«CROWDTUNING: AUTO-TUNING PRAGMATIQUE ET REPRODUCTIBLE VIA CROWDSOURCING ET ANALYSES PRÉDICTIVES» PAR ABDUL WAHID MEMON

Présentée par : Abdul Wahid Memon Discipline : informatique Laboratoire : LI-PaRAD

Résumé :
Le réglage des heuristiques d'optimisation de compilateur pour de multiples cibles ou implémentations d'une même architecture est devenu complexe. De plus, ce problème est généralement traité de façon ad-hoc et consomme beaucoup de temps sans être nécessairement reproductible. Enfin, des erreurs de choix de paramétrage d'heuristiques sont fréquentes en raison du grand nombre de possibilités d'optimisation et des interactions complexes entre tous les composants matériels et logiciels. La prise en compte de multiples exigences, comme la performance, la consommation d'énergie, la taille de code, la fiabilité et le coût, peut aussi nécessiter la gestion de plusieurs solutions candidates. La compilation itérative avec profil d'exécution (profiling feedback), le réglage automatique (auto tuning) et l'apprentissage automatique ont montré un grand potentiel pour résoudre ces problèmes. Par exemple, nous les avons utilisés avec succès pour concevoir le premier compilateur qui utilise l'apprentissage pour l'optimisation automatique de code. Il s'agit du compilateur Milepost GCC, qui apprend automatiquement les meilleures optimisations pour plusieurs programmes, données et architectures en se basant sur les caractéristiques statiques et dynamiques du programme. Malheureusement, son utilisation en pratique, a été très limitée par le temps d'apprentissage très long et le manque de benchmarks et de données représentatives. De plus, les modèles d'apprentissage «boîte noire» ne pouvaient pas représenter de

façon pertinente les corrélations entre les caractéristiques des programme ou architectures et les meilleures optimisations. Dans cette thèse, nous présentons une nouvelle méthodologie et un nouvel écosystème d'outils(framework) sous la nomination Collective Mind (cM). L'objectif est de permettre à la communauté de partager les différents benchmarks, données d'entrée, compilateurs, outils et autres objets tout en formalisant et facilitant la contribution participative aux boucles d'apprentissage. Une contrainte est la reproductibilité des expérimentations pour l'ensemble des utilisateurs et plateformes. Notre cadre de travailopen-source et notre dépôt (repository) public permettent de rendre le églage automatique et l'apprentissage d'optimisations praticable. De plus, cM permet à la communauté de valider les résultats, les comportements inattendus et les modèles conduisant à de mauvaises prédictions. cM permet aussi de fournir des informations utiles pour l'amélioration et la personnalisation des modules de réglage automatique et d'apprentissage ainsi que pour l'amélioration des modèles de prévision et l'identification des éléments manquants. Notre analyse et évaluation du cadre de travail proposé montre qu'il peut effectivement exposer, isoler et identifier de façon collaborative les principales caractéristiques qui contribuent à la précision de la prédiction du modèle. En même temps, la formalisation du réglage automatique et de l'apprentissage nous permettent d'appliquer en permanence des techniques standards de réduction de complexité. Ceci permet de se contenter d'un ensemble minimal d'optimisations pertinentes ainsi que de benchmarks et de données d'entrée réellement représentatifs. Nous avons publié la plupart des résultats expérimentaux, des benchmarks et des données d'entrée à l'adresse http://c-mind.org tout en validant nos techniques dans le projet EU FP6 Milepost et durant un stage de thèse HiPEAC avec STMicroelectronics.

Abstract :
Tuning general compiler optimization heuristics or optimizing software for rapidly evolving hardware has become intolerably complex, ad-hoc, time consuming and error prone due to enormous number of available design and optimization choices, complex interactions between all software and hardware components, and multiple strict requirements placed on performance, power consumption, size, reliability and cost.Iterative feedback-directed compilation, auto-tuning and machine learning have been showing a high potential to solve above problems. For example, we successfully used them to enable the world's first machine learning based self-tuning compiler, Milepost GCC, which automatically learns the best optimizations across multiple programs, data sets and architectures based on static and dynamic program features. Unfortunately, its practical use was very limited by very long training times and lack of representative benchmarks and data sets. Furthermore, "black box" machine learning models alone could not get full insight into

correlations between features and best optimizations. In this thesis, we present the first to our knowledge methodology and framework, called Collective Mind (cM), to let the community share various benchmarks, data sets, compilers, tools and other artifacts while formalizing and crowdsourcing optimization and learning in reproducible way across many users (platforms). Our open-source framework and public optimization repository helps make auto-tuning and machine learning practical. Furthermore, cM let the community validate optimization results, share unexpected run-time behavior or model mispredictions, provide useful feedback for improvement, customize common auto-tuning and learning modules, improve predictive models and find missing features. Our analysis and evaluation of the proposed framework demonstrates that it can effectively expose, isolate and collaboratively identify the key features that contribute to the model prediction accuracy. At the same time, formalization of auto-tuning and machine learning allows us to continuously apply standard complexity reduction techniques to leave a minimal set of influential optimizations and relevant features as well as truly representative benchmarks and data sets. We released most of the experimental results, benchmarks and data sets at http://c-mind.org while validating our techniques in the EU FP6 MILEPOST project and during HiPEAC internship at STMicroelectronics.

INFORMATIONS COMPLÉMENTAIRES

Denis BARTHOU, Professeur des Universités, à l'Université de Bordeaux/Laboratoire Bordelais de Recherche en Informatique (LaBRI) - UMR 5800 - Talence - Rapporteur
Cédric BASTOUL, Professeur des Universités, à l'Université de Strasbourg /Département Informatique - Strasbourg - Rapporteur
William JALBY, Professeur des Universités, à l'Université de Versailles Saint-Quentin-en-Yvelines/Laboratoire d'Informatique Parallélisme Réseaux Algorithmes Distribués (LI-PaRAD) - Versailles - Directeur de thèse
Pablo Heitor DE OLIVEIRA CASTRO HERRERO, Maître de Conférences, à l' Université de Versailles Saint-Quentin-en-Yvelines/Laboratoire d'Informatique Parallélisme Réseaux Algorithmes Distribués (LI-PaRAD) - Versailles - Examinateur
Grigori FURSIN, Chercheur, au CTuning Foundation - Cachan - Examinateur
Karine HEYDEMANN, Maître de Conférences, à l'Université Pierre et Marie Curie /Laboratoire d'Informatique de Paris 6 (LIP6) - Paris - Examinateur

Contact : dredval service FED : theses@uvsq.fr