N° d'ordre : 3506

THÈSE

présentée à

L'Université Bordeaux 1

École Doctorale de Mathématiques et Informatique

par Monsieur Samuel THIBAULT

pour obtenir le grade de

Docteur

Spécialité : Informatique

Ordonnancement de processus légers sur architectures multiprocesseurs hiérarchiques : BubbleSched, une approche exploitant la structure du parallélisme des applications

Soutenue le : 6 Décembre 2007

Après avis de :

M.	Jean-François	MÉHAUT	Professeur des Universités	Rapporteur
M.	Thierry	PRIOL	Directeur de Recherche INRIA	Rapporteur

Devant la commission d'examen formée de :

M.	Serge	DULUCQ	Directeur adjoint LaBRI	Président
M.	Panagiotis	HADJIDOUKAS	Université d'Ioannina, Grèce	Examinateur
M.	Jean-François	MÉHAUT	Professeur des Universités	Rapporteur
M.	Raymond	NAMYST	Professeur des Universités	Directeur de thèse
M.	Thierry	PRIOL	Directeur de Recherche INRIA	Rapporteur
M.	François	ROBIN	Directeur délégation ANR "Calcul Intensif"	Examinateur
M.	Pierre-André	WACRENIER	Maître de conférence	Directeur de thèse

Samuel THIBAULT Ordonnancement de processus légers sur architectures multiprocesseurs hiérarchiques : BubbleSched, une approche exploitant la structure du parallélisme des applications 2007

Samuel THIBAULT Ordonnancement de processus légers sur architectures multiprocesseurs hiérarchiques : 2007

Résumé : La tendance des constructeurs pour le calcul scientifique est à l'imbrication de technologies permettant un degré de parallélisme toujours plus fort au sein d'une même machine : architecture NUMA, puces multicœurs, SMT. L'efficacité de l'exécution d'une application parallèle irrégulière sur de telles machines hiérarchiques repose alors sur la qualité de l'ordonnancement des tâches et du placement des données, pour éviter le plus possible les pénalités NUMA et les défauts de cache. Les systèmes d'exploitation actuels, pris au dépourvu car trop généralistes, laissent les concepteurs d'application contraints à « câbler » leurs programmes pour une machine donnée.

Dans cette thèse, pour garantir une certaine portabilité des performances, nous définissons la notion de *bulle* permettant d'exprimer la nature structurée du parallélisme du calcul, et nous modélisons l'architecture de la machine cible par une hiérarchie de listes de tâches. Une interface de programmation et des outils de débogage de haut niveau permettent alors de développer simplement des ordonnanceurs dédiés, efficaces et portables. Différents ordonnanceurs mettant en œuvre des approches variées ont été développés, en partie notamment par des stagiaires encadrés au sein de l'équipe, ce qui montre à la fois la puissance et la simplicité de l'interface. C'est ainsi une véritable plate-forme de développement et d'expérimentation d'ordonnanceurs à bulles qui a été intégrée au sein de la bibliothèque de threads utilisateur MARCEL. Le support OPENMP du compilateur GCC, GOMP, a été étendu pour utiliser cette bibliothèque et exprimer la nature structurée des sections parallèles imbriquées à l'aide de bulles. Avec la couche de compatibilité POSIX de MARCEL, ces supports ont permis de tester les différents ordonnanceurs à bulles développés, sur différentes applications. Les gains obtenus, de l'ordre de 20 à 40%, montrent l'intérêt de notre approche.
Mots-clés : Calcul intensif, parallélisme, supports d'exécution, threads, multiprocesseur, NUMA, multicore

Abstract: The current trend of constructors for scientific computation is towards an imbrication of technologies that permits an ever increased degree of parallelism within a single machine: NUMA architectures, multicore chips, SMT. The efficiency of the execution of an irregular parallel application on such hierarchical machines hence relies on the quality of thread scheduling and data placement, so as to avoid NUMA penalties and cache misses as much as possible. Current Operating Systems fail to achieve this because they are too generic, and thus application developers end up tuning their program for a given machine.

In this thesis, in order to guarantee some portability of performances, we define the notion of *bubble* that allows to express the parallel structure nature of the computation, and we model the architecture of the target machine by a hierarchy of runqueues. A high-level programming interface and several debugging tools then permit to easily develop dedicated, efficient, and portable schedulers. Several schedulers that implement various approaches have been developed, some of which by trainees, which shows both the powerfulness and the simplicity of the interface. To sum it up, this is a real platform for developing and experimenting with bubble schedulers which has been integrated in the user-level thread library MARCEL. The OPENMP support of the GCC compiler, GOMP, was extended to use that library and express the structured nature of nested parallel sections thanks to bubbles. Along with the POSIX compatibility layer of MARCEL, these ports permitted to test the bubble schedulers that were implemented on various applications. The achieved performance benefits, about 20 to 40%, show the interest of our approach.
Keywords: High-Performance Computing, parallelism, runtime systems, threads, multiprocessors, NUMA, multicore