計算機アーキテクチャ 131－9
（1998. 9. 22）

分岐先アドレスの性質を利用した 2レベル表による分岐先バッファの容量削減

山田 祐司† 小林 良太郎†
安藤 秀樹† 島田 俊夫†

マイクロプロセッサの分岐予測機構には高い精度が望まれている。分岐先アドレスはBTB(Branch Target Buffer:分岐先バッファ)を用いて予測する手法が一般的だが、高い予測成功率を得るためにはBTBには多くのエントリ数が必要となりハードウェア量が大きくなるという問題がある。

本稿では分岐先アドレスの性質を利用してBTBのハードウェア量を削減する手法として2レベル表方式を提案する。評価の結果、2レベル表方式は従来のBTB方式に対して分岐先アドレス予測成功率をほとんど低下させることなく分岐先アドレス部のハードウェア量を約52%削減することができた。また、同程度のハードウェア量では分岐先アドレス予測成功率を約1.07%高めることができた。

Reducing the Amount of Branch Target Buffers with a 2-Level Table Scheme, Utilizing the Characteristics of Branch Target Addresses

YUJI YAMADA,† RYOTARO KOBAYASHI,† HIDEKI ANDO† and TOSHIO SHIMADA†

Accurate branch prediction is required in microprocessors. Branch target addresses are generally predicted with a BTB(Branch Target Buffer). To achieve high prediction accuracy, BTBs require many entries, thus considerably increasing the amount of hardware.

This paper proposes a new scheme called a *2-level table scheme* to reduce the amount of hardware of BTB, utilizing the characteristics of branch target addresses. Our 2-level table scheme reduces the amount of the brannch target address part of BTB by approximately 52% with little reduction of branch target address prediction accuracy. our scheme improves branch target address prediction accuracy by approximately 1.07% with the almost same amount of hardware.

1. はじめに

マイクロプロセッサの実行において分岐命令による制御流の乱れは性能低下を引き起こす。これを回避するために分岐予測を用いて投機的実行が行われるが、プロセッサの高性能化技術としてスーパースカラ方式の採用やパイプライン段数の増加傾向があり、分岐予測ミスペナルティはますます大きくなってきている。従って、分岐予測機構にはより高い精度が望まれている。

分岐予測は、分岐方向と分岐先アドレスの二つの予測値を出力する。分岐方向の予測手法には様々な機構(例えばgshare機構[11])があるが、分岐先アドレスの予測手法には**BTB**(Branch Target Buffer:分岐先バッファ)を用いるのが一般的である[10]。BTBとは分岐先アドレスを保持するキャッシュで、分岐命令アドレスをインデクスとして当該命令の分岐先アドレスを出力する。

分岐先アドレスの予測成功率を高めるためにはエントリにおける分岐命令間の競合を避けることが必要である。このためBTBには多くのエントリ数が必要となりハードウェア量が大きくなるという問題がある。従って、予測成功率を低下させずにBTBのハードウェア量を削減するためにはエントリあたりのハードウェア量を削減することが必要である。BTBのエントリはタグ部と分岐先アドレス部とからなるのでこれらのビット数を削減しなければならない。タグ部のハードウェア量を削減するための手法はいくつか提案されている[3][4][12]。しかし、分岐先アドレス部のハードウェア量を削減するための研究は多くはなされておらず満足な方法は見出されていない。

本研究の目的は、分岐先アドレス予測成功率をほとんど低下させることなくBTBのハードウェア量を削減することである。本稿では分岐先アドレスの性質を利用してBTBの分岐先アドレス部のハードウェア量を削減する手法として**2レベル表方式**を提案する。

以下、2章では関連研究について述べる。3章では分岐命令アドレスと分岐先アドレスとの関係を調査し、我々のハードウェア量削減のための提案手法について述べる。4章では提案手法の評価と検討を行う。5章で本稿をまとめる。

2. 関連研究

2.1 分岐先アドレス予測成功率を高める研究

分岐先アドレス予測成功率を高めるために、分岐命令の種類に応じた予測機構を組み合わせる手法が研究されている[1][5]。従来のBTB以外の分岐先アドレス予測手法について紹介する。

サブルーチン・コールとリターンの対に着目した機構としてリターン・スタック[9]がある。これは、ハードウェアとしてスタックを用意してそこにリターン・アドレスを記憶し、リターン命令実行の際に用いるものである。

間接分岐命令ではその都度分岐先アドレスが変わる。間接分岐先アドレスをより正確に予測する手法として、分岐命令アド

† 名古屋大学大学院工学研究科
Graduate School of Engineering, Nagoya University

レスと分岐履歴とを組み合わせたものを BTB へのインデクスとする手法が提案されている。履歴には、「分岐方向の履歴」を用いる方法[2]や、これまでの実行パスの分岐命令アドレスの部分ビット列を組み合わせた「パス履歴」を用いる方法[8]が提案されている。

2.2 BTB のハードウェア量を削減させる研究

BTB のエントリはタグ部と分岐先アドレス部とからなる。各部分のハードウェア量削減に関する研究について紹介する。

2.2.1 タグ部の削減

BTB におけるエントリあたりのタグ部のビット数と分岐先アドレス予測成功率とのトレードオフの調査がなされている[3][4]。エントリ数 512、ダイレクト・マップ方式の場合において、SPECint92 において、タグ・ビット数が 2 ビットの場合でもタグ・ビットを全て保持する場合の精度の 99.9%を得ることができると報告している。

CAT:Caching Address Tags と呼ばれる機構[12]では、キャッシュのタグ部の情報と別に用意したタグ・キャッシュとの情報を組み合わせてタグ値を生成する。これによってエントリに保持するタグ部のビット数を削減させる。

2.2.2 分岐先アドレス部の削減

従来の BTB では分岐先アドレスをそのまま保持する。これに対して、**変位保持方式**[6]では分岐命令アドレスと分岐先アドレスとの差分 (変位) を保持する。分岐先アドレスの予測は現在の分岐命令アドレスに BTB からの出力を加算することによって行なう。変位のほとんどはある値以下に分布していることに着目して、分岐予測成功率をほとんど低下させることなく BTB の分岐先アドレス部のハードウェア量削減を実現した。しかしこの方式では、変位が BTB に保持できない程に大きい場合は正しい分岐先アドレスを算出できないという問題点がある。

3. 分岐先アドレスの性質を利用した予測手法

本章では、分岐先アドレスと分岐命令アドレスとの間の性質を調べ、それを利用した分岐先アドレスの予測手法を提案する。

3.1 分岐距離

分岐命令アドレスと分岐先アドレスとの差分を**分岐距離**という。分岐距離のほとんどはある値以下に分布している[7]。図 1 に分岐距離の累積分布 (動的分岐数) を SPECint95 について調べた結果 (4 章での測定と同様、100M 命令分のトレースを用いて測定) を示す。図中の分岐距離は、2 進数で表記したときに要するビット数 (符号ビットも含む) で表す。

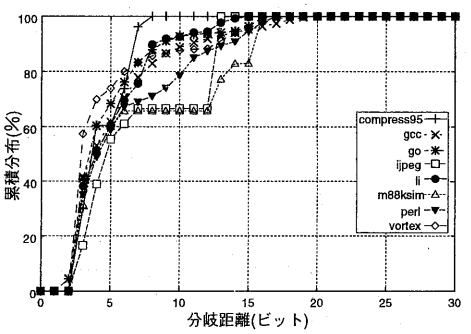

図 1　分岐距離の累積分布

図 1 より、分岐距離はほとんどが 15 ビット以下であり、20 ビット以上のものは存在しないことがわかる。

3.2 PC 連結方式

前節の結果より、分岐命令アドレスと分岐先アドレスの上位には内容が同一である箇所が存在する。分岐距離が小さい程この同一であるビット数は多くなる。そこで、分岐先アドレスの下位部分のみを BTB に保持し、これと現在の分岐命令アドレスとを連結することにより目的の分岐先アドレスを生成する手法が考えられる。この方式を **PC 連結方式**と呼ぶことにする。

図 2に PC 連結方式の動作を示す。命令アドレスを BTB へのインデクスとする。命令アドレスの総ビット数が A ビットであることに対し、保持する分岐先アドレス情報は分岐先アドレスの下位部分の $a(a < A)$ ビットだけである点が特徴である。分岐先アドレスを全て保持する従来の BTB 方式に対して、エントリあたりのハードウェア量を $A-a$ ビット削減することができる。

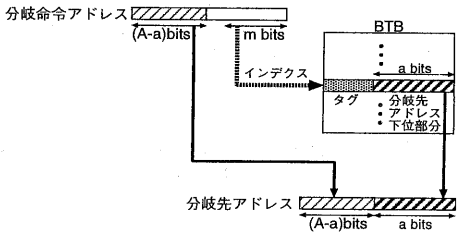

図 2　PC 連結方式による分岐先アドレス予測

分岐先アドレス部のハードウェア量は以下の式で表される。

$$\text{ハードウェア量} = \text{エントリ数} \times a$$

PC 連結方式は容易に BTB のハードウェア量を削減することができるが、変位保持方式と類似した問題点がある。すなわち、分岐先アドレス部ビット数で表現できる分岐距離よりも大きな分岐距離の場合には、正しい分岐先アドレスを表現できないことである。分岐先アドレスを正確に予測するためには分岐先アドレス部ビット数を多くしておかねばならない。しかし、そうすると分岐距離が小さい場合は分岐先アドレス部のハードウェアを無駄にしてしまう。

3.3 2 レベル表方式

PC 連結方式での問題点を解決するために、我々は 2 レベル表方式を提案する。2 レベル表方式では、分岐距離が大きく PC 連結方式では正しく分岐先アドレスを表現できない場合に備えて、PC 連結方式のテーブルに加えて分岐先アドレスの上位部分を保持する別のテーブルを用意する。

図 3に 2 レベル表方式の動作を示す。分岐先アドレスの下位部分を保持するテーブルを第 1 テーブルと呼び、上位部分を保持するテーブルを第 2 テーブルと呼ぶことにする。分岐命令アドレスを双方のテーブルへのインデクスとする。分岐先アドレスの上位にどちらの情報を用いるかの選択には、第 1 テーブルの各エントリに付加した判定用ビットの値を用いる。このビットを **F ビット** (Far ビット；分岐距離が大きい時に第 2 テーブルを使用させるため) と呼ぶことにする。F ビットの値が 0 なら分岐命令アドレスの上位を用い、1 なら第 2 テーブルからの出力を用いるようにする。

分岐先アドレス部と F ビットのハードウェア量の合計は以下の式で表される。

$$\text{ハードウェア量} = \text{第 1 テーブルのエントリ数} \times (a+1) + \text{第 2 テーブルのエントリ数} \times (A-a)$$

テーブル更新の際には、まず実際に算出された分岐先アドレスと分岐命令アドレスとを比較する。分岐距離が小さく第 2 テーブルを使用する必要がない場合、第 1 テーブルのみを更新し、第 1 テーブルの当該エントリの F ビットには 0 をセットす

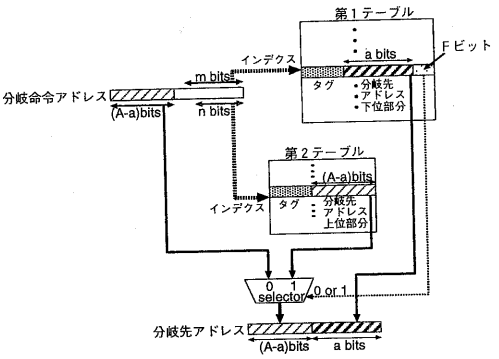

図3　2レベル表方式による分岐先アドレス予測

る。分岐距離が大きく第2テーブルを使用する必要がある場合、第1テーブルと第2テーブルの両方を更新し、第1テーブルの当該エントリのFビットには1をセットする。第1テーブルのエントリは常に用いられるが、第2テーブルのエントリは分岐距離が大きい場合にのみ用いればよいので、第2テーブルのエントリ数は第1テーブルのエントリ数に比べて少なくて済む。従って、その分全体のハードウェア量が少なくて済むことが期待される。

4. 2レベル表方式の評価

本章では、まず測定条件や構成の決定をし、続いて2レベル表方式がどれだけハードウェア量を削減できるのかを評価、検討する。

4.1 評価環境

トレース駆動シミュレーションで評価をおこなった。トレースはNEC EWS4800/360AD(CPUはMIPS R4400)でコンパイルしたプログラムを実行して採取した。ベンチマーク・プログラムはSPECint95とし、評価では実行の最初の100M命令分のトレースを使用した。

4.2 共通の測定条件

全てのBTB構成方式において共通する条件を以下に挙げる。

- リターン命令の場合には、リターン先アドレスはリターン・スタック[9]を用いて予測し、BTBを用いない。リターン・スタックの深さは十分あるとする。
- 分岐方向は全て正しく予測できるものとする。
- BTBの更新は、無条件分岐の場合には常に行なう。条件分岐の場合には分岐方向がtakenの場合に行なう。
- 予測成功率とは、動的な無条件分岐とtakenの条件分岐に対し、BTBによって予測した分岐先アドレスが正しかった割合を意味する。
- 命令アドレスは30ビットとする。

以降で、上限予測成功率という言葉を用いる。比較対象とする方式に対する上限予測成功率とは、それと同じエントリ数及び連想度(2レベル表方式では第1テーブルのエントリ数及び連想度)を持つ従来方式のBTBで得られる予測成功率を意味する。この値は評価において予測成功率の基準として使用する。

4.3 BTBのエントリ数と連想度

本節ではPC連結方式及び2レベル表方式の第1テーブルのエントリ数と連想度を定める。図4に従来方式のBTBでエントリ数と連想度を変化させて予測成功率を測定した結果を示す。2本で組になっている棒グラフのうち、左側がダイレクトマップの場合、右側が連想度2の場合の予測成功率である。図中のエントリ数65536、連想度64は、非常に大きいエントリ数と連想度として設定した。

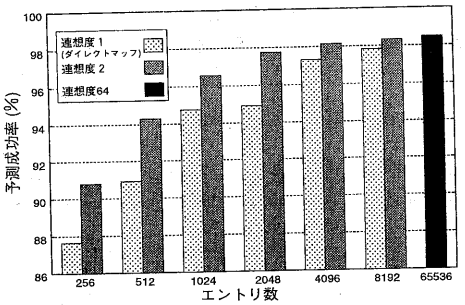

図4　BTBのエントリ数及び連想度と予測成功率

図4より、現実的なハードウェア量でかつ限界の予測成功率に近い予測成功率を得ることのできる構成として、PC連結方式及び2レベル表方式の第1テーブルをエントリ数2048、連想度2とする(この構成は、非常に大きいエントリ数と連想度の構成における予測成功率の99%以上の予測成功率を得ている)。以降(2レベル方式では第1テーブルについて)特に断りの無い限りこの構成を使用する。

4.4 PC連結方式の評価

本節ではPC連結方式で予測成功率を低下させずにどれだけハードウェア量を削減できるかを調べる。図5に分岐先アドレス部ビット数を変化させて予測成功率を測定した結果を示す。図におけるハードウェア量とは、分岐先アドレス部のハードウェア量を表し、タグ部のハードウェア量は含まれていない。図中で矢印で示すポイントは、各々のベンチマークにおいて最も少ないハードウェア量で上限予測成功率を得ることができる箇所である。横軸付近の括弧つき数字はBTBの分岐先アドレス部ビット数を表す。

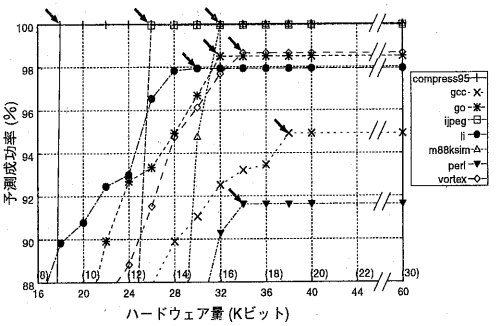

図5　PC連結方式でのハードウェア量と予測成功率

compress95、ijpeg、m88ksimはハードウェア量を小さくしていくと矢印で示した箇所から急激に予測成功率が低下する。これらのプログラムでは非常に小数の分岐が多く実行されていて、矢印で示した箇所より少ないハードウェア量ではこれらの分岐先アドレスが表現できなくなるためである。これら以外のプログラムでは、プログラム内の多くの分岐が実行されるので、比較的ゆるやかに予測成功率が低下していく。

上限予測成功率を得るための分岐先アドレス部ビット数の最小値はベンチマークごとに異なるが、全てのベンチマークに渡って予測成功率を低下させないためにはそれらの最大値を選ばなければならず、分岐先アドレス部ビット数は19ビット必要であるといえる。

4.5 2レベル表方式の評価

4.5.1 第2テーブルに必要なエントリ数

本節では2レベル表方式における第2テーブルに必要なエントリ数を見積もる。図6にベンチマークごとに第2テーブルを使用する必要のある静的分岐数を測定した結果を示す。なお、gccでは、第1テーブルの分岐先アドレス部ビット数が13より小さい場合、第2テーブルを使用する必要のある静的分岐数は4000を越えており、測定の意味はないので測定していない。

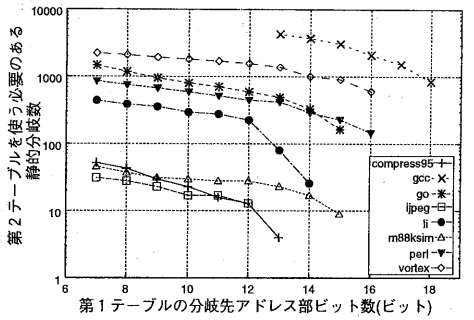

図6 第2テーブルを使用する必要のある静的分岐命令数

第1テーブルの分岐先アドレス部ビット数が7の場合の第2テーブルを使用しなくてはならない静的分岐命令数によって、ベンチマークを以下のように3つのグループに分類する。

- 少ないもの(100未満):ijpeg,compress95,m88ksim
- 多いもの(100以上1000未満):li,perl
- 非常に多いもの(1000以上):go,vortex,gcc

以降、この3つのグループを代表して、gcc、li、m88ksimの結果を示し考察する。

4.5.2 第2テーブルを完全連想で構成した場合

本節では第2テーブルのエントリ数を定めた場合の評価を行う。図7に第2テーブルを完全連想で構成し、各々のエントリ数において第1テーブルの分岐先アドレス部ビット数を変化させて予測成功率を測定した結果を示す。図中には、上限予測成功率、上限予測成功率の99.9%、及び上限予測成功率の99%の値を併せて示す。図におけるハードウェア量とは、分岐先アドレス部とFビットのハードウェア量の合計を表し、タグ部のハードウェア量は含まれていない。

いずれのベンチマークにおいても、PC連結方式に対して2レベル表方式は同等の予測成功率をより少ないハードウェア量で実現している。

m88ksimでは上限予測成功率をPC連結方式よりも非常に少ないハードウェア量で実現している。m88ksimは第2テーブルを使用する必要のある静的分岐数が少ないためである。

liでは、上限予測成功率付近で比較すると、第2テーブルのエントリ数が256の場合に予測成功率に対するハードウェア量が最も少なくて済む。エントリ数が256より少ない場合ではエントリ数が不足して置き換えがなされてしまうので予測成功率が低下する。これを避けるためには、第2テーブルを使用する頻度を低くするために第1テーブルの分岐先アドレス部のビット数を大きくしなくてはならず、ハードウェア量が大きくなってしまう。エントリ数が256より多い場合ではエントリが十分に使用されずハードウェアが無駄になっている。また、上限予測成功率の99.9%付近で比較すると、エントリ数が128の場合に予測成功率に対するハードウェア量が最も少なくて済む。

gccはliに比べ第2テーブルを使用する必要のある静的分岐数が多いので、上限予測成功率の99.9%付近では第2テーブルのエントリ数が512の場合に、上限予測成功率の99%付近で

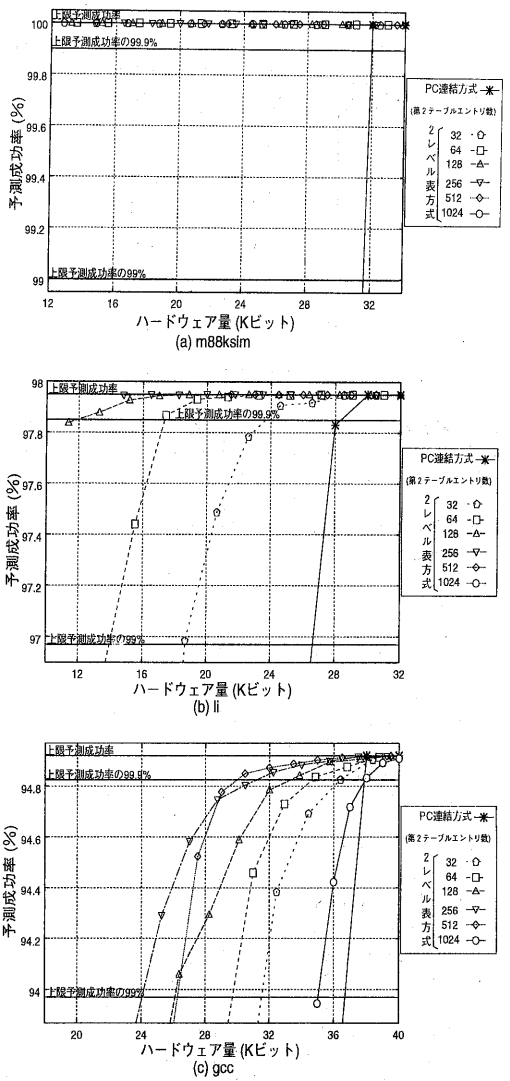

図7 2レベル表方式、第2テーブルを完全連想にした場合のハードウェア量と予測成功率

はエントリ数が256の場合に、予測成功率に対するハードウェア量が最も少なくて済む。

4.5.3 第2テーブルをセット連想にした時の測定

連想度を持つ場合には、比較や置き換えエントリの選択のためのハードウェア量や遅延の問題がある。従って、前節で用いたような極端に大きな連想度のテーブルの実現は現実的には困難である。本節では第2テーブルをセット連想で構成し、完全連想での結果とも比較し評価を行なう。

図8に第2テーブルをセット連想で構成しエントリ数及び連想度を変化させて予測成功率を測定した結果を示す。グラフの見方は図7と同様である。

図8より、2レベル表方式は第2テーブルをセット連想で構成した場合でも、PC連結方式に対して同等の予測成功率をより少ないハードウェア量で実現できることがわかる。また、連想度2の場合に比べ、連想度4の場合には同等の予測成功率を

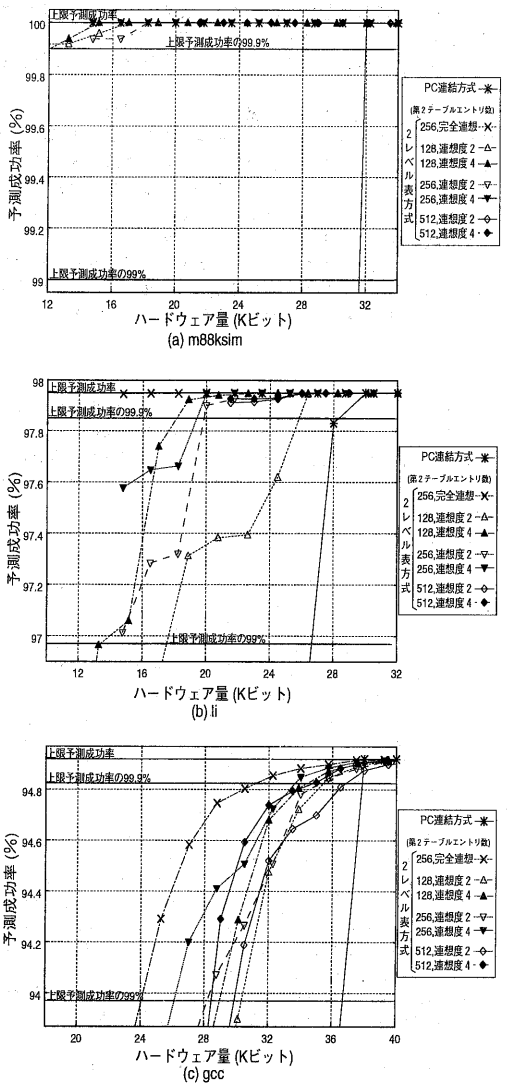

図8 2レベル表方式、第2テーブルをセット連想にした場合のハードウェア量と予測成功率

より少ないハードウェア量で実現でき、同等のハードウェア量でより高い予測成功率を得ている。

m88ksim ではエントリ数256、連想度4の構成で上限予測成功率をPC連結方式の約46%のハードウェア量で得ることができる。

li では m88ksim 程に多くはハードウェア量を削減できないが、エントリ数256、連想度4の構成で上限予測成功率をPC連結方式の約81%のハードウェア量で得ることができる。上限予測成功率より低い予測成功率で比較した場合はより多くハードウェア量を削減できる。エントリ数128、連想度4の構成で上限予測成功率の99%をPC連結方式の約44%のハードウェア量で得ることができる。

ただし、gcc のように第2テーブルを使用する必要のある静的分岐数が非常に多い場合、2レベル表方式では上限予測成功率とほとんど等しい予測成功率を得ようとすると、PC連結方式に対するハードウェア量の削減は少ない。エントリ数256、連想度4の構成で上限予測成功率の99.9%をPC連結方式に対して約10%しかハードウェア量を削減できない。しかし、上限予測成功率の99%程度の予測成功率で良いならば、約30%程度ハードウェア量を削減できる。

4.5.4 全てのベンチマークに渡る評価

本節では全ベンチマークの平均予測成功率を用いて2レベル表方式がどれだけハードウェア量を削減できたかを比較する。図9にハードウェア量と平均予測成功率(全ベンチマークによる算術平均値)の関係を示す。グラフの見方は図7と同様である。

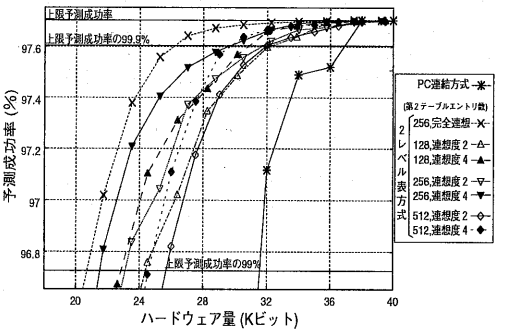

図9 2レベル表方式、ハードウェア量と平均予測成功率

図9より、全ベンチマークの平均値で比較すると、第2テーブルのエントリ数が256の場合に予測成功率に対してハードウェア量が最も少なくて済む。図9を用いて上限予測成功率(97.70%)、上限予測成功率の99.9%(97.61%)付近、及び99%(96.73%)付近のの予測成功率における、各々の機構でのハードウェア量を比較した。それを表1にまとめる。構成の詳細を表2に示す。予測成功率は全ベンチマークによる算術平均値である。

表1 平均予測成功率とハードウェア量の比較

構成		予測成功率(%)	ハードウェア量		
			Kビット	相対量(%)	
従来方式		97.70	60.00	100	157.89
最適なPC連結方式*		97.70	38.00	63.33	100
2レベル表方式	A2	97.60	30.50	50.83	80.26
	A4	97.58	28.75	47.92	75.66
	B2	96.84	23.50	39.17	61.84
	B4	96.81	21.75	36.25	57.24

表2 表1における構成の詳細

構成		第1テーブルの分岐先アドレス部ビット数(ビット)	第2テーブル	
			エントリ数	連想度
最適なPC連結方式*		19	-	-
2レベル表方式	A2	12	256	2
	A4	11	256	4
	B2	8	256	2
	B4	7	256	4

* 最も少ないハードウェア量で上限予測成功率を得られる構成

表1より、2レベル表方式を用いた場合、上限予測成功率の99～99.9%の予測成功率を、最適なPC連結方式に対して約58～80%のハードウェア量で実現でき、分岐先アドレスの全てのビットを保持する従来方式に対して約36～51%のハードウェア量で実現できることがわかる。

4.6 エントリ数が異なる従来方式との比較

前節で述べたように、2レベル表方式は予測成功率をほとんど低下させることなく、ハードウェア量を従来方式の半分程度にまで削減できる。そこで、本節では、ハードウェア量が半分の従来方式、すなわちエントリ数を半分にした従来方式との比較をし、2レベル表方式を適用させると適用前と同程度のハードウェア量で予測成功率がどの程度向上できるのかを調べる。

図10にエントリ数1024の従来方式(連想度2)、エントリ数2048のPC連結方式(連想度2)、4.5.4節の表1で示した2レベル表方式(第1テーブル：エントリ数2048、連想度2、第2テーブル：エントリ数256、連想度2及び4)を比較した結果を示す。グラフの見方は図7と同様である。

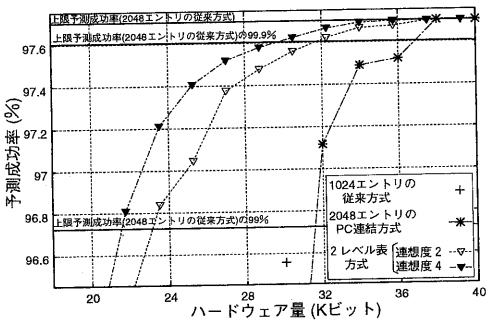

図10 エントリ数1024の従来方式との比較

2048エントリのPC連結方式は、1024エントリの従来方式と同等のハードウェア量では、1024エントリの従来方式と同等の予測成功率を達成することはできない。

一方、2レベル表方式は、1024エントリの従来方式と同等の予測成功率をより少ないハードウェア量で実現できる。また、1024エントリの従来方式のハードウエア量(30Kビット)とほぼ同程度のハードウェア量で1024エントリの従来方式よりも約1.07%高い予測成功率を得ていて、この予測成功率はエントリ数2048の従来方式での予測成功率の約99.9%である。すなわち、2レベル表方式を適用すると、1024エントリの従来方式のハードウェア量で2048エントリの従来方式に匹敵する予測成功率を得ることができる。

5. まとめ

マイクロプロセッサの高性能化技術として分岐予測機構にはより高い精度が望まれている。しかし、分岐先アドレスを予測するBTBには、高い予測成功率を得るためにはハードウェア量が大きくなるという問題がある。

そこで本稿では、分岐命令アドレスと分岐先アドレスとの距離(分岐距離)の性質を利用してBTBのハードウェア量を削減する手法として2レベル表方式を提案した。これによって、従来のBTB方式に対して分岐先アドレス予測成功率をほとんど低下させることなく分岐先アドレス部のハードウェア量を約52%削減させることができた。また、2レベル表方式は、従来方式と同程度のハードウェア量で分岐先アドレス予測成功率を約1.07%高めることができた。

今回の測定では、BTBのタグ部のハードウェア量を考えずに行った。しかし、連想度を持ったテーブルを実際に構成する時にはタグは不可欠となる。そこで、提案手法に対してタグ部ビット数を変化させて測定し、タグ部及びBTB全体のハードウェア量と予測成功率とのトレードオフを評価することが考えられる。

2レベル表方式において、第2テーブルに要求されるエントリ数が多い場合、分岐先アドレス予測成功率が上限予測成功率に対して悪化する。ある範囲のアドレス空間では分岐先アドレスの上位は同一の値であるので、そのような分岐命令間で第2テーブルのエントリを共有させれば、第2テーブルに必要なエントリ数を減少させる事が可能になると考えられる。今回は第2テーブルへのインデクスには命令アドレスの下位ビットをそのまま用いたが、上記の目的を達成できる第2テーブルへのインデクス手法を検討することが将来の課題として挙げられる。

謝辞

本研究の一部は、文部省科学研究費補助金基盤研究(C)「広域命令レベル並列によるマイクロプロセッサの高性能化に関する研究」(課題番号1068034)及び財団法人堀情報科学振興財団助成「広域命令レベル並列性を利用するコンピュータ・アーキテクチャとコンパイラに関する研究」の支援により行った。ここに感謝の意を表する。

参考文献

1) B. Calder and D. Grunwald, "The Precomputed Branch Architecture," University of California, San Diego, Technical Report CS97-526, March 1997.
2) P. Chang, E. Hao, and Y. N. Patt, "Target Prediction for Indirect Jumps," In *Proc. 24th Int. Symp. on Computer Architecture*, pp. 274-283, May 1997.
3) B. Fagin, "Partial Resolution in Branch Target Buffers," *IEEE Transactions on Computers*, vol.46, no.10, pp.1142-1145, October 1997.
4) B. Fagin and K. Russeell "Partial Resolution in Branch Target Buffers," In *Proc. 28th Int. Symp. on Microarchtecture*, pp.193-198, December 1995.
5) B. Fagin and A. Mital, "The Performance of Counter and Correlation-Based Schemes for Branch Target Buffers," *IEEE Transactions on Computers*, vol.44, no.12, pp.2-36, December 1995.
6) 原, 安藤, 中西, 中屋, "分岐先バッファにおける分岐先情報の削減," 第49回情報処理学会全国大会 IL-2 pp.1-2, 1994年9月.
7) J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach, 2nd Edition*, Morgan Kaufmann Publishing Inc., San Francisco, CA, 1996.
8) K. Driesen and U.Hölzle, "Accurate Indirect Branch Prediction," In *Proc. 25th Int. Symp. on Computer Architecture*, June 1998.
9) D. R. Kaeli and P. G. Emma, "Branch History Table Prediction of Moving Target Branches Due to Subroutine Returns," In *Proc. 18th Int. Symp. on Computer Architecture*, pp.34-42, May 1991.
10) J. K. F. Lee and A. J. Smith, "Branch Prediction Strategies and Branch Target Buffer Design," *IEEE Transactions on Computers*, vol.17, no.1, pp.6-22, January 1984.
11) S. McFarling, "Combining Branch Predictors," *WRL Technical Note TN-36*, Digital Equipment Corporation, June 1993.
12) H. Wang, T.Sun, and Q.Yang, "Minimizing Area Cost of On-Chip Cache Memories by Caching Address Tags," *IEEE Transactions on Computers*, vol.46, no.11, pp.1187-1201, November 1997.