Modèles de Programmation Parallèle

Imane YOUKANA

Université Mohamed Khider - Biskra

Programmation Parallèle ?

Effectuer beaucoup de calculs en utilisant plusieurs cœurs ou processeurs

- Le principe du parallélisme est simple
  - Exécuter en même temps des instructions indépendantes
- La mise en œuvre nécessite de connaître
  - Les architectures des machines parallèles
  - Les techniques de parallélisation
  - Les techniques de programmation

Introduction à l'algorithmique et la programmation parallèles

- Savoir Paralléliser un problème en fonction de la cible
- Savoir mettre en œuvre en fonction du paradigme de parallélisation

Architecture - Machines parallèles

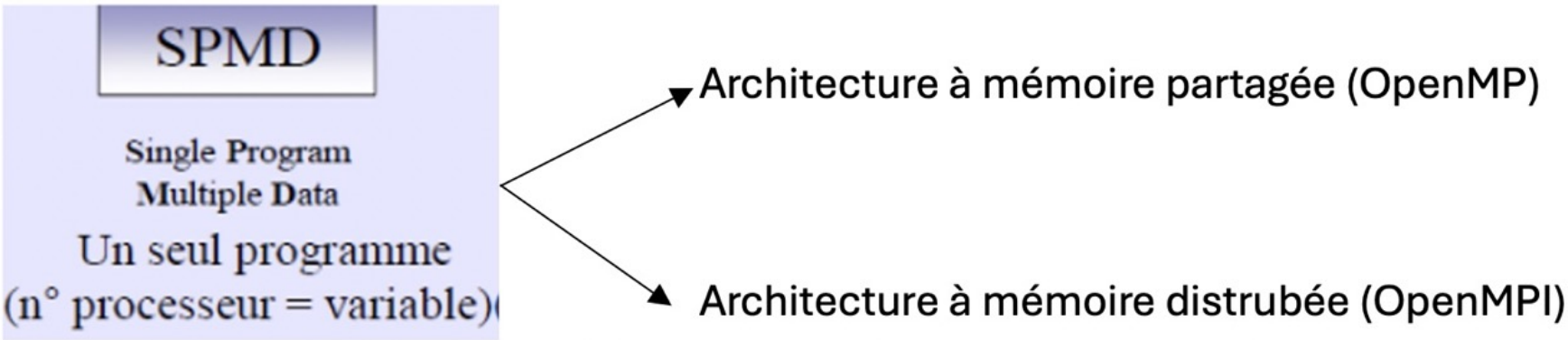
La classification de Flynn : en fonction du flot d'instructions et de données

- Single Instruction Single Data (SISD)
- Single Instruction Multiple Data (SIMD)
- Multiple Instruction Multiple Data (MIMD) –> SPMD
- Multiple Instruction Single Data (MISD)

Modèle dominant : SPMD

SPMD: un seul programme copié sur chaque processeur qui s'exécute de manière asynchrone

Architecture - Machines parallèles

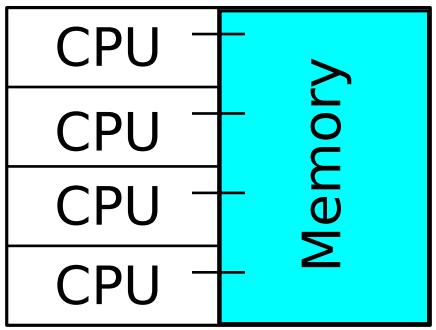


Classification simplifiée

- Architecture à mémoire partagée
- Architecture à mémoire distribuée

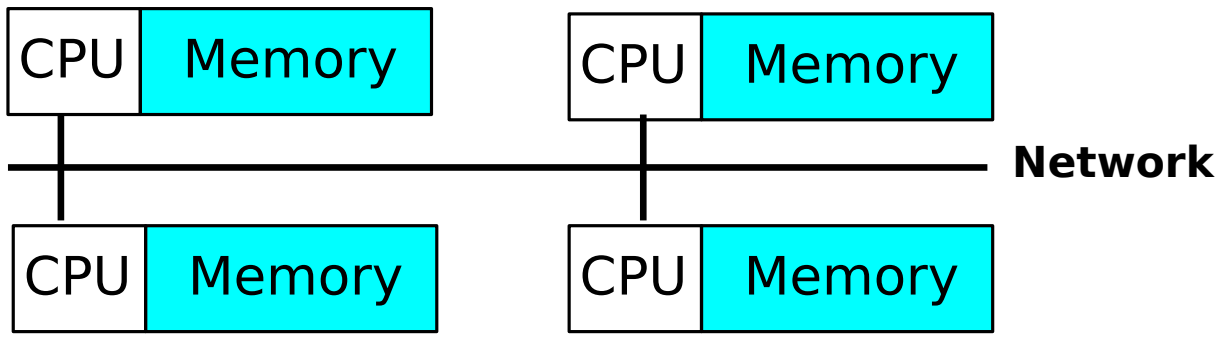
Classification simplifiée

- Architecture à mémoire partagée
  * Tous les processus partagent le même espace mémoire
  * Cela semble facile à programmer car on n'a pas à se soucier de ce que chaque processus peut voir
  * Cependant il faut gérer dans le code les accès concurrents en mémoire ce qui est complexe (synchronisations)
  * En pratique on a de grandes pertes de performances si les données ne sont pas bien gérées, c'est en fait très compliqué
- Architecture à mémoire distribuée



Classification simplifiée

- Architecture à mémoire partagée
- Architecture à mémoire distribuée
  * Chaque processus est indépendant en mémoire
  * Il faut gérer la distribution des données aux différents processus
  * Il faut éventuellement introduire des communications pour échanger des informations utiles entre processus



Les modèles du parallélisme

Qu'est ce qui est indépendant ?

- Le parallélisme de contrôle
  - Faire plusieurs choses en même temps
- Le parallélisme de données
  - Répéter une action sur des données similaires
- Le parallélisme de flux
  - Travailler à la chaine

Les modèles de programmation

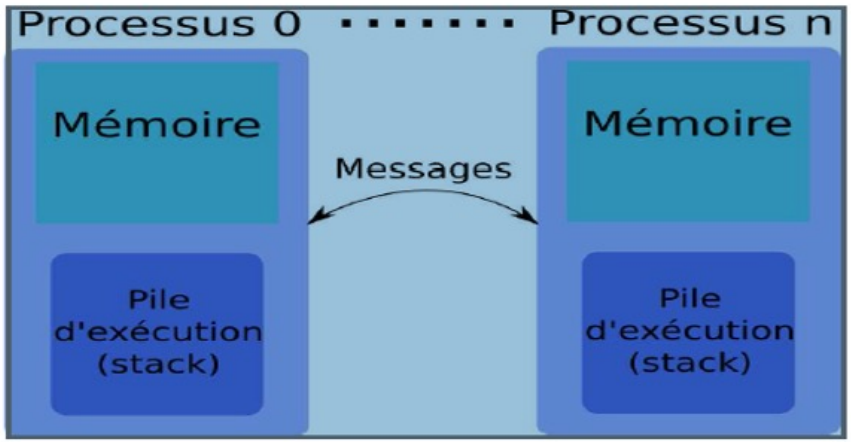
Architecture à mémoire partagée

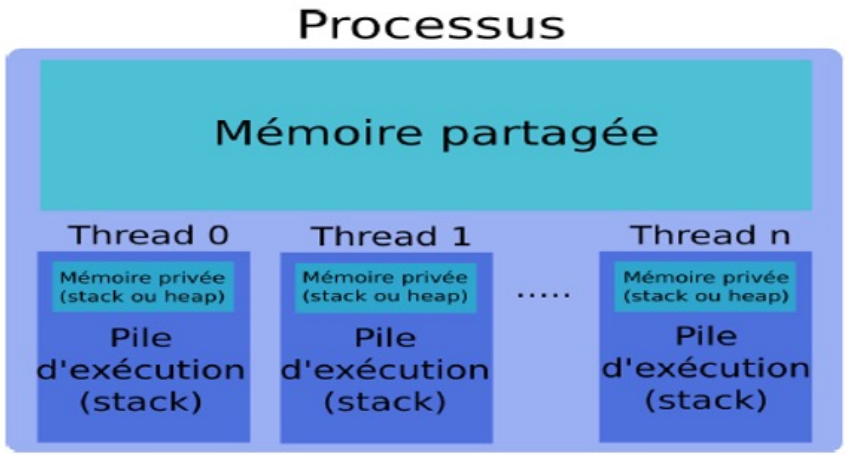
- PRAM = Parallel Random Access Machine (modèle abstrait)
- OpenMP, HPF

Architecture à mémoire distribuée

- BSP = Bulk Synchronous Parallelism
- Programmation parallèle par passage de messages
  - PVM, MPI

Modèles de programmation





OpenMPI versus OpenMP

Processus vs. thread

Les processus et les threads représentent des instances de code en cours d'exécution (donc une notion dynamique).

Processus

- représente un programme en cours d'exécution.
- qui possède un espace mémoire privé, indépendant de celui des autres processus. –(deux processus qui veulent collaborer doivent le faire en utilisant un mécanisme d'échange de messages) — par exemple, pour des processus Unix, en utilisant des pipes.
- Le contexte d'un processus est lourd : code, registres (dont Stack Pointer, Frame Pointer et Instruction Pointer), pile, tas (heap) , fichiers, vecteur d'interruptions, etc.
- Créer et activer un processus — ainsi qu'effectuer un changement de contexte — est donc (très!) coûteux!

Processus vs. thread

Thread

- représente (généralement) une fonction (une méthode) en cours d'exécution.
- Un processus peut contenir un ou plusieurs threads. Ces threads partagent alors la mémoire (e.g., code, tas) ainsi que diverses autres ressources. Parce qu'ils partagent un même espace mémoire, deux threads qui veulent collaborer peuvent le faire par l'intermédiaire de variables partagées.
- Le contexte d'un thread est léger : registres (dont IP), pile.
- Créer et activer un thread et effectuer un changement de contexte est donc moins coûteux que dans le cas d'un processus!

Questions à se poser

Machine à mémoire partagée :

- comment répartir les calculs ?
- comment limiter ou gérer les accès concurrents aux données

Machine à mémoire distrubée:

- comment répartir les données ?
- comment répartir les calculs ?