ETUDE ET REALISATION D'UN SIMULATEUR PARALLELE POUR LE MODELE PRAM.

François SIEWE, Maurice TCHUENTE
Université de Yaoundé I, BP 812 Yaoundé (CAMEROUN)

ABSTRACT : Parallel simulators appear today as the best way to vulgarize parallel programming environments. They are also useful for parallel programming training as well as development of application programs. We have implemented a parallel simulator for PRAM model. The particularity of our parallel simulator is that it is an implementation of a denotational semantics of PRAM model.

Keywords : PRAM model, denotational semantics, shared memory, GET, PUT

RESUME : Compte tenu des coûts des machines parallèles relativement élevés, les simulateurs parallèles apparaissent aujourd'hui comme le meilleur moyen de vulgariser les environnements de programmation parallèle aussi bien pour l'apprentissage de la programmation parallèle que pour la mise au point des programmes d'application. Un simulateur parallèle est un logiciel qui simule sur une machine ayant une architecture différente, généralement séquentielle, un modèle de programmation parallèle. Il doit offrir à l'utlilisateur le même environnement de programmation que sur la machine réelle (celle dotée d'une architecture adéquate). Nombre de simulateurs parallèles ont été développés pour les modèles parallèles distribués à passage de messages : PSIM pour la programmation des machines ANUPAM[11], C-STOLIC pour la programmation des réseaux systoliques linéaires[10], PVM pour la programmation des réseaux de stations de travail[12]. Dans ce papier, nous proposons un simulateur parallèle, SIPRAM, pour le modèle PRAM (Parallel Random Access Machine) qui est une architecture parallèle à mémoire partagée synchrone.

Le modèle PRAM est le modèle parallèle le plus utilisé en théorie notamment pour l'étude de la complexité des algorithmes parallèles. Il tire son avantage du fait qu'il est facile à programmer (par rapport aux modèles distribués), et aussi du fait qu'il se situe dans la continuité du mode de programmation séquentielle. De nombreux algorithmes parallèles pour des problèmes très variés ont été conçus sur la base de ce modèle. Un simulateur PRAM permettrait justement de mettre au point ces algorithmes et aussi de valider certains résultats théoriques. Contrairement aux autres simulateurs sus-cités, qui sont en général des bibliothèques de procédures ou de fonctions pour les langages C et FORTRAN, notre simulateur a la particularité d'être une implémentation d'une sémantique dénotationnelle du modèle sous-jacent. La sémantique dénotationnelle est une méthode de spécification formelle de la sémantique des langages de programmation. Elle consiste à modéliser l'effet de l'exécution d'un programme par des objets mathématiques, notamment des fonctions. Notre approche consiste à écrire une sémantique dénotationnelle du modèle parallèle et d'impléménter ensuite les fonctions sémantiques ainsi obtenues. On obtient alors un interpréteur capable d'exécuter la structure abstraite des programmes. Cette structure abstraite est générée par un analyseur syntaxique. L'approche par la sémantique garantit la fiabilité du système. Le langage de programmation que nous avons adopté pour SIPRAM est un langage impératif de type PASCAL comprenant en plus, les instructions de lecture (GET) et d'écriture (PUT) en mémoire commune. Pour augmenter l'expressivité du langage, de nouveaux opérateurs venant du langage C, C*, POMPC (?:, >?, <?) ont été ajoutés. En plus, les expressions matricielles sont traitées comme des expressions arithmétiques usuelles par surcharge d'opérateurs : La multiplication de matrices est noté "*" (comme la multiplication usuelle), l'addition de matrice "+", la soustraction "-" et la transposition "'". Par exemple A, B et C étant des matrices nxn, V un vecteur de taille n, les expressions suivantes sont valides : C:=A*B; A[:, 1:2:n] := B[:, 1:2:n], la norme du vecteur V est SQRT(V'*V). SIPRAM tourne actuellement sous DOS et dans bientôt sous UNIX.

Mots clés : modèle PRAM, Sémantique dénotationnelle, mémoire commune, GET, PUT

BIBLIOGRAPHIE

[1] Machael J. QUINN, Désignning efficient algorithm for parallel computers
[2] André SCHIPER, Programmation concurrente, seconde édition
[3] Géné H. GOLUB, Charles F. Van LOAN, MATRIX COMPUTATION, Seconde édition, THE JOHNS HOPKING UNIVERSITY PRESS
[4] Luc BOUGE, Sémantique du parallélisme : un tour d'horizon
[5] LLOYD Allison, A practical intoduction to denotational semantics
[6] CLAUDE KAISER, Ada, pivot d'un atelier d'enseignement des problèmes fondamentaux de la concurence entre processus, T.S.I, vol. 13-n°5 / 1994
[7] Luc BOUGE, Jean-Luc LEVAIRE, control structures for data-parallel SIMD languages : semantics and implementation, FUTURE GÉNÉRATION OF COMPUTER SYSTEM, 1992
[8] Cécile Germain-Renaud, Jean-Paul SANSONNET, Les ordinateurs massivement parallèles, ARMAND COLIN 1991
[9] Peter D. MOSSES, Denotational semantics, Chap. 11, HANDBOOK OF THEORICAL COMPUTER SCIENCE.
[10] Frédéric RAIMBAULT, ETUDE ET REALISATION D'UN ENVIRONNEMENT DE SIMULATION PARALLELE POUR LES ALGORITMES SYSTOLIQUES, Thèse de doctorat, Université de RENNE I
[11] ANUPAM OVERVIEW, African regional workshop on parallel processing and its applications, 31 July-11 August 1995, Yaoundé (Cameroun)
[12] PVM 3 User's Guide and Reference Manual, Septembre 1994