Modèles et techniques en programmation parallèle hybride et multi-cœurs

Introduction, rappels sur l'architecture matérielle, différents types de parallélismes

Marc Tajchman

CEA - DEN/DM2S/STMF/LMES

10/08/2020

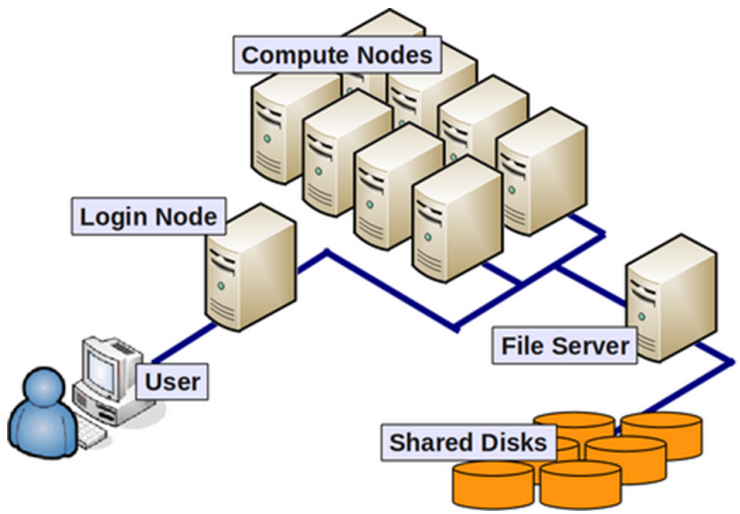
Rappels sur l'architecture matérielle

La plupart des ordinateurs actuels ont des fonctionnalités parallèles (exécution simultanée plusieurs traitements) :

- Les ordinateurs personnels (ordinateurs portables ou de bureau, téléphones portables, tablettes, etc.) ont presque toujours des unités de calcul (ou cœurs) multiples. Dans chaque cœur, il y a souvent plusieurs circuits arithmétiques (addition, multiplication, etc), des pipelines.
- Beaucoup d'ordinateurs ont de plus des dispositifs spécialisés dans le calcul (exemple: (GP)GPU, carte graphique utilisée pour du calcul "massivement parallèle")
- Les machines parallèles regroupent plusieurs nœuds de calcul (chaque noeud de calcul est similaire à un ordinateur personnel, sans clavier, ni souris), les nœuds de calculs sont connectés entre eux.

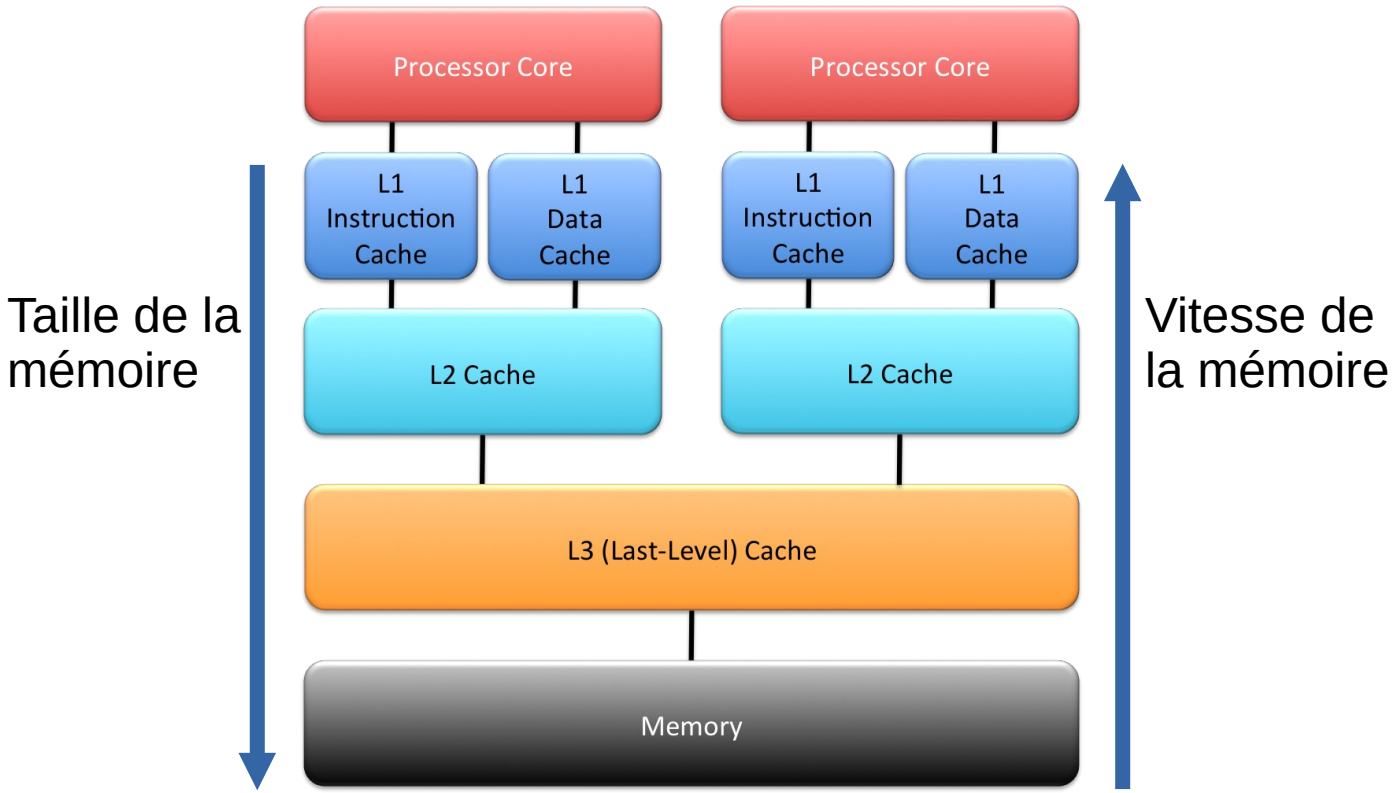
Machine parallèle typique:

- un ensemble de nœuds de calcul chacun contenant un ou plusieurs cœurs et de la mémoire, parfois un GPU
- les nœuds sont connectés entre eux par un réseau
- et aussi à un système qui gère les données d'entrée et produites (disques partagés)
- les utilisateurs ne se connectent pas directement aux noeuds de calcul mais passent par un frontal (ordinateur qui gère les connexions multiples à la machine parallèle)

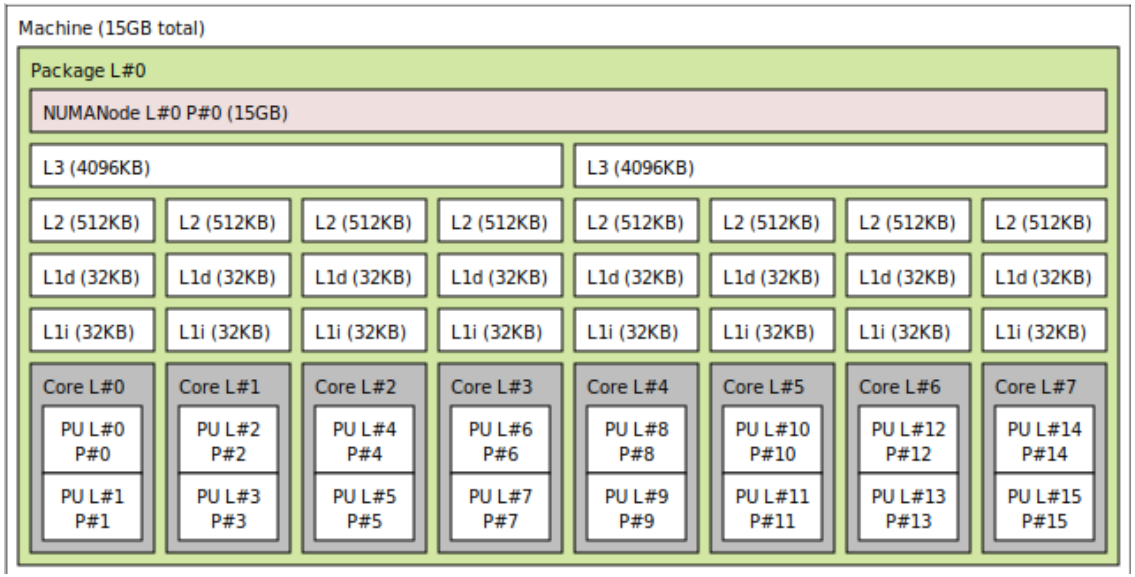


(les machines les plus puissantes actuellement contiennent plusieurs centaines de milliers de nœuds)

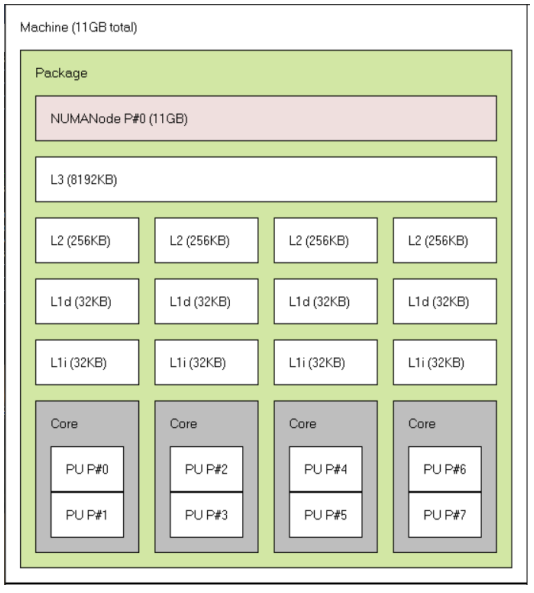
Schéma interne d'un nœud (variable suivant le modèle de processeur et la génération utilisée):

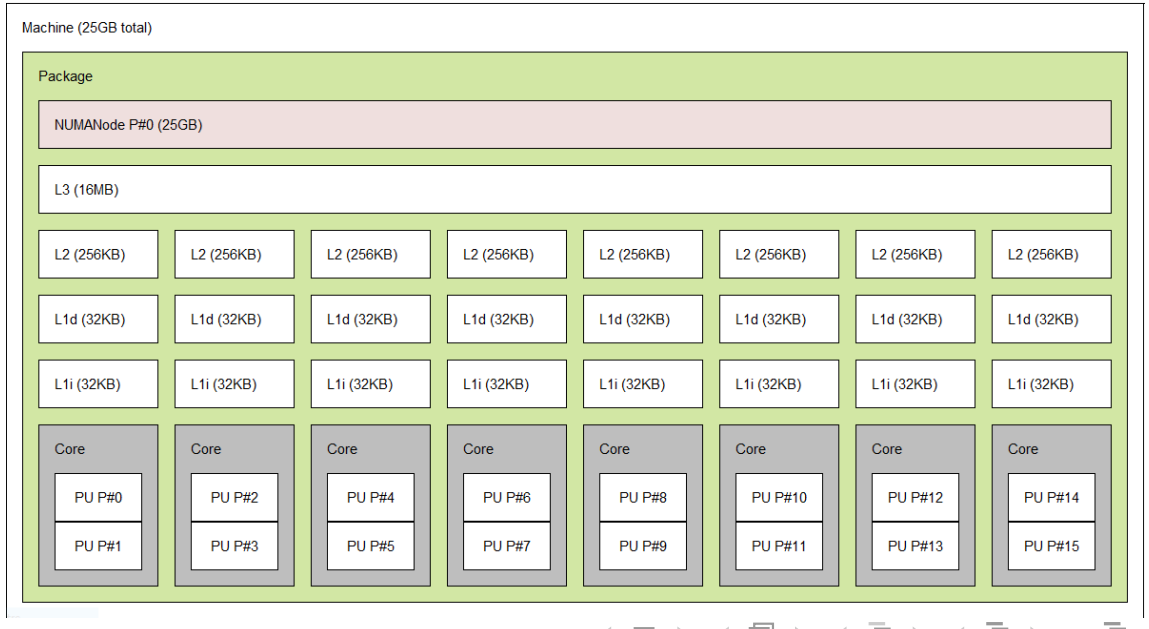


Exemple : processeur AMD Ryzen 4900



Exemples : processeurs Intel i5 8400H - i9 9900K





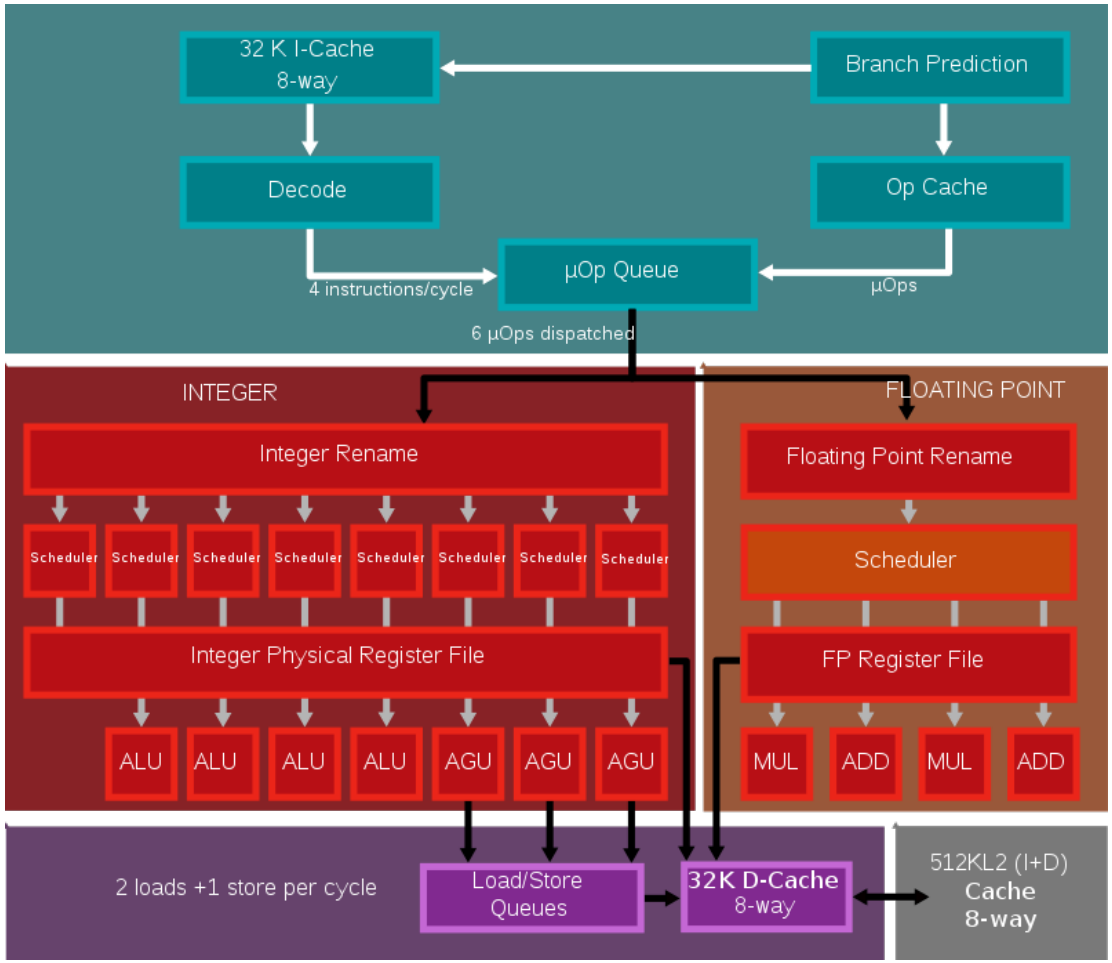
Jusque 6 types de mémoires différentes:

- registres : mémoire très rapide dans chaque cœur
- mémoires caches L1, L2, L3 : mémoires rapides, partagées partagées entre une partie des cœurs
- mémoire centrale : mémoire moins rapide mais de plus grande taille
- disques : mémoire lente.

A noter que la différence entre mémoire centrale et disque tend à diminuer avec la généralisation des SSD.

Les échanges entre la mémoire centrale et les mémoires cache se font par blocs de données (appelées “lignes de cache”) et pas “à l’unité” pour des raisons de performances.

Schéma interne d'un cœur (variable suivant le modèle de processeur et la génération utilisée):



AMD - Zen2 μarch (simplifié)

Situation actuelle (et encore pour plusieurs années):

vitesse de calcul (d'un processeur)

- $\approx$ *vitesse de la mémoire interne (registres) du processeur*
- $>$ *vitesse des différentes mémoires cache, intermédiaires entre le processeur et la mémoire centrale du nœud*

  (vitesse L1 $>$ L2 $>$ L3, taille L1 $<$ L2 $<$ L3)
- $\gg$ *vitesse de la mémoire centrale d'un nœud*
- $\gg$ *vitesse du réseau qui connecte les nœuds*

La gestion des différentes mémoires est faite par une puce appelée contrôleur mémoire.

Fonctionnement :

- Quand un cœur a besoin d'une donnée, il la demande au contrôleur mémoire.

- Le gestionnaire mémoire regarde si la donnée est dans la mémoire cache L1, L2 ou L3 de ce processeur, sinon le bloc de la mémoire centrale qui contient la donnée est recopié dans la mémoire cache (dans L3, puis L2 et L1).

- Ensuite, la donnée est recopiée dans un registre de ce cœur.

- Quand un cœur a modifié une donnée, il le notifie au gestionnaire mémoire
- Le gestionnaire mémoire recopie la donnée vers la mémoire cache (L1, L2 puis L3) et vers la mémoire centrale
- Il vérifie aussi qu'une autre copie de la donnée n'existe pas dans la mémoire cache d'un autre cœur.
- Si c'est le cas, les autres copies sont mises à jour

Ce système s'est généralisé parce qu'il est très compliqué/cher de produire de la mémoire qui suive les performances des processeurs.
Permet de maintenir de bonnes performances avec de la mémoire de grande taille mais plus lente.

Cette gestion peut représenter une partie importante du temps d'exécution de programmes parallèles (pour assurer la cohérence des différentes parties de la mémoire et leurs mises à jour correctes).

Ce qu'il faut retenir:

- En général, plusieurs niveaux de parallélisme disponibles
  1. Entre les nœuds de calcul (avec par ex. MPI)
  2. Entre les cœurs dans un même nœud de calcul (avec par ex. MPI ou OpenMP)
  3. Dans un cœur, entre les circuits arithmétiques
  4. Dans un GPU (avec par ex. Cuda/OpenCL)
- On peut combiner ces différents niveaux de // (en faisant un peu attention)
- Dans ce cours, on s'intéressera principalement:
  - au // entre cœurs (OpenMP)
  - au // dans un GPU
  - au // mixte (MPI/OpenMP/Cuda)