Les techniques de pipeline logiciel

Christine.Eisenbeis@inria.fr

janvier 2002

Parallélisme d'instructions, un peu d'histoire

- Problème original = microprogrammation, réduction de la taille du code
- –> Compaction
- Problème très bien résolu dans le cas de blocs de base (suite d'instructions sans branchements vers l'extérieur ou vers l'intérieur). **Compaction locale**.
- **Peu de parallélisme (facteur de 2 ou 3)**

Limitation aux blocs de base trop stricte?

[Riseman, Foster, 1972] Dédoubler (virtuellement) la machine à chaque branchement conditionnel

- **parallélisme d'environ 50**

[Nicolau, Fisher, 1984] Exécution d'un programme, puis compaction de la trace

- **parallélisme de 3 à 1000!**

Mais ... supposition d'une infinité de ressources ...

Nécessité d'un algorithme de **compaction globale**

Premiers pas

[Tokoro et al. 1981]

- Compaction locale, puis transfert aux frontières des blocs de base, selon une série de règles.

[Fisher, 1981]

- Ordonnancement des traces (*trace scheduling*). Compilateur Bulldog (thèse Ellis, 1985). Machine Multiflow.
  - extraction d'une trace (chemin le plus sûrement exécuté)
  - compaction de la trace
  - correction du programme

Puis ...

[Nicolau, 1985]

- Ordonnancement par percolation (*percolation scheduling*)
  - Mouvements élémentaires
  - Corrections

[Hwu et al., 1992]

- Formation de superblocs
- Instructions à prédicats (élimination des branchements conditionnels)
- Formation d'hyperblocs

Et les boucles?

Techniques suivant les principes précédents

[Tokoro]

- transfert d'opérations d'itération en itération

[Fisher]

- formation de trace = déroulage de boucles

[Rau, 80]

- prise en compte des contraintes de recouvrement des itérations, ordonnancement modulo (*modulo scheduling*). Essai successifs de II sur un ordonanncement local.

[Kogge, 81]

- enchaînement des tables de réservation hardware

Puis très vite ...

[Touzeau 84, **Lam 87**, Eisenbeis 86] Pipeline logiciel, *software pipelining*

[Aiken, Nicolau, 88]

- *Perfect pipelining* , dérouler le code jusqu'à trouver un motif répétitif

[Hanen, 86; Allan, 90]

- Modélisation par réseaux de Petri

[Rau, 94]

- Iterative modulo scheduling

Et ...

[Gasperoni, Schwiegelshohn, 91]

- première méthode directe

[Wang, Eisenbeis, 93]

- généralisation de GS, pipeline logiciel décomposé

[Leiserson, Saxe, 90; Huard et al. 01]

- *retiming*

Branchements dans les boucles

[Lam, 87]

- ordonnancement hiérarchique (fusion des deux branches)

[Ebcioglu, Nakatani, 90]

- approche à la Tokoro, transfert des instructions par la flèche de retour

[Warter et al., 92]

- if-conversion virtuelle (conversion des branchements en code à prédicats, ordonnancement, puis retour à un code sans prédicats).

Allocation en registres

[Lam, 87]

- expansion modulo des variables (MVE)

[Eisenbeis et al., 88]

- MVE + minimisation du degré de déroulage

[Rau, 92]

- procédé hardware pour le pipeline, décalage automatique, rotating register file

Allocation en registres

[Hendren et al., 92]

- coloriage de graphes d'intervalles circulaires + recopie de registres

[Lelait et al., 95]

- Meeting graph

Mise en oeuvre ...

- Alpha
- SGI (Pro64 compiler)
- Itanium (rotating register file)
- processeurs embarqués

Aujourd'hui

Vers des approches exactes.

Temps de compilation plus grand, compilation itérative, approches exactes par programme linéaire en nombres entiers (voir par exemple Sawaya, 97)