

Programmation Répartie - Introduction aux modèles et paradigmes de programmation parallèle et distribuée.

T.DUFAUD

UVSQ - IUT Vélizy - Informatique

UNIVERSITÉ DE
VERSAILLES
ST-QUENTIN-EN-YVELINES



IUT DE
VÉLIZY

R5-05 - 11/2025

Contexte de ce
cours

Modèles de
programmation

Paradigme de
programmation
parallèle

Conception

- 1 Contexte de ce cours
- 2 Modèles de programmation
- 3 Paradigme de programmation parallèle
- 4 Conception

Conception d'algorithme parallèle

- Quels sont les concepts que l'on peut utiliser pour **analyser ou écrire un algorithme parallèle** ?
- Quels **aspects** doit-on considérer ?
- Quelles **structures** d'algorithmes peuvent être envisagées ?

Modèle de programmation et paradigme

- On considère des **Modèles de programmation, relatifs à la traduction d'un algorithme**
- Il n'y a pas de modèle unique en programmation parallèle !
- **Plusieurs aspects** sont à considérer : Architecture matérielle, logicielle, algorithme
 - **Architecture** à mémoire distribuée ou à mémoire partagée, ou les deux ? Hardware homogène, hétérogène, multiniveaux ?
 - Critère de **qualité logicielle** : modularité, portabilité des codes et des performances, maintenabilité...
 - **Algorithme** : structure des données, dépendance entre les données, dépendance entre les tâches...
- Le parallélisme peut s'exprimer selon différentes **structures d'algorithme : paradigmes**
 - exemple : client/serveur

Contexte de ce
cours

Modèles de
programmation

Paradigme de
programmation
parallèle

Conception

Modèles de programmation

Modèle de programmation parallèle

- Lié à l'**expression de l'algorithme**
- identifie quelles sont les **entités parallèles du programme...**
- ... et comment sont exprimées **leurs interactions**
- On distingue deux grandes classes de modèle de programmation :
 - Parallélisme de **tâche**
 - Parallélisme de **donnée**

Parallélisme de tâche

- **décomposition d'une tâche en plusieurs sous-tâches**
- variables partagées entre les (sous-)tâches **OU** communication par message entre tâche
- **on associe une tâche à un processus** (support d'exécution de la tâche)

Parallélisme de donnée

Manipulation de structure homogène

- on peut considérer un tableau (dans lequel on place les données)
- **Haut niveau d'expression du parallélisme**, à chaque case du tableau on peut associer une tâche.
- **Découpage des structures de donnée sur les processus** (Un ensemble de case, *structure locale*, est manipulée par un processus)
- **Chaque processus travaille sur la structure locale**
- à la fin d'un travail on restructure les données → communication

Remarque

Ne pas confondre avec les modèles de fonctionnement qui sont liés à l'architecture (cf. Taxonomie de Flynn 1966)

- SISD : Single Instruction Single Data
- SIMD : Single Instruction Multiple Data
- MIMD : Multiple Instruction Multiple Data

Paradigme de programmation parallèle

Paradigmes

- **Structure d'algorithme** pour exécution sur une machine parallèle.
- Un programme est la combinaison de plusieurs paradigmes.
- liste non-exhaustive :
 - parallélisme de phase
 - itération parallèle
 - diviser pour régner (parallélisme récursif)
 - pipeline
 - Maître/Esclave
 - Client/Serveur
 - SPMD
 - ...

- Les **tâches d'une boucle sont exécutées indépendamment**
- Par exemple : lancer plusieurs mobile. `for (i=1:NB_MOBILES) do mobile[i].start()`

Remarques : Modèle de programmation : *par tâche*, paradigme : *parallélisme itératif*

Maître

- Exécute le code séquentiel
- Initie des processus esclaves
- transmet du travail à des processus esclaves (effectue des requêtes à des Esclaves)
- attend le résultat des esclaves

Esclave

- Attend le travail (envoyé par le maître)
- Exécute le travail (en parallèle des autres Esclaves)
- retourne le résultat au Maître

Remarques

- **Avantage** : simple à mettre en oeuvre, communication de 1 à tous.
- **Inconvénient** : Goulot d'étranglement
 - distribution des calculs de 1 pour tous
 - centralisation des résultats

Client

- Effectue des requêtes à un serveur

Serveur

- Effectue le travail correspondant aux requêtes (envoyées par le client)
- retourne le résultat au Client

Remarques

- **Avantage** : simple à mettre en oeuvre, communication de tous à 1.
- distribution du résultat
- centralisation des calculs

Single Program Multiple Data

- Un même programme
- Exécuté sur des données structurées et distribuées
- Exemple : parcours d'une structure de donnée par une boucle. Le processus n'exécute que les parties qui concerne les données qu'il gère.

Contexte de ce
cours

Modèles de
programmation

Paradigme de
programmation
parallèle

Conception

Conception

Les questions à se poser

- Où est le parallélisme dans l'application ?
 - dépendance entre données ?
 - section critique ?
- Découpage en tâches élémentaires
 - évident ?
 - transformer le programme ?
 - l'algorithme ?
- Partitionnement des données