Cours de recherche de Master Compilation avancée et optimisation de programmes

Alain Darte et Fabrice Rastello

CNRS et Inria
Laboratoire de l'Informatique du Parallélisme
École normale supérieure de Lyon

Présentation étudiants, 13 Septembre 2007

Qu'est-ce que la compilation ?

La compilation, c'est le chaînon manquant qui relie le "software" au "hardware", le logiciel à l'architecture. C'est extrêmement important pour comprendre ce qu'est un langage, un programme, un ordinateur, un circuit, etc. C'est le coeur de l'informatique et des procédés d'automatisation.

Deux aspects

- Traduction
- Optimisation ☛ le sujet principal de ce cours

Optimisations

- Agressive ou just-in-time
- Front-end ou back-end
- Algorithmique et/ou heuristique

Compsys : du logiciel vers le matériel

Compsys Compilation and Embedded Computing Systems.

But “Adapter et étendre les techniques d'optimisation de la compilation/parallélisation de programmes à la compilation/synthèse pour systèmes de calcul enfouis.”

- Optimisation de code (bas niveau) pour processeurs spécifiques (type DSP, VLIW).
- Transformations de code de haut niveau (au niveau des boucles principalement).
- Conception (compilation) d'accélérateurs matériels par synthèse de haut niveau.
- Développement de logiciels d'optimisation.

Pourquoi ces recherches ?

Pour faire bref :

- La frontière entre processeur (unité programmable) et accélérateur (non programmable) est de plus en plus floue.
- Le besoin industriel est fort dans le monde et en Europe avec Thales, Philips, STMicroelectronics, ...
- La synthèse de haut niveau monte de plus en plus vers le logiciel, et compilation et synthèse se rejoignent.
- Tous les principes d'exécution (parallélisme, pipeline, mémoires distribuées, communications, threads) se retrouvent dans les systèmes embarqués.
- Un rapprochement entre entre les communautés synthèse (informatique et micro-électronique) et compilation (informatique et mathématiques) est nécessaire.

Problématique

Comprendre ce qui peut se faire automatiquement dans le domaine de la compilation (souvent avec des problèmes liés à la mémoire et au parallélisme) :

- formalisation des problèmes (modèle, objectif).
- étude des problèmes (NP-complétude ?, algorithmes).
- étude des modèles (limites, contre-exemples).

Établir des liens entre différents problèmes/théories.

Applications

- Parallélisation automatique (et compilation de HPF).
- Optimisations avancées en compilation “traditionnelle”.
- Compilation de circuits.

Thèmes abordés en cours

Représentations intermédiaires

Control-flow graph (CFG), static single assigment (SSA) et psi-SSA, dominance, loop-forest, etc.

Analyses

Calcul de dépendances, de durée de vie, de flot de données, treillis, points fixes

Transformations

Fusion de boucles, pavage, contraction de tableaux : algorithme de KMW, ILP, graphes, polyèdres, "lattices"

Allocation de registres

Affectation et allocation, vidage en mémoire, "coalescing" : graphes, optimisation combinatoire, graphes chordaux

Ordonnancement

DAG, pipeline logiciel : ILP, approximabilité, retiming

Format et principes d'évaluation

Intervenants Fabrice Rastello (Inria) et Alain Darte (CNRS) + intervention probable de B. Dupont de Dinechin (STMicro).

Format Cours magistraux (avec exercices) pour

- donner quelques bases
- présenter quelques techniques en détails
- introduire quelques problèmes

complétés par les exposés des étudiants.

évaluation Devoir(s) à la maison, attitude en cours, qualité de l'exposé et du rapport.

Network processor Myricom

- Une seule unité pipelinée de profondeur 3 (ou 4 selon la version) : moves, jumps, ALU, load/store.
- Pas de hiérarchie mémoire.
- Peu de registres, avec une sémantique particulière.

Agere Payload Plus

- VLIW de largeur 4.
- 1 ALU par “slice” de 1 octet.
- 32 registres de 4 octets, un pour chaque “slice”, plus registres spéciaux Y et Q :
  - Q = mémoire temporaire, “slice” par “slice”
  - Y = écrit “slice” par “slice”, lu par tout le monde. Seul moyen de communication !
  - Code à . . . 2 adresses (sauf pour Q et Y) !

Ubicom IP3023

- Processeur scalaire “in order”.
- 8 contextes de registres pour support “multi-thread”.
- Possibilité de mélanger les instructions de différents “threads” cycle par cycle.
- 256Ko I-scratchpad + 64Ko D-scratchpad.

ST220 family

- Processeur VLIW (largeur 4).
- Direct-mapped I-Cache (512 lignes de 64o).
- Prédication limitée (select = move conditionnel).
- Multiply-add, auto-incréments.

Registres étranges

- Registres d'adressage avec incréments et décréments. Ex : TI TMS320C25, Motorola DSP56K
- Registres rotatifs. Ex : Itanium
- "Clusters" de registres. Ex : Agere Payload
- "Register windows". Ex : WMIS Microcontroller
- Registres pour load/store doubles. Ex : IBM PowerPC405, Intel StrongARM pour IXP-1200

Support matériel pour l'exécution

- Cache d'instructions
- Scratchpad pour les instructions et les données
- Support pour l'exécution des boucles
- Compression de code. Ex : Atmel Diopsis Dual Core DSP
- Processeurs extensibles (custom functional units). Ex : Xtensa (Tensilica)
- Puissance : drowsy modes, ...