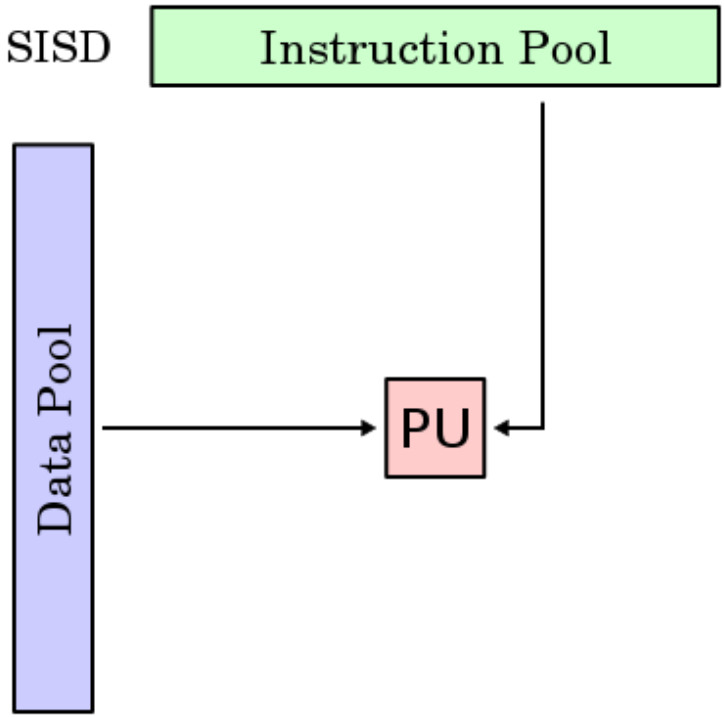
- Le seul moyen d'augmenter les performances est l'augmentation du nombre d'unité de traitement travaillant en parallèle
- L'argument du coût des transferts des données tient toujours !
- Difficulté de la programmation de ces machines

Plan

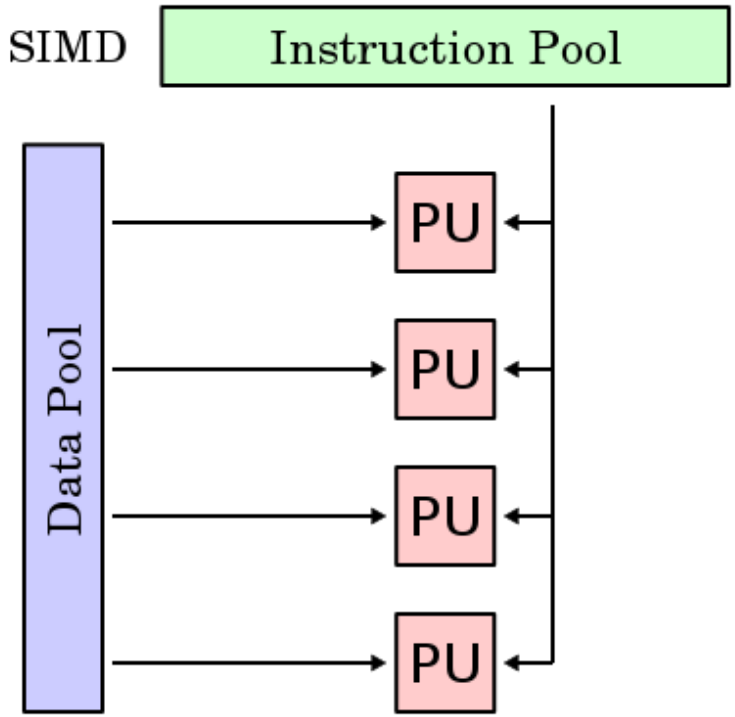
Classification des Architectures

	Single Data	Multiple Data
Single Instruction	*SISD* typical thread	*SIMD* vector processors GPUs SSE instructions
Multiple Instruction	*MISD* rare possibly set of filters	*MIMD* cluster of computers

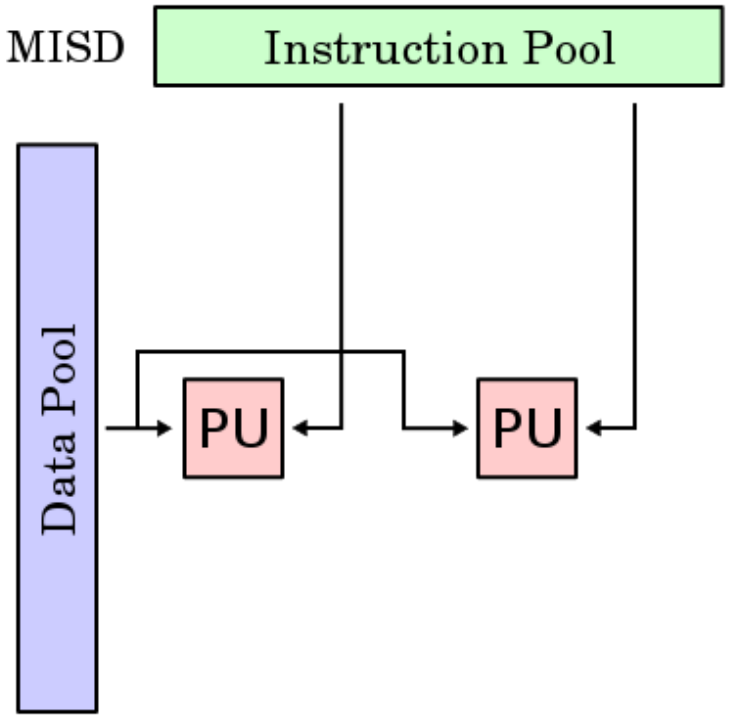
Classification des Architectures



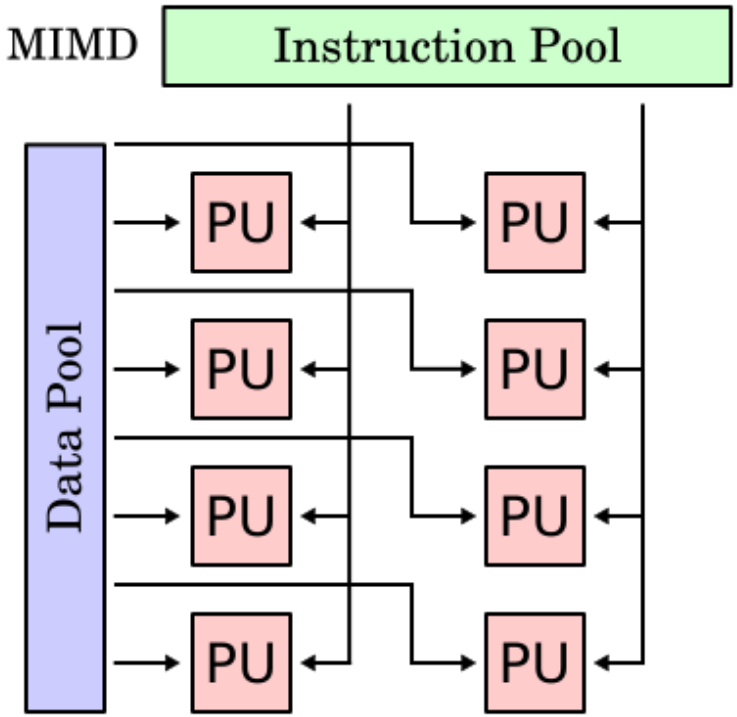
Classification des Architectures



Classification des Architectures



Classification des Architectures



Type de parallèlisme

Objectifs du parallèlisme

- Pouvoir accélérer une application en
  - Divisant cette applications en sous-tâches
  - Exécuter ces sous-tâches en parallèles sur des unités différentes
- Il faut donc:
  - Trouver le parallélisme dans l'application
  - Trouver le bon grain de calcul/échange de données
  - Avoir des connaissances pour concevoir une solution efficace sur la machine cible

Le parallélisme par l'exemple

Etre attentif !

- Pour avoir une application parallèle, il faut que l'application soit décomposable en sous-problèmes suffisamment indépendants
- Il faut pouvoir organiser le travail à répartir.
- Surcoût dû à la répartition du travail
- Trouver le meilleur algorithme parallèle
- Pas forcement celui qui est le plus efficace en séquentiel !

Qu'espère-t-on ?

Une grosse accélération

Idéalement, on espère avoir une accélération de P sur P processeurs !

En général ...

- Malheureusement, c'est rarement le cas
  - Parties séquentielles d'un algorithme
  - Problèmes de surcoût dus à des calculs redonnants, les coûts de transfert de données
- Parfois le gain est superieur à P
  - Différences de vitesse mémoire (mémoire vive vs caches)
  - Moins de calcul grâce au parallélisme (recherche dans des arbres)
  - Applications pour lesquelles l'exécution sur un processeur est impossible

Type de parallèlisme

Le parallèlisme gratuit

- Parallélisme au niveau du bit (BLP, Bit Level Parallelism)
- Parallélisme d'instructions (ILP, Instruction Level Parallelism)
  - Exécuter plusieurs instructions par cycle d'horloge
  - Super-scalar, VLIW, EPIC
- Parallélisme au niveau du système

Limitations

- Niveau d'intelligence des processeurs et des compilateurs
- Complexité des applications
- Nombre d'éléments en parallèle

Type de parallèlisme

Par type d'interaction

- Mémoire partagée
- Passage de message

Par type de décomposition

- Parallélisme de tâche
- Parallélisme de donnée

Modèle de programmation

Mémoire partagée

- Chaque processus interagit de manière asynchrone avec un banc mémoire globalement accessible
- Gestion manuelle des verrous (mutex, sémaphores, etc...)

Mise en oeuvre

- pthread
- OpenMP

Mémoire partagée

OpenMP

- Version séquentielle

```
1  for(int i=0;i<size;++i)
2    out[i] = (a[i] +b[i])*0.5;
```

- Version paralléle

```
1  #pragma omp for
2  for(int i=0;i<size;++i)
3    out[i] = (a[i] +b[i])*0.5;
```

Modèle de programmation

Passage de Message

- Chaque processus interagit avec les autres via l'envoi de message contenant des données.
- Communications point à poitn ou collectives
- Difficulté : Ou sont les données ?

Mise en oeuvre

- MPI
- Charm++

Passage de Message

MPI

- Version paralléle

```
1  MPI_Scatter(a,size/n,MPI_DOUBLE
2              ,a,size,MPI_DOUBLE,0,MPI_COMM_WORLD
3              );
4  MPI_Scatter(b,size/n,MPI_DOUBLE
5              ,b,size,MPI_DOUBLE,0,MPI_COMM_WORLD
6              );
7
8  for(int i=0;i<size/n;++i)
9    out[i] = (a[i] +b[i])*0.5;
10
11 MPI_Gather(out,size/n,MPI_DOUBLE
12            ,out,size,MPI_DOUBLE,0,MPI_COMM_WORLD
13            );
```

Métriques utiles

Objectifs

- Mesurer la qualité d'un code paralléle
- Savoir repérer les problèmes
- Comaprer des implantations

Types de métriques

- Métrique d'évaluation
- Métrique de diagnostique

Accélération

Définition

$$\Gamma_p = \frac{T_1}{T_p}$$

avec:

- **p** : nombre d'éléments de calcul
- T_1 : temps d'exécution séquentiel
- T_p : temps d'exécution paralléle

Efficacité

Définition

$$E_p = \frac{\Gamma_p}{p} = \frac{T_1}{pT_p}$$

avec:

- **p** : nombre d'éléments de calcul
- Γ_p : accélération idéale pour p éléments de calcul
- T_1 : temps d'exécution séquentiel
- T_p : temps d'exécution paralléle

Loi d'Amdahl

Définition

- Soit:
  - $p \in \mathbb{N}$, le nombre d'éléments de calcul
  - $f \in [0, 1]$, le ratio purement séquentiel d'un algorithme
  - $T(p)$ le temps d'exécution parallèle définie comme:

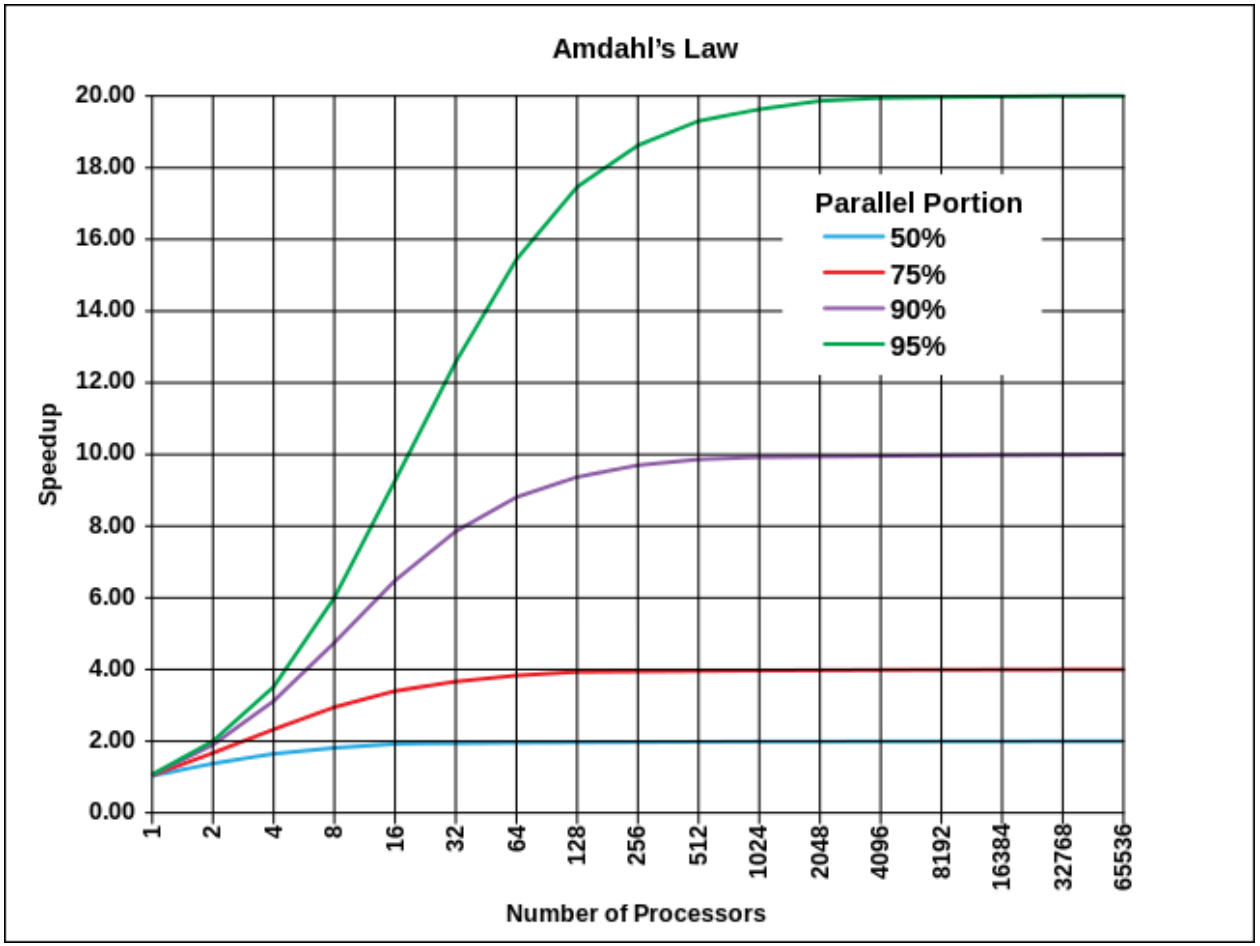
$$T(p) = T(1)\left(f + \frac{1}{p}(1 - f)\right)$$

- Alors:

$$\Gamma(p) = \frac{T(1)}{T(p)} = \frac{T(1)}{T(1)\left(f + \frac{1}{p}(1 - f)\right)} = \frac{1}{f + \frac{1}{p}(1 - f)}$$

Loi d'Amdahl

$$\lim_{p \to \infty} \Gamma(p) = \frac{1}{f}$$



Loi de Gustafson-Barsis

Définition

$$\Gamma(p) = f + p \cdot (1 - f)$$