複数パス実行時におけるキャッシュトラフィックの傾向分析

入江英嗣[†]，斎藤英一[‡]，吉瀬謙二[†]，坂井修一[†]，田中英彦[†]
東京大学大学院工学系研究科[†]　株式会社日立製作所 日立工業専門学院[‡]

1　はじめに

現在のプロセッサ研究の傾向の一つとして並列性の抽出がある。複数の実行ユニットを効率よく利用するために、より多くの並列性の抽出を目指した研究が数多く行われている。このような、大規模並列実行を目指したプロセッサが、大量の命令を同時に実行するためには、必要とされるデータ供給が記憶階層から行われなければならない。しかし、現在のキャッシュシステムを考えた場合、単純にアドレス計算ユニットやポート数、容量等を増やすだけでは、実際の性能向上につながらず、設計に様々なトレードオフが存在する。本稿では、大量の命令を同時に実行するプロセッサのモデルとして、複数パス実行に注目し、複数パス実行においてキャッシュに要求される性能について考える。

2　複数パス実行

大量の命令をフェッチするためには分岐命令をまたいだ分岐が必要となるが、分岐命令をまたいでフェッチするためには、分岐命令の結果を予測しなければならない。分岐予測ミス時のペナルティを緩和するため、複数パス実行方式が提案されている。複数パス実行では、分岐の成立、不成立両方のパスを投機的に実行し、分岐命令の実行結果をうけて、不要となったパスをパイプラインからフラッシュする。パスの展開の戦略について、全てのパスを展開する Eager Execution、分岐予測の確信度が低いときにパスを展開する Selective Eager Execution 等がある。

複数パス実行の採用により、シングルパス実行時よりも、さらに 1 サイクルあたりのキャッシュトラフィックが増加することが予想される。

- 複数パス実行により、IPC が向上し、1 サイクルあたりの処理量が増える
- 無効なパスによるアクセスが加わる

等の影響があるためである。

3　1 サイクルあたりのキャッシュト ラフィック

単一パス実行プロセッサであれば、1 サイクルあたりに処理しなければならないロード/ストア命令数の平均は、平均 IPC に全実行命令中のロード/ストア命令の割合をかけることで概算することができる。

しかし、複数パス実行時では、フラッシュされたパスに含まれる命令の影響が単一パス実行時に比べて大きくなり、キャッシュトラフィックの見積もりは単純に概算できない。シミュレータを用いて、複数パス実行時の、1 サイクルあたりのロード/ストア命令数を計測した。SPEC95 の li を用いた。

複数パスの実行のパラメタをいくつか用意して、結果を比較した。グラフの左から一つ目は、パスを展開せず、分岐予測にしたがって単一パスのフェッチを行う [図 1(a)]。二番目から四番目までは Selective Eager Execution モデルで、それぞれ

- 最大 2 本のパスを同時に実行する。実行確率が高いと判断した主パスを最大 32 命令までフェッチし、副パスを最大 16 命令をフェッチする [図 1(b)]
- 最大で 9 本のパスを同時に実行する。主パスを 16 命令フェッチし、4 命令までフェッチする副パスを最大で 8 本まで展開する [図 1(c)]
- 主パスを 16 命令フェッチし、8 命令までフェッチする副パスを最大で 4 本まで展開する [図 1(d)]

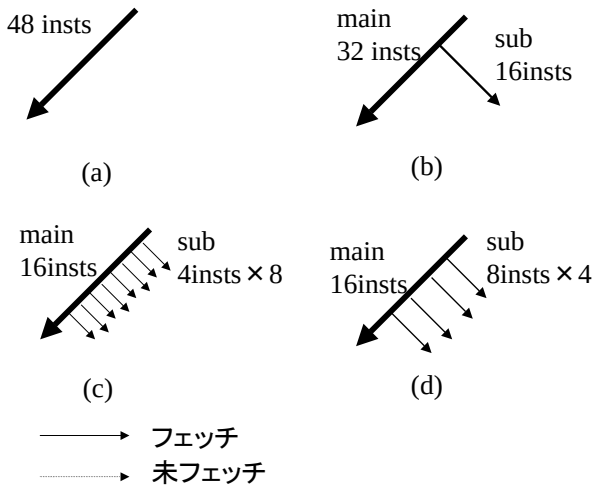

図 1: パス展開のパラメタ

(a)〜(d) について毎サイクルフェッチする命令の上限は 48 である。また、1024 エントリの命令ウィンドウを持っている。

結果 (図 2) より、複数パス実行によって性能向上が見込まれることが分かった。このパラメタのときは毎サイクル平均 5〜7 回のキャッシュアクセス要求がでているが、予想したように、実行性能の向上の割合に対して、

Cache Traffic Analysis in Multipath-Execution processor
Hidetsugu IRIE, Eiichi SAITO, Kenji KISE,
Shuichi SAKAI, and Hidehiko TANAKA
[†] Graduate school of Engineering, The University of Tokyo
[‡] Hitachi, Ltd.

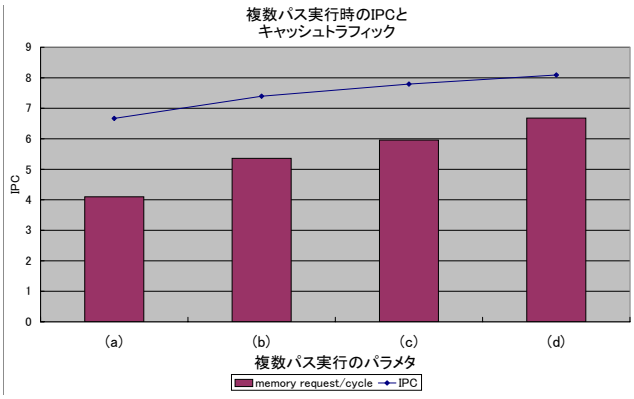

図 2: 複数パス実行時の実行性能とキャッシュトラフィック

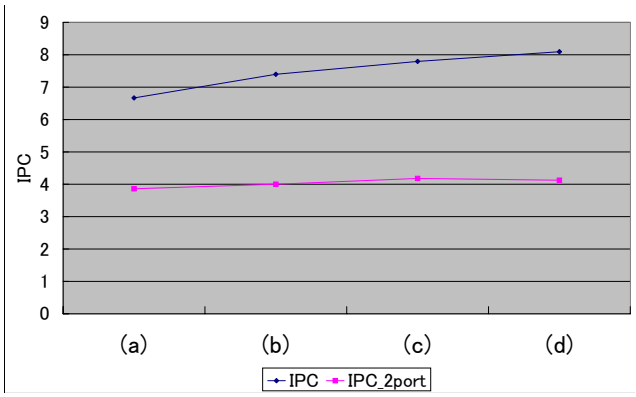

図 3: キャッシュのポート数の制限による性能低下

キャッシュトラフィックの増加の割合の方が高い。単一パス実行に比べて 21 % IPC が向上している右端のデータについては、キャッシュトラフィックは単一パス実行に比べて 62 %増加している。

4 ポート数による制限

前節の結果から、複数パス実行時には毎サイクル大量のキャッシュアクセス要求が出されることが予想される。しかし、キャッシュにとって動作速度は重要な指標であり、単純にポート数、容量、連想度等を増やしたり、なにか特別な操作を追加したりすることは、かえってシステム全体の性能を落としてしまうことに繋がる恐れがあり、キャッシュの複数ポート化には注意深い設計を必要とする。現在行われている手法として、

- キャッシュを倍速で動かす
- コヒーレンシを保った複製キャッシュを用いる
- キャッシュをバンク分けして、異なるバンクへのアクセスを同時に行う

等の方法があるが、どの方法もスケーラビリティに欠け、バンクを用いた手法においても、1 サイクルに 4 データ程度の供給が限界である [1]。

キャッシュへの 1 サイクルあたりのアクセス回数が制限されたとき、性能がどれだけ低下するかを前節と同じパラメタを用いて調べた。

キャッシュへのアクセスを 1 サイクルに 2 度までと制限した場合の性能を、図 3に示す。

複数パス実行による性能向上がほとんど得られていないことが分かる。一番性能が上がっているケースについても、単一パス実行からの性能向上は 8.3 %であり、キャッシュアクセスに制限がない場合の 54 %の性能に留まっている。

5 複数データ供給の手法

キャッシュから 1 サイクルの間に供給できるデータ数の限界が、実行性能に深刻な影響を与えることを前節で述べた。前節で述べたキャッシュの複数ポート化手法も併用して、複数データ供給のための有効な手法が必要とされている。

キャッシュのデータ供給能力を上げるために、同時に必要とされる複数のデータを一塊にして、一度に読み出す、という方法が考えられる。この方法の実現のためには、”同時に必要とされる ”と思われるデータの組合せをどう見つけ出すか、という問題と、そのように組み合わされたデータにどのように高速にアクセスするか、という問題の二点を解決しなくてはならない。

この手法の簡単な実装例として、access combining[2] がある。access combining では、読み出してきたキャッシュラインのデータ全てをロード要求と比較し、該当するデータを全て読み出す。シングルパス実行時のトレースを用いた単純な解析では、1 サイクルあたりのデータ供給量が 10 %〜30 程度向上することが認められた。複数パス実行環境ではさらに有効な手法となる可能性がある。

6 まとめ

複数パス実行シミュレータを用いて、複数パス実行時のキャッシュトラフィックがどの程度の値になるのかを調べ、複数パス実行によって実行性能を向上させるほど、キャッシュトラフィックの増加が大きいことを示した。

また、キャッシュのポート数の制限が、性能向上に深刻な影響を与えることを示した。

現在行われているキャッシュの複数ポート化手法では限界があり、スケーラビリティのある複数ポート化手法や複数データ供給手法が必要とされている。

今回は li　30000000 命令について評価を行った。今後シミュレーションの規模を拡大して評価する。

参考文献

[1] Jude A. Rivers, Gary S. Tyson, Edward S. Davidson, and Todd M. Austin. On High-Bandwidth Data Cache Design for Multi-Issue Processors. In Proc. of MICRO-30, pp. 46-56, 1997.

[2] K. M. Wilson, K. Olukotun and M. Rosenblum. Increasing Cache Port Efficiency for Dynamic Superscalar Microprocessors In Proc. of ISCA-23, pp.147-157, May 1996.