Calcul parralèle. Modèle PRAM

Mihaela JUGANARU-MATHIEU
mathieu@emse.fr

École Nationale Supérieure des Mines de St Etienne

2012-2013

Plan

Modèles du calcul séquentiel :

- machine de Turing
- $\rightarrow$ modèle RAM (Random Access Memory)

Complexité d'un algorithme séquentiel A :

- temps : $T_s(A_n)$
- espace mémoire $S(A_n)$ *space*

n étant la taille des données (d'entrée) et A_n l'algorithme A instancié pour un problème de taille n

Classe P - les algorithmes dits "polynomiaux" : on peut déterminer une solution exacte dans un temps borné par une fonction polynomiale de n (solution par une machine de Turing déterministe).

Classe NP - les algorithme pour lesquels on peut établir un temps polynomial que une solution est correcte (solution par une machine Turing nondéterministe).

On a $P \in NP$. On a pas encore prouvé ni $P = NP$, ni $P \neq NP$. Classes à part des problèmes : NP-complets et NP-difficiles.

Concepts - PRAM

Une (machine) **PRAM** se compose de :

- ensemble illimité de processeurs, chaque processeur connait son indice
- une mémoire globale partagée infinie et le temps d'accès à une cellule de mémoire est supposé constant
- un ensemble fini d'instructions qui est le programme commun exécuté de manière synchrone

Concepts - programme

Un programme est une succession ordonnée des pas (étapes, instructions simples). Chaque processeur execute de manière synchrone une même instruction d'une durée unitaire et est capable de determiner la suivante.
Un pas peut être :

- un calcul
- un accès mémoire : lecture ou écriture
- un test
- vide

Concepts - CREW, EREW, CRCW

Les accès mémoires peuvent être supposées :

- concurrents (les processeurs accèdent en même temps à une cellule mémoire - variable)
- exclusifs (les processeurs ne peuvent pas accéder en même temps à une variable).

Concepts - CREW, EREW, CRCW

Trois sous-modèles possibles :

- CRCW
- CREW
- EREW

Le modèle CRCW pose problème pour les écritures : la valeur définitive. Trois possibilités :

- COMMUNE-CRCW : la valeur écrite doit être commune
- ARBITRAIRE-CRCW : on écrit une valeur arbitraire parmi les valeurs des procesesurs
- PRIORITAIRE-CRCW : selon le numéro de priorité (indice) de chaque processeur

Complexité d'un algorithme parallèle :

- nombre de processeurs : $H(A_n)$ *hardware*
- temps parallèle : $T_{//}(A_n)$
- travail : $W(A_n) = H(A_n) * T_{//}(A_n)$ *work*

On mettra en évidence un facteur de type O pour le H et pour le $T_{//}$. Par abus de langage : $O_{//}(.,.)$.

On propose des algorithmes parallèle sur un modèle PRAM et on analyse leur indice de complexité.

Un algorithme parallèle est dit :

- **efficace** si la solution parallèle est "intéressante" : le temps parallèle est polylogarithmique et le travail est un facteur du temps séquentiel multiplié par un facteur logarithmique
- **optimal** si la solution parallèle est efficace et le travail est du même ordre que le temps du meilleur algorithme séquentiel

Exemple : le produit de n valeurs.

Classes de complexité NC

La classe NC (Nick's Class) formalisée par N. Pippenger :

$$NC = \text{ensemble de problèmes } P \text{ tel que}$$
$$\exists A = (A_n) \text{ famille algorithmes :}$$
$$A_n \text{ résout } P_n \text{ en coût } O_{//}(\log^{O(1)} n, n^{O(1)})$$

Sous-classes de NC :

$$NC^k,\ EREW^k,\ CREW^k,\ CRCW^k$$

On a :

$$NC^k = EREW^k \subset CREW^k \subset CRCW^k \subset EREW^{k+1} \subset NC^{k+1}$$

Comment on peut trouver une solution parallèle ?

au départ il faut déjà bien comprendre le problème ; connaitre une solution séquentielle peut aider

Techniques de parallélisation

Techniques pour construire des algorithmes parallèles :

- étude du graphe de précédence
- diviser pour paralléliser
- casser par la redondance

Principe de Brent

Si on conçoit un algorithme pour PRAM pour un nombre important de processeurs, on peut facilement le transformer pour moins de processeurs.

Si on dispose d'une solution sur $H(A_n)$ processeurs, la solution est transformable sur m processeurs avec $m < H(A_n)$.
On répète les opérations sur le même processeur. On peut rajouter des pas vides de calcul, mais ce nombre restera borné par $T_{//}(A_n)$.