RISCプロセッサ向け命令スケジューリング *

5E−7

林 正和　堀田耕一郎　五十嵐寛　松山学　滝内政昭 †

富士通株式会社‡

1 はじめに

近年のRISCプロセッサによるシステムの高性能化には著しいものがある．これらの性能向上においては，最適化コンパイラの役割りが非常に大きい([1])．

特に，命令スケジューリング技術は，演算パイプラインのストールを避け，演算器を有効に利用することによって，各アーキテクチャの資源を無駄なく利用するという技術であり，それぞれのアーキテクチャの性能を引き出すためには，必須の技術である．

本稿では，コンパイラに命令スケジューリングを実装する場合の，必要な要素技術を考察し，その効果及び問題点を報告する．

2 コンパイラの構成と基本要件

ここでは，命令スケジューリング処理を持ったコンパイラのフェーズ構成及び関連する最適化について述べる．

2.1 コンパイラの構成

命令スケジューリング実現のアプローチとしては，コンパイラの『外付けツール』として，コンパイラが出力するアセンブラを入力する方式と，[1]で述べたように，コンパイラの一つのフェーズとする方法が考えられる．前者は，既存のコンパイラと容易に結合できるため，適用の範囲は大きいというメリットがあるが，性能を追求するには，限界がある．高性能なオブジェクトを得るためには，コンパイラの各フェーズで収集した様々な情報（例えば，データフロー情報など）を利用できる後者の構造の方が良いと考える．

また，命令スケジューリングをする場合には，中間テキストと命令が1:1に対応し，さらに，それらの中間テキストは既に，最適化されている必要がある．これらの要件から命令スケジューリングフェーズは，[1]で示したような命令出力の直前で実施する構造にするのが良い．

2.2 最適化

コンパイラで実施する各種最適化において，ループアンローリングは，スケジューリング範囲内の命令の数を増やすことによって，命令の並列実行の可能性を広げることができるため，命令スケジューリングに対して，非常に有効な最適化の一つと言える．また，並列性を得るためには，仮想レジスタ(テンポラリ)のリネーミングを行ない，命令間の不要なデータ依存関係を取り除く必要がある．

3 命令スケジューリング処理

ここでは，命令スケジューリング処理において必要な技術や実現手法について述べる．

3.1 データ依存解析

命令スケジューリング処理の時には，データ依存解析は必須の技術である．一般に，二つのテンポラリ間の定義・参照によるデータ依存関係は，解析が容易であるが，メモリデータ間の依存関係の解析は難しい．これをどこまで解析できるかは，二つのメモリデータの実メモリ上での領域が重なるかどうかを見極めるMemory Aliasingの能力に依存している．図1はループアンローリングした結果の中間テキストである．ここで，STORE命令(a)がアクセスする領域と，LOAD命令(b)がアクセスする領域が重ならないということが分からないと，(b)は(a)を追い越して動かすことができない．その結果，ループアンローリング等で命令の並列度を増やしても，命令スケジューリングができなくなる．

3.2 アーキテクチャ変更への対応

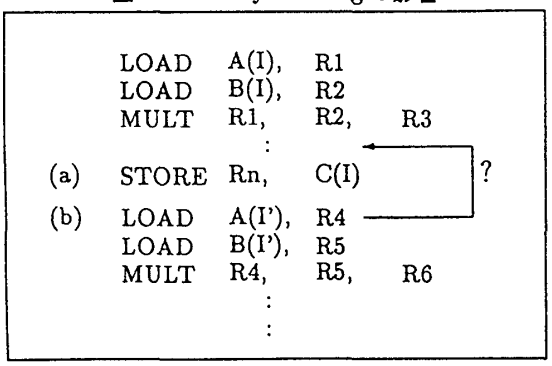
命令スケジューリングフェーズは，アーキテクチャの仕様変更への対応や複数アーキテクチャ展開を容易にする構造が望ましい．そのため，命令スケジューリングフェーズは，ターゲット依存部分と非依存部分に分け，かつ依存部分は，ターゲットマシンに関する情報や命令に関する情報がテーブルデータとして表現されるように工夫するべきである．これらのテーブルを修正し，細かいチューニングを行なうことによって，アーキテクチャの変更・改善に容易に対処

---

*Instruction Scheduling for RISC Processors

†Masakazu HAYASHI,KohIchiro HOTTA,
Yutaka IGARASHI,Manabu MATSUYAMA,
Masaaki TAKIUCHI
‡FUJITSU LIMITED

図 1: Memory Aliasing の影響



できる.

ここで, アーキテクチャ非依存部分とは, DAG (Directed Acyclic Graph) を作成する部分, ある規則に従って, 中間テキストを並び変える部分, 命令出力部とのインタフェース等である.

3.3 命令スケジューリングの範囲

通常のプログラムでは, 約 20% の割合で, 分岐命令が存在する ([4]). このため, 基本ブロック内での命令スケジューリング技術だけでは, 同時に数命令実行できるような並列 RISC プロセッサが必要とする並列性を得ることができない. 従って, 命令スケジューリングの範囲として, 分岐を越えて命令を移動するという広域スケジューリング技術 [2, 3] をサポートすると大きな効果が得られると考えられる.

また, 広域スケジューリングを有効に適用するためには, コンパイラによる分岐予測技術が重要な技術となる. 特に, 分岐が多い非科学技術系プログラムでは, 分岐予測の正確さが, 性能に大きく影響すると考えられる. 分岐予測をより正確にするためには, プロファイル情報が有効である. 従って, プロファイル情報を取り込めるしくみを備えることが, 広域スケジューリングコンパイラの要件となる.

4 効果

ここまでの要件に基づいて, 命令スケジューリング機能をサポートしたコンパイラを開発した. ただし, 現状では, スケジューリングの範囲は, 基本ブロック内に限定してある.

表 1 にスケジューリングの効果をスケジューリング抑止時との比で示す. 測定対象プログラムは, Dhrystone や compress など比較的分岐が多い非科学技術計算系プログラム (8 本) と, 行列積, 連立一時方程式の解法など, 比較的ループと配列演算が多い科学技術計算系プログラム (6 本) から選んだ. また, 測定マシンは, SPARC Station 2(SS-2) である.

表 1 から, 基本ブロック内の命令スケジューリングだけでも, ある程度の性能向上が見られることが分かる. また, 科学技術系プログラムのほうが, 非科学技術系プログラムよりも, 命令スケジューリングの効果が大きい. これは, 科学技術計算系プログラムの方が, ループアンローリングの効果等で, スケジューリング範囲内の命令の並列性が, 高くなるためと考えられる.

表 1: 命令スケジューリングの効果

	プログラム	
マシン	非科学技術系	科学技術系
SS-2	1.03 ～ 1.07	1.08 ～ 1.25

非スケジューリング時 1.0

5 まとめ

RISC プロセッサの命令スケジューリングをコンパイラに実装する場合のコンパイラやスケジューリング部に必要な機能・技術に関して考察した. さらに, それに基づいたコンパイラを開発し, 基本ブロック内での命令スケジューリングにおける効果を挙げた. 今後の主流である並列 RISC プロセッサにおいては, 広域スケジューリングをサポートすることが, 高性能化のためには重要であると考えている.

今後は, 実際に広域スケジューリングをサポートし, 本稿で述べた事の確認や非科学技術系プログラムにおけるスケジューリングの効果・改善を検討していく所存である.

参考文献

[1] 堀田 他, 「*RISC プロセッサ向け最適化コンパイラの構成要件*」, 情報処理学会第 46 回全国大会, 1993

[2] Alexander Aiken, Alexandru Nicolau, *'A Development Environment for Horizontal Microcode'* IEEE Trans. on Software Engineering, May 1988

[3] John R. Ellis, *'Bulldog: A Compiler for VLIW Architectures'* The MIT Press, 1986

[4] John L. Hennessy, David A. Patterson, *'Computer Architecture A Quantitative Approach'*, Morgan Kaufmann Publishers ,INC. 1990