



Master Informatique

Programmation distribuée M1 / 178UD02

C1 – Introduction aux systèmes distribués

Thierry Lemeunier

thierry.lemeunier@univ-lemans.fr

Plan du cours

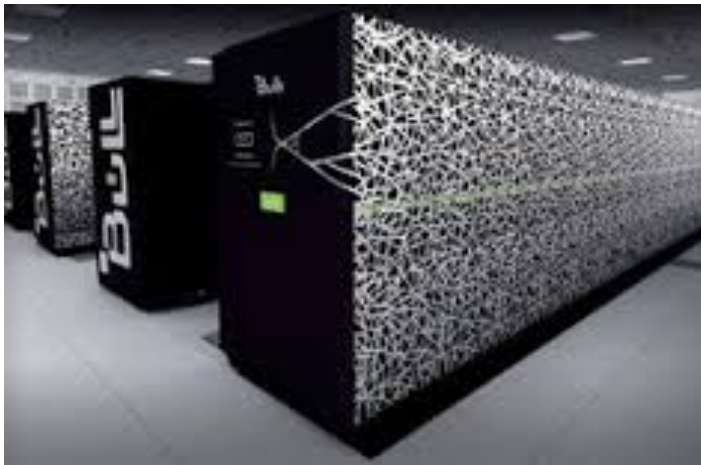
- Une définition des systèmes distribués
- Architectures des systèmes distribués
- Les applications distribuées
- Les grands paradigmes de communication

Une définition des systèmes distribués (1/2)

■ Origine des systèmes distribués :

- A l'origine : systèmes centralisés (*mainframe*) effectuant seuls toutes les tâches
- Création de microprocesseurs de plus en plus puissant et bon marché
- Invention des réseaux locaux (*Local-Area Network*) et globaux (*Wide-Area Network*)

➔ Apparition de systèmes distribués : un système informatique plus ou moins complexe constitué d'un ensemble de sous-systèmes communiquant



Supercalculateur Curie (33^{ème} en 2014 sur www.top500.org)
du Commissariat à l'Energie Atomique (CEA) développé par Bull SA

- 5040 nœuds de : 64 Go et 2 processeurs Intel Xeon 8 cœurs 2.7Gh
- 10080 processeurs 8 cœurs (= 80640 cœurs au total)
- Mémoire totale : 322 téraoctets (= 322 560 Go)
- Stockage central : 5 pétaoctet (= 5000 To)
- OS : Linux optimisé par Bull et le CEA

En 2014, la puissance de crête est de 2 pétaflops (2 000 000 de milliards d'opérations sur des nombres flottants par seconde).

Une définition des systèmes distribués (2/2)

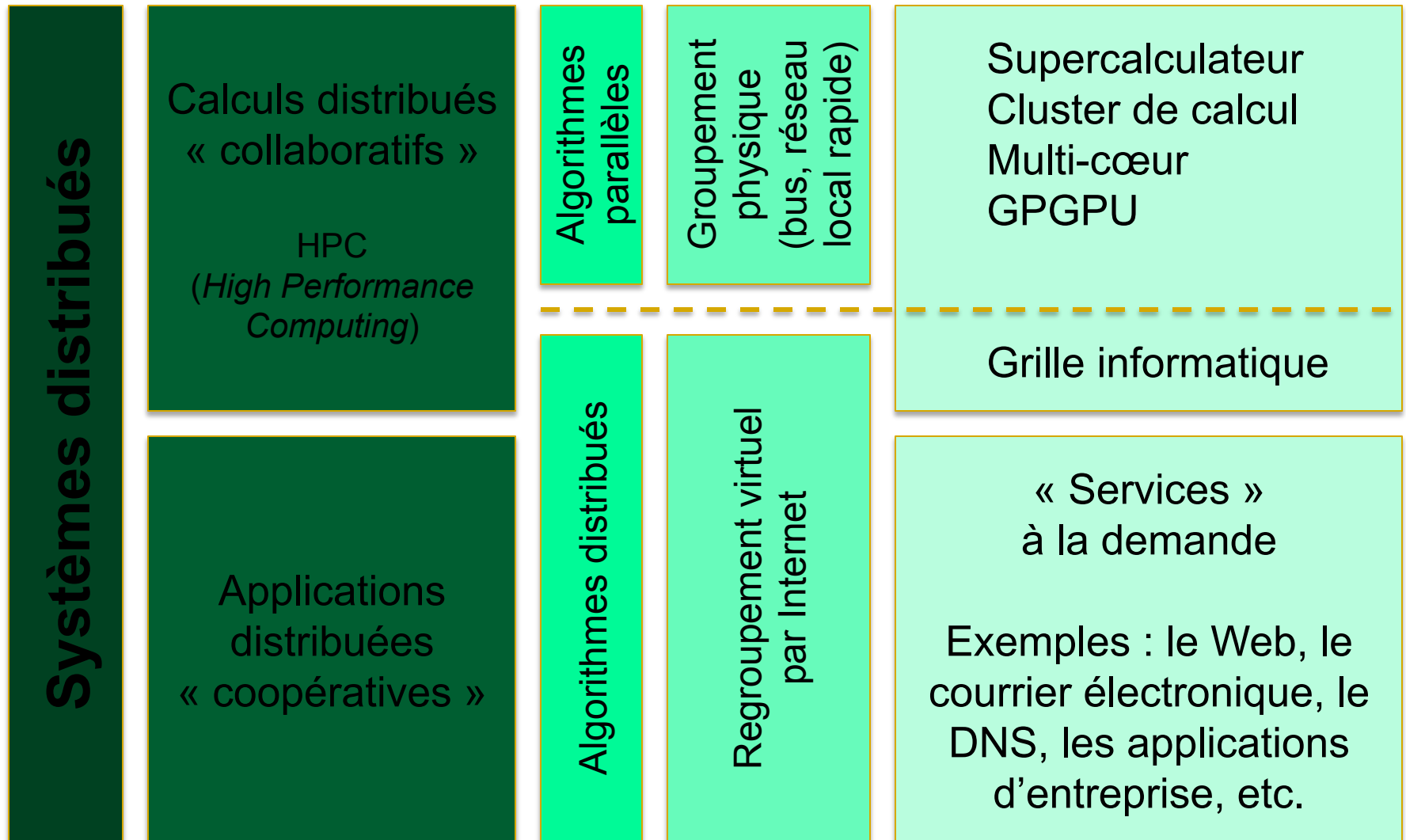
- Une définition de Andrew S. Tanenbaum :

« *A distributed system is a collection of independant computers that appears to its users as a single coherent system* »

- ➔ Les ordinateurs sont indépendants
 - ➔ Facilité d'extension et augmentation de la taille
 - ➔ Fonctionnement continu avec maintenance « à chaud »
 - ➔ La distribution géographique est possible
- ➔ Les utilisateurs ont l'impression d'utiliser un système unique
 - ➔ La communication et la distribution sont invisibles
 - ➔ Accès uniforme aux ressources quelque soit l'endroit et le moment
- ➔ Définition large qui regroupe tout ce qui n'est pas *mainframe* ou *PC offline* en partant des applications client-serveur les plus simples jusqu'au système massivement parallèle !

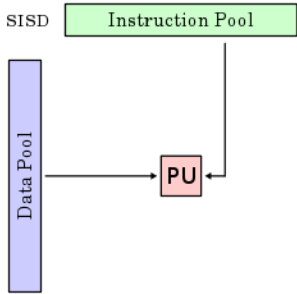
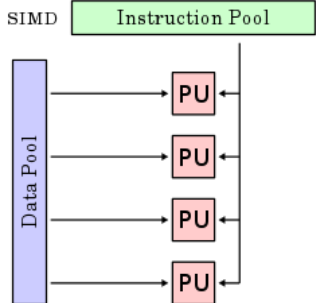
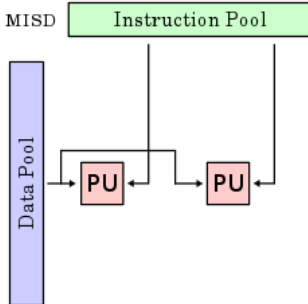
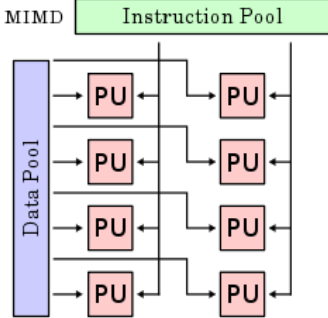
Cette définition = objectifs à atteindre

Architecture des systèmes distribués (1/13)



Architecture des systèmes distribués (2/13)

- Le calcul parallèle s'appuie sur :
 - des structures physiques spécifiques (très nombreuses)
 - des algorithmes spécialités (*hors du cadre de ce cours*)
- Taxonomie de Flynn (pas forcément liée à une architecture physique)

	Single Data	Multiple Data
Single Instruction	SISD Architecture Van Neumann classique 	SIMD Processeur scalaire 
Multiple Instruction	MISD 	MIMD 
	« <i>Parallélisme de traitement</i> »	« <i>Parallélisme de données</i> »

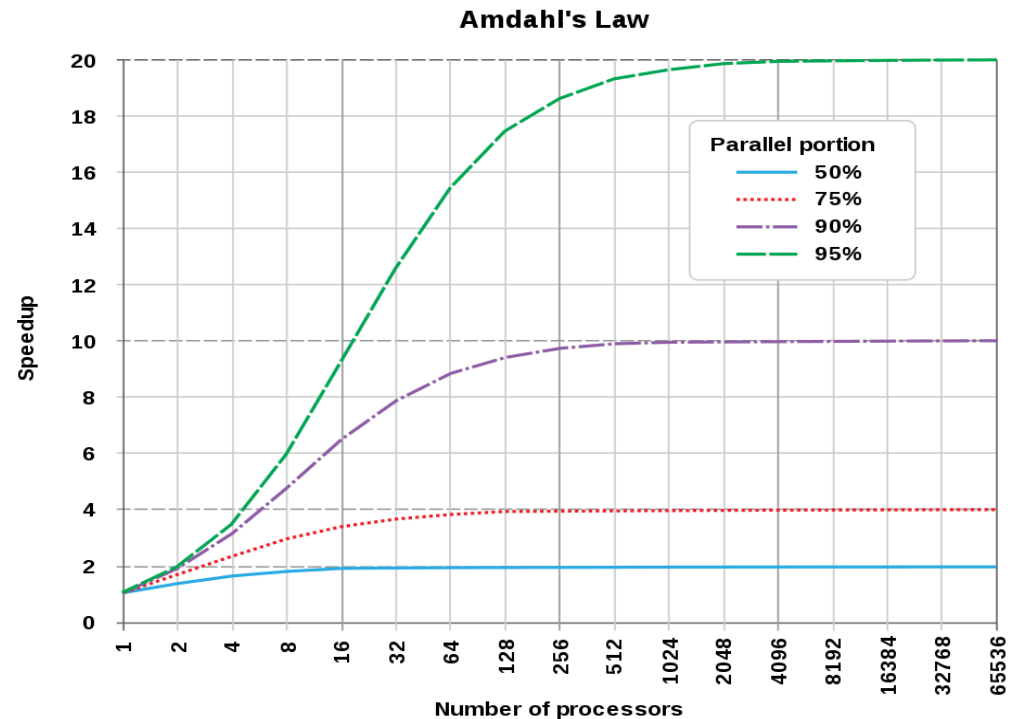
Architecture des systèmes distribués (3/13)

■ Mesure de la performance en HPC

- Floating-point Operations Per Second (FLOPS)
- ≠ Million Instructions Per Second (MIPS)
- Logiquement, le temps de traitement devrait être une fonction inverse directe et linéaire du nombre d'unité de traitement

kiloFLOPS	kFLOPS	10^3
megaFLOPS	MFLOPS	10^6
gigaFLOPS	GFLOPS	10^9
teraFLOPS	TFLOPS	10^{12}
petaFLOPS	PFLOPS	10^{15}
exaFLOPS	EFLOPS	10^{18}

- Loi d'Amdahl (années 1960) :
 - « mesure » empirique du degré optimal du parallélisme
 - Tout programme contient des parties parallélisables et des parties séquentielles non parallélisables



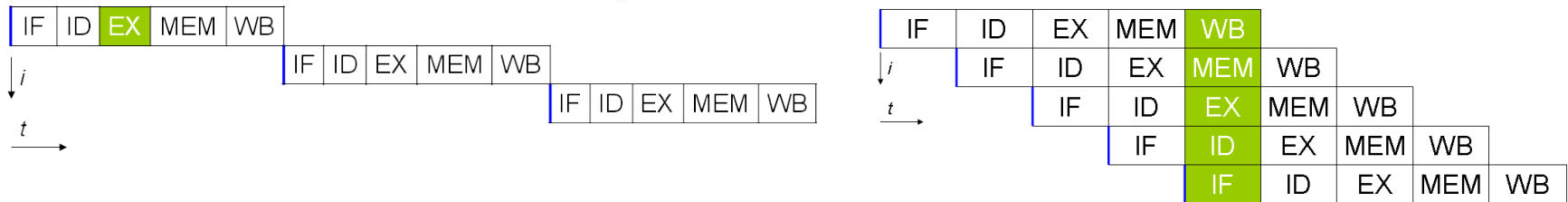
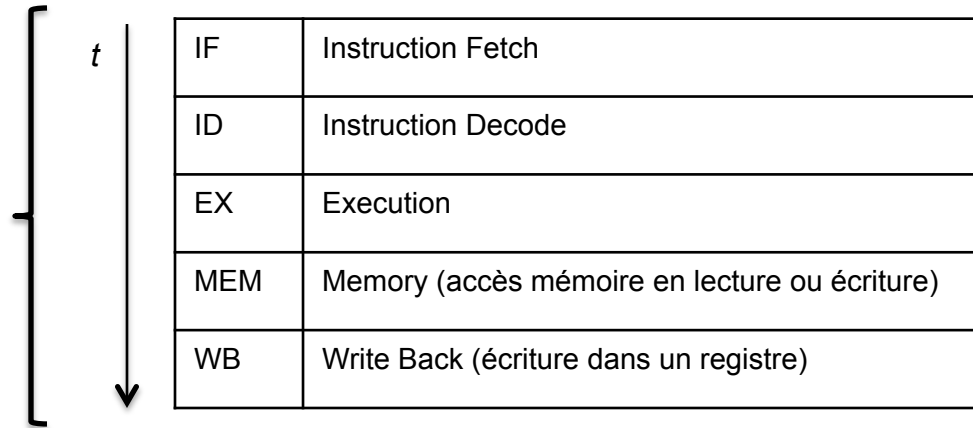
Architecture des systèmes distribués (4/13)

■ Comment faire du parallélisme matériel

□ Exécution en // sur un seul processeur (CPU)

- Nécessite des instructions indépendantes
- Utilise un seul CPU à **pipeline** (début années 1980 avec le Pentium d'Intel pour les PC)
- Processeur super-scalaire \neq processeur scalaire à une instruction par cycle

Cycle d'un CPU RISC
(*Reduced Instruction Set Computer*)

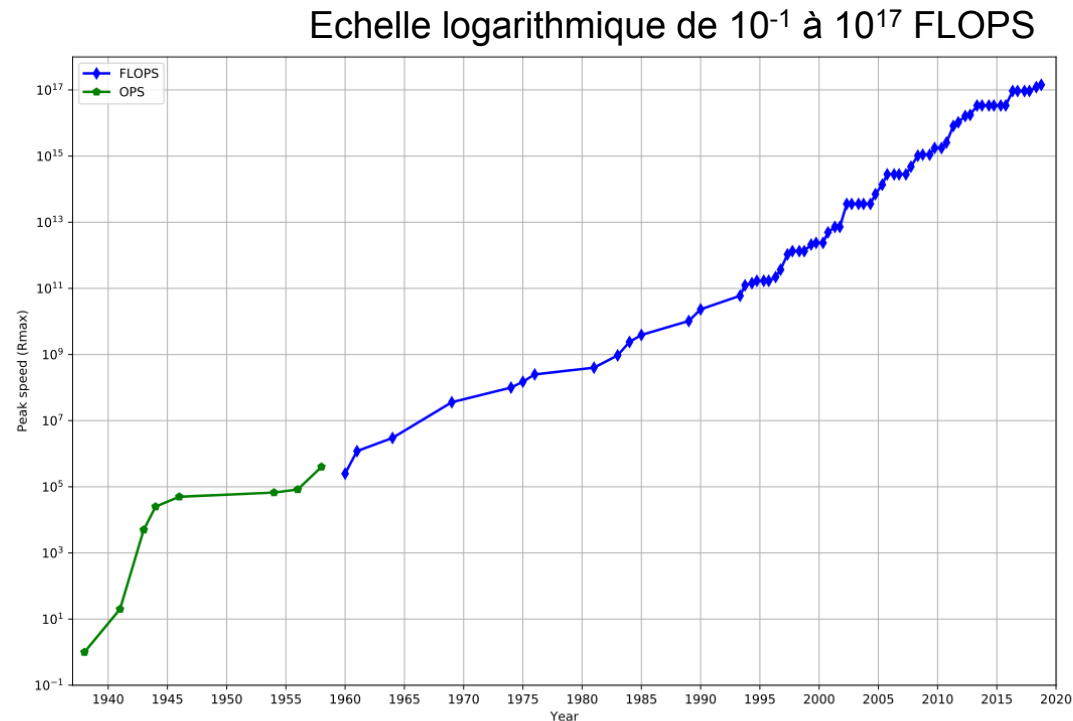


- Exécution en // sur plusieurs CPU (ou plusieurs cœurs)
- Exécution en // sur plusieurs cœurs graphiques (GPU)

Architecture des systèmes distribués (5/13)

■ Supercalculateur (de nos jours)

- ❑ Architecture massivement parallèle multi processeurs (des milliers de CPU) qui s'apparente au regroupement de centaines de cluster de calcul
- ❑ Eloignement le plus réduit possible entre processeurs et la mémoire
- ❑ Bus de communication spécifique très haut débit (fibre optique) de différentes topologies
- ❑ Site TOP500
<http://www.top500.org>
- ❑ OS : dérivé de GNU/Linux
- ❑ CPU : Intel principalement
- ❑ Problèmes matériels :
 - consommation énergétique
 - refroidissement
 - gestion de la cohérence des mémoires caches



Architecture des systèmes distribués (6/13)

■ Supercalculateur : accès à la mémoire principale

- Les CPU doivent être le moins éloignés possible de la mémoire utilisée
- Si il y a une seule mémoire (vive) principale
 - Latence d'accès et bande passante sont identiques pour chaque CPU :
UMA (Uniform Memory Access)
 - Latence d'accès et bande passante sont différentes pour chaque CPU :
NUMA (Non-Uniform Memory Access)

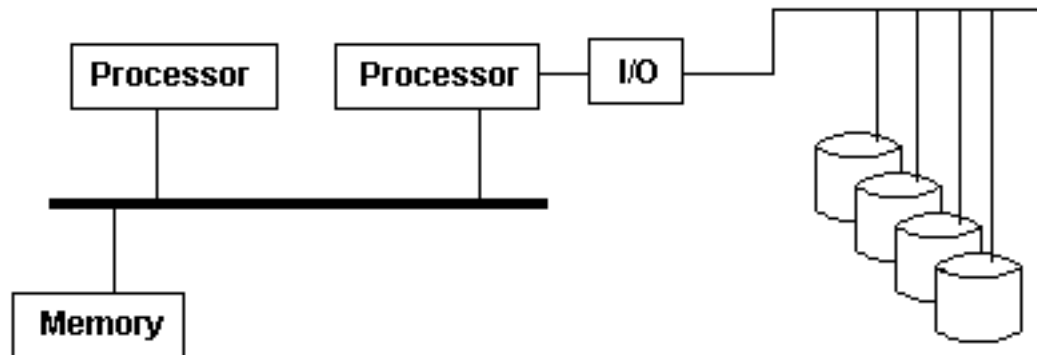
□ 3 modes d'accès à la mémoire principale (= au moins 3 architectures)

- UMA {
 - Mémoire partagée : même espace d'adressage à tous les CPU
 - AMP (Asymetric Multi Processing)
 - SMP (Symmetric Multi Processing)
- NUMA {
 - Mémoire distribuée :
 - Chaque CPU a sa mémoire et son espace d'adressage
 - Partage de données par message réseau
 - Mémoire partagée distribuée : DSM (Distributed Shared Memory)
 - Différentes zones mémoires avec différents bus

- **Problématique de la cohérence des données en cache** issues de la mémoire centrale : système de gestion physique de purge des données obsolètes

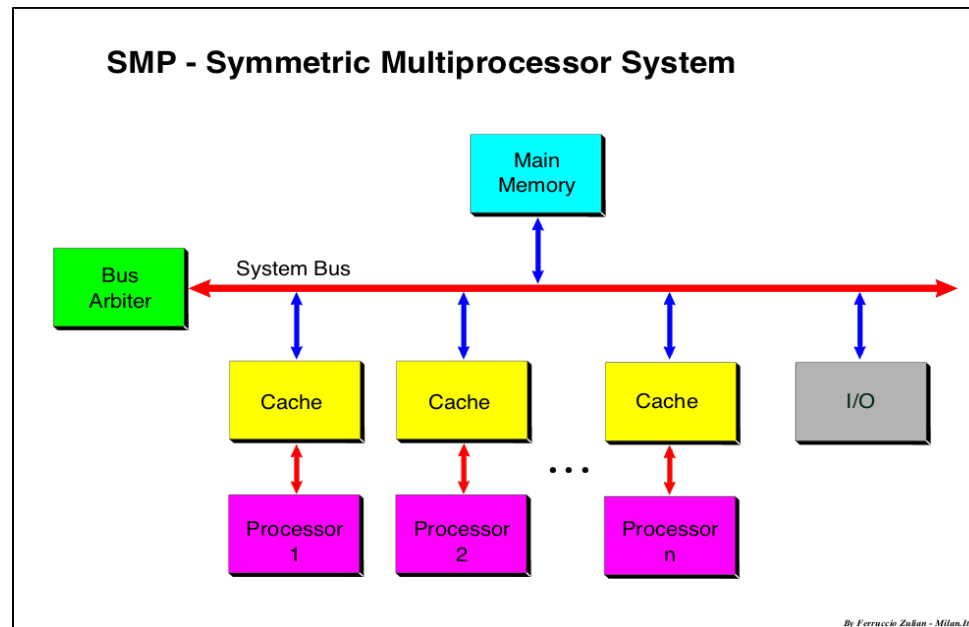
Architecture des systèmes distribués (7/13)

- Supercalculateur AMP (≈ années 1960-1970)
 - Plusieurs CPU identique
 - Accès mémoire de type UMA
 - Communication par bus
 - Certains CPU ont des rôles spécifiques
 - Exemples :
 - Un CPU exécute l'OS
 - Un seul CPU gère les périphériques I/O
 - Plusieurs CPU gèrent les périphériques I/O
 - Certains CPU gèrent certains périphériques I/O
 - Il existe une limite du nombre de CPU admissibles du fait du goulet d'étranglement de l'accès mémoire



Architecture des systèmes distribués (8/13)

- Supercalculateur SMP (≈ années 1960-1985)
 - ❑ OS identique exécuté sur chaque CPU identique
 - ❑ Accès mémoire de type UMA
 - ❑ Communication par bus ou matrice de connexion
 - ❑ Accès identique aux périphériques I/O
 - ❑ Limite du nombre de CPU (une dizaine au maximum) du fait du goulet d'étranglement de l'accès mémoire et de la complexité du système physique de gestion des caches

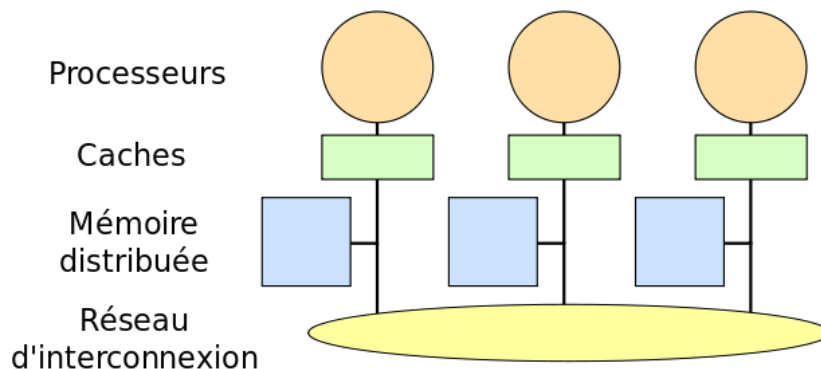


Architecture des systèmes distribués (9/13)

■ Supercalculateurs NUMA et DSM

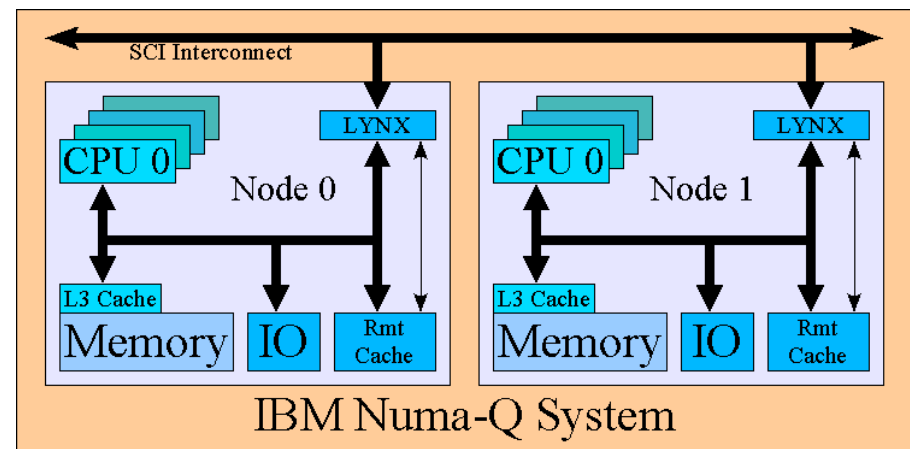
- ❑ NUMA : le temps d'accès dépend de la zone mémoire accédée
- ❑ Système de type NUMA « simple » (à partir des années 1990)
 - Mémoire distribuée pour chaque CPU (espace d'adressage privé)
 - Transmission d'informations par message réseau
- ❑ Système de type DSM (à partir des années 2000)
 - Hybridation = « SMP » + de la mémoire distribuée
 - Différentes zones mémoire sont accessibles par différents bus
 - ❑ une zone mémoire « locale » partagée par un groupe de CPUs : un node = CPU(s) + Zone mémoire
 - ❑ une zone mémoire « distante » :
 - par exemple : une zone mémoire partagée
 - par exemple : sur IBM Numa-Q (jusqu'à 4x16=64 CPU), accès à la mémoire des autres nodes

Système à mémoire distribuée



[Source : Wikipedia]

Système à mémoire partagée distribuée

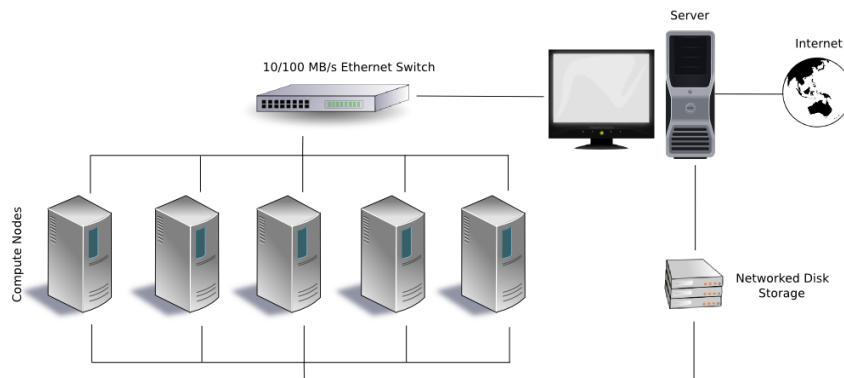


[Source : <http://lse.sourceforge.net/numa/>]

Architecture des systèmes distribués (10/13)

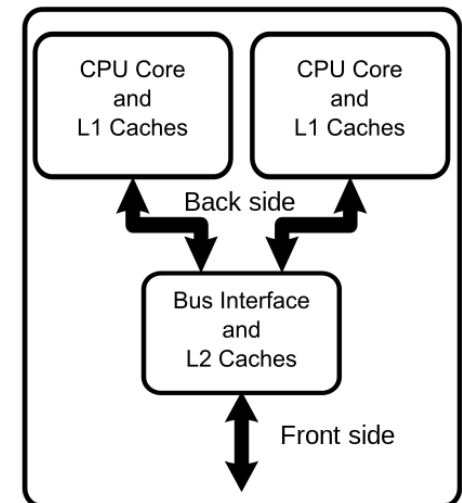
- Cluster de calcul ou grappe de calcul ou ferme de calcul (fin années 1970)
 - Groupement de plusieurs nœuds complets standard (grand public ou industriel) (CPU + RAM + Mémoire de masse + Interfaces I/O) :
 - Nœuds de calcul (homogènes exécutant le même Linux)
 - Nœuds de stockage partagé (partage NFS de partitions de disques par exemple)
 - Un ou plusieurs nœuds « frontaux »
 - Nœud frontal :
 - Machine d'accès et d'uniformisation (= une seule machine)
 - Répartiteur de charge ; gestion de la disponibilité (panne ; montée en charge)
 - Exécute un logiciel de type SSI (Single System Image)
 - Le tout est relié par un ou deux réseaux locaux
 - Un réseau standard (Ethernet) pour l'administration
 - Un réseau haute vitesse (GigaBit Ethernet, InfiniBand, etc.) peu étendu (quelques mètres) pour les nœuds de calcul et de stockage
 - Exemples de cluster « domestiques » : cluster de PS3, cluster de mini PC, etc.

Architecture typique d'un cluster



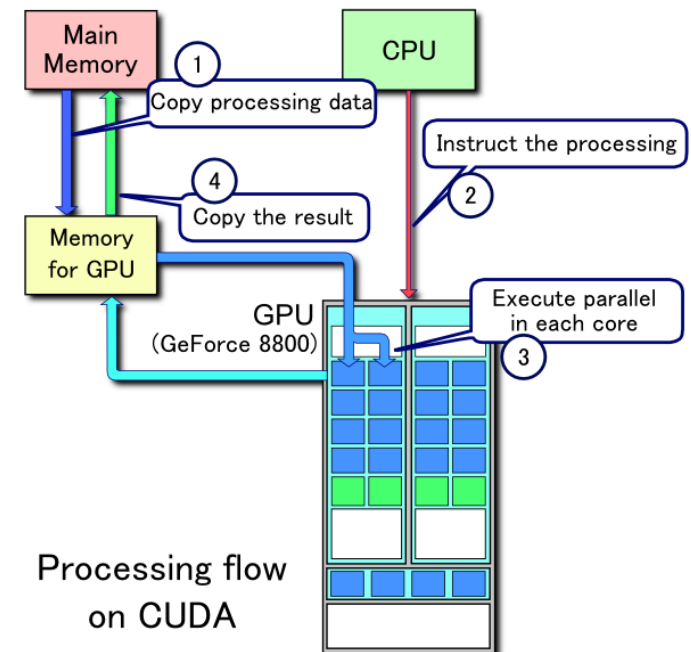
Architecture des systèmes distribués (11/13)

- Calcul parallèle sur CPU multi-cœur (à partir de 2006)
 - Un CPU constitué de plusieurs unités de traitement indépendantes appelée « cœur » ayant chacune sa mémoire cache
 - Chaque cœur exécute un flux d'instruction indépendant
 - De plus, chaque cœur peut être superscalaire
 - De plus, chaque cœur superscalaire peut être vu par l'OS comme deux cœurs virtuels (Simultaneous Multithreading) (disponible dès 2002)
 - Un CPU multi-cœur \approx miniaturisation d'un système SMP



Architecture des systèmes distribués (12/13)

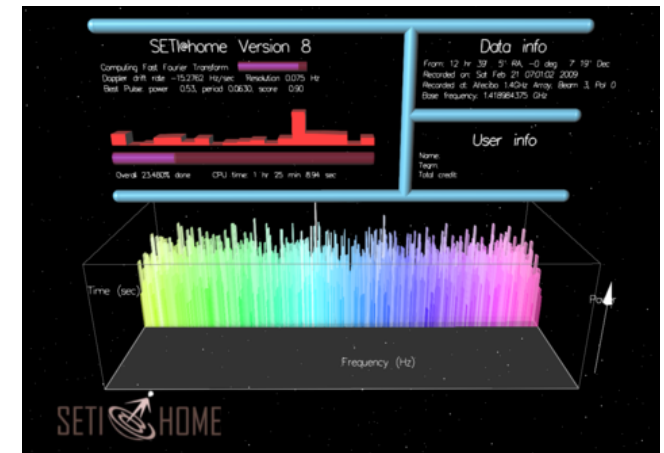
- Calcule parallèle avec le GPGPU (à partir de 2007)
 - GPGPU : General-Purpose computing on Graphics Processing Units
 - Un GPU est constitué d'une myriade de cœurs de traitement graphiques (*shaders*) programmables
 - Plateformes de programmation : transformer un programme // en primitives de traitement graphique
 - CUDA pour les cartes Nvidia
 - ATI Stream pour les cartes AMD
 - OpenCL (indépendant du matériel) par le consortium Kronos Group
 - On écrit une routine (« compute kernel ») qui est transférée puis exécutée en parallèle par les différents cœurs graphiques sur un ensemble de données
 - Un GPU \approx processeur vectoriel



Architecture des systèmes distribués (13/13)

- Grid Computing ou grille de calcul (à partir de 2000)
 - Forme de calcul parallèle la plus distribuée et (souvent) grand public
 - Supercalculateur virtuel de taille dynamique constitué de nœuds complets indépendants hétérogènes reliés par Internet
 - Un serveur central assure l'envoi des données et requête les nœuds pour des calculs (par exemple par HTTP)
 - Une grille est efficace dans le cas de calculs indépendants
 - Les nœuds sont
 - volontaires et à temps perdu
 - ou contractuels
- Exemples :
 - SETI@home : identification de signe d'intelligence extraterrestre
 - newGRID : étude des maladies neurodégénératives
 - plateforme BOINC (*Berkeley Open Infrastructure for Network Computing*) abritant une quarantaine de grilles actives
- Taille : des centaine de milliers de nœud (jusqu'à 600 000) pour des performances d'environ 30 PFLOPS pour BOINC

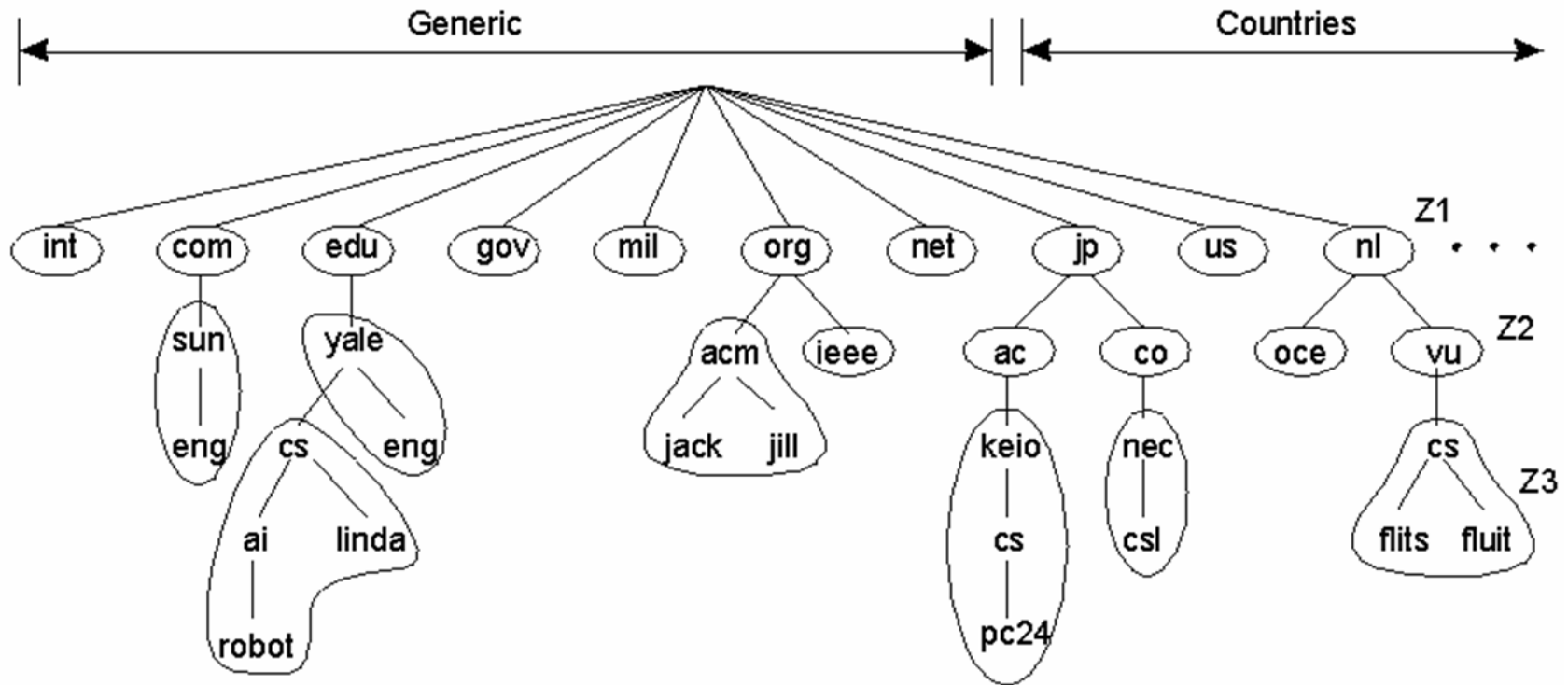
Interface du client SETI@home



Les applications distribuées

- Une application distribuée s'appuie sur un système distribué
- Les applications distribuées répondent :
 - à un besoin de puissance de calcul plus ou moins important
 - à un besoin de stockage de l'information plus ou moins important
 - à un besoin d'échange de données plus ou moins important
- Deux grands types d'applications :
 - Applications scientifiques ou de R&D
 - Physique nucléaire (simulation de réaction nucléaire...)
 - Prévision météorologique (modélisation du système climatique...)
 - Modélisation moléculaire (nouveau matériau, génétique...)
 - Réalité virtuelle (simulateur de vol...)
 - Conception coopérative (avionique...)
 - etc.
 - Applications d'entreprises ou commerciales ou grand public
 - Système de réservation nationaux ou internationaux
 - Les systèmes d'informations des entreprises
 - Application bancaires et d'assurances
 - Internet et le Web tel que : courrier, édition et diffusion de contenu, bureau virtuelle, commerce électronique, jeux en ligne...
 - etc.

Exemple d'une application distribuée : le DNS



- DNS = donne l'adresse IP (193.52.29.139) à partir d'un nom (lucke.univ-lemans.fr)
- Le « nommage » des machines accessibles sur Internet est réparti en grand domaines puis en zones, sous zones et ainsi de suite.
- Il y a un 1 serveur par domaine qui transmet la demande de localisation au serveur de la zone considérée qui transmet au serveur de la sous zone et ainsi de suite.
- La charge du service de résolution est ainsi répartie sur les différents serveurs du système.

Les grands paradigmes de communication

■ Les sockets réseaux

- ❑ API d'accès à la couche transport (TCP, UDP, etc.) des réseaux informatiques
- ❑ Premières implémentations : fin années 1970 / début 1980 (en cogitation depuis 1971)

■ RPC (*Remote Procedure Call*)

- ❑ Une machine cliente demande qu'une « procédure » soit exécutée sur une machine serveur et attend la réponse tout cela via un réseau de communication
- ❑ Premières implémentations : début des années 1980 (en cogitation depuis 1975)

■ Objets Distribués (Java RMI, CORBA, DCOM, etc.)

- ❑ « RPC orienté objet » : invocation d'une méthode d'un **objet distant**
- ❑ Premières implémentations : début des années 1990

■ MOM (*Middleware Oriented Message*)

- ❑ Le client et le serveur communiquent par **message**
- ❑ Premières implémentations : milieu des années 1990 ?
- ❑ Une dizaine de standards récents (années 2010) existent !

■ SOA (*Service-Oriented Architecture*)

- ❑ Le client et le serveur communiquent via des **composants** distribués
- ❑ Premières implémentations : début des années 2000 ?