4F-2 明示的な分岐制御の可能性の検討および制御機構の提案

中村 友洋, 吉瀬 謙二, 辻 秀典, 田中 英彦
東京大学工学部

1 はじめに

命令レベルの並列性を利用するにあたって、分岐制御が重要となる。既存のプロセッサにおいても、多くのものがハードウェアによる分岐予測機構を設けて、分岐命令によるオーバーヘッドの削減を行なっている。ところが、満足できる性能まで達していないにも関わらず、その大きな向上は望めなくなっている。その 1 つの原因は、あくまでも分岐方向を過去の履歴から予測するという、根本的な制御方式によるところが大きい。予測であるためにある確率で予測が外れるからである。そこで本研究では、分岐制御を明示的に行なうことを目指し、その可能性を調査・検討した。また、このような制御を行なうための機構モデルについても提案する。

2 目的と背景

現在のマイクロプロセッサは、スーパースカラ構造や VLIW 構造をとり、命令レベルの並列性を利用し、高性能化を目指してきた。しかしこのような構造が抱える問題の 1 つに分岐処理の問題がある。現在多くのマイクロプロセッサは数段のパイプラインを持ち、さらに複数命令を同時に発行・実行するため、分岐命令の出現頻度の高さとも相俟って、性能的にクリティカルな問題となっている。

現在までに分岐命令を効率的に処理するための機構として分岐履歴情報を用いたものなどがさまざま提案・実装されているが、未だ完全な分岐予測機構はなく、より効率的な分岐制御機構の検討が課題となっている。表 1 は、主な分岐予測機構の分岐予測成功率の比較である。この表から、静的な分岐予測機構では 65〜70%程度、動的な分岐予測機構でも 80〜90%程度の予測成功率しか得られていないことが分かる。

分岐予測戦略	予測成功率
Branch Always	70.1%
Branch Backward	64.5%
Pentium 2-bit	82.0%
PowerPC604	82.5%
2-Level Adaptive	91.3%

表 1: 分岐予測戦略の予測成功率

命令レベルの並列性をより多く動的に抽出するには、命令ウィンドウのサイズを大きくする必要があるが、そのためには分岐をまたがる命令の解析が必須である。なぜならば、ベーシック・ブロックのサイズは一般的な整数系アプリケーションでは 4〜8 命令程度と小さいためである。つまり、数十命令を解析して命令レベルの並列性を抽出するには、数個〜十数個の分岐予測を正しくおこなう必要がある。ところが、現在の分岐予測機構の予測成功率には、このような要求に十分応えられるだけの性能があるとはいい難い。

3 分岐処理コスト

分岐処理のコストとしては主に次の 2 つが挙げられる。

制御コスト 分岐予測等の分岐制御のコスト

修正コスト 分岐予測等の失敗による修正のコスト

制御コストとしては、例えば BHT(Branch History Table) をひいて分岐予測をするときにかかるコストがある。BHT をひく処理は分岐命令のデコード時におこなわれ、そこで分岐予測が修正された場合には、すでに分岐不成立と予測して次命令をフェッチしているので少なくとも 1 命令分は無駄な処理を行なったことになる。

修正コストは BHT のような分岐予測機構の予測結果が間違っていたときに課せられるペナルティである。最近のマイクロプロセッサでは、これは非常にコストの高い処理になる。これらのプロセッサの一部に装備されている re-order バッファなどの複数の命令を一時的に保存しておくバッファには、分岐予測に基づいてフェッチされた命令が入っており、これらをフラッシュして新しいパスへのフェッチ・ポイントを計算して移動する必要がある。さらにバッファをフラッシュしたために次の命令列の解析を行ない命令を実行ユニットに発行するまでは実行ユニットのパイプラインは処理がない状態 (ストール状態) となる。よってバッファ・サイズの大きさなどによるが大きなペナルティを課せられることがある。

分岐予測成功率の低下を招かずに制御コストを現状から削減するのは難しい。しかし修正コストにはまだコスト削減の可能性が残されている。

4 明示的な分岐制御機構

修正コストの削減方法の 1 つの可能性として、明示的な分岐制御機構を検討する。

4.1 分岐条件決定タイミングの最適化

分岐予測ミスの修正は、分岐ミスを検出し修正処理を行なうまでの時間を短縮することで削減が可能である。そのためには、分岐条件を決定する命令 (XXXcc 命令と呼ぶ) の実行時点で分岐予測ミスの修正処理を開始することで、この時間を短縮できる。しかしこの効果を十分に発揮するためには、分岐命令に対して、XXXcc 命令を出来る限り早く実行することが重要である。

A Proposal of Mechanism for Explicit Branch Control
Tomohiro NAKAMURA, Kenji KISE, Hidenori TSUJI, Hidehiko TANAKA,
University of Tokyo, Department of Engineering

図 1は、サンプル・プログラムとして compress (SPECint92) について、各ベーシック・ブロック内でデータフロー解析を行なって XXXcc 命令を分岐命令からできるかぎり離すようにスケジューリングした場合に、XXXcc 命令と分岐命令との間にとることのできる命令数である。ただしここでのスケジューリングはベーシック・ブロック内という厳しい制約の元でのものである。実際にはベーシック・ブロックにまたがるスケジューリングを行なってより間隔をあけることができる場合もある。ここでの値は最も厳しい制約のもとでのものであり、データフロー解析を行なっているので、安全なスケジューリングとなっている。

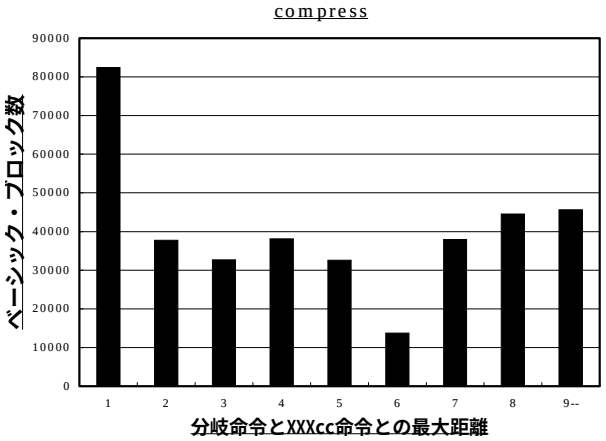

図 1: XXXcc 命令の最適化スケジューリング

4.2 新しい分岐制御ハードウェアの提案

前節で述べた、XXXcc 命令のスケジューリングに関する最適化を行なっても、現在のハードウェアでは修正コストの削減は望めない。そこで本節では、XXXcc 命令の実行時に分岐予測の修正処理を行なうハードウェア機構 (CCC 機構: Condition Code Cache) の提案を行なう。

CCC の実装に必要とされる項目は次の通りである。

- XXXcc 命令のアドレスからエントリをひけること
- XXXcc 命令の結果から次の分岐命令のアドレスとその分岐方向を決定できること

このような要件を満たすモデルとして図 2を考える。図 2で、PC(XXXcc) は XXXcc 命令のプログラムカウンタ、PC(Bicc) は PC(XXXcc) フィールドで指定される XXXcc 命令によって分岐条件が決定される分岐命令のプログラム・カウンタを意味する。cond. フィールドは条件分岐命令中の cond. (condition) フィールドの内容である。この CCC 機構により分岐条件決定時に修正処理を開始することができ、XXXcc 命令を分岐命令から離すことにより、修正コストの削減を実現できる。

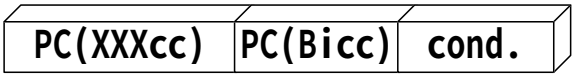

図 2: CCC(condition code cache) の実装モデル

5 性能評価

XXXcc 命令の最適スケジューリングおよび CCC 機構の実装による性能評価を簡単に述べる。compress の場合、分岐処理にかかるコストは表 2のようになった。CCC 機構により削減できる分岐処理コストは、修正コストのみであり、表 2では修正コストの約半分を CCC 機構により削減できたことが分かる。なお、削減できない修正コストとは、XXXcc 命令を分岐命令に比べてあまり早い時期に実行出来なかったことによるもので、XXXcc 命令のスケジューリングに関してより多くの自由度を与えることでさらに削減が可能であると思われる。

制御コスト	削減できない修正コスト	削減された修正コスト
32.31%	32.48%	35.21%

表 2: compress の分岐処理コスト比率

各種プログラムに対して、このようなコスト削減の効果を分岐予測成功率に換算した値を図 3に示す。この換算は、CCC 機構により得られたコスト削減効果を、CCC 機構をもたない分岐予測機構で同様のコスト削減を実現しようとしたときに必要となる分岐予測成功率を示したものである。最高で、96%以上の分岐予測成功率を達成したことと同等の性能を示している。

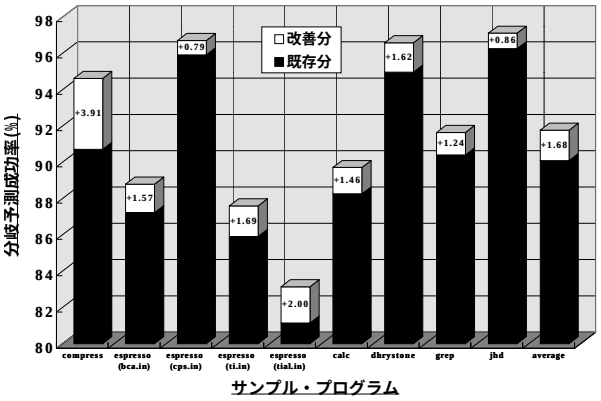

図 3: CCC 機構による修正コスト削減効果の分岐予測成功率への換算

6 おわりに

本研究では、明示的な分岐制御機構の可能性を調査し、それを実現する方式として、XXXcc 命令の最適スケジューリングおよび CCC 機構の提案を行なった。また性能評価によって分岐予測成功率換算で 1〜4%程度の性能向上を確認した。90%程度の予測成功率にさらに 1〜4%程度の成功率向上は決して小さい値ではない。今後はベーシック・ブロックをまたがった命令スケジューリングを考え、より CCC 機構が有効に働くスケジューリング方式の提案などを検討したい。

謝辞

本研究の一部は文部省科学研究費 (一般研究 (B) 課題番号 07458052「大規模データパスプロセッサの研究」) による。

参考文献

[1] 中村友洋, 吉瀬謙二, 金指和幸, 田中英彦, “トレース・ドリブン・シミュレーションによる分岐予測機構の検討”, 第 51 回情報処理学会全国大会, pp.6-13–6-14, 1995