計算機アーキテクチャ 141－11
（２００１．１．２６）

値予測を用いた分岐予測機構の計算機性能に与える影響

中村 幸司　片山 清和　布施 裕基　安藤 秀樹　島田 俊夫

名古屋大学大学院工学研究科

命令の実行結果を予測する値予測は、将来のスーパースカラ・プロセッサには必須の技術になると思われる。我々は値予測を分岐予測に利用する機構を提案してきた。この機構は、分岐命令のオペコードとソースオペランドの値を予測し、分岐方向を予測するものであり、従来の分岐予測機構とのハイブリッド構成で使用することで最終的な予測精度を改善する。本論文は、値予測を利用した分岐予測機構と従来の分岐予測機構とのハイブリッド予測機構が計算機性能にどの程度影響を与えるか測定した。SPECint95 ベンチマークを用いた評価の結果、命令発行幅が増え、深いパイプラインを持つ将来のスーパースカラ・プロセッサでは最大 18.0%、平均 5.6%計算機性能の向上が見込めることを確認した。また、選択器の性能を向上させるために、選択表のインデクスに分岐方向の履歴情報を付加する手法を提案する。測定の結果、将来のスーパースカラ・プロセッサでは計算機性能を最大 21.5%、平均 9.9%引き上げることを確認した。

The Impact of Branch Prediction with Value Prediction

Kouji Nakamura, Kiyokazu Katayama, Yuuki Fuse, Hideki Ando, and Toshio Shimada

Graduate School of Engineering, Nagoya University

Value prediction that predicts the results of instructions will be an essential technique for the future superscalar processors. We proposed a branch prediction mechanism that uses value prediction. This mechanism predicts operand values and opcode of a branch, and then predicts its outcome. This mechanism improves branch prediction accuracy by being incorporated in a hybrid predictor with conventional branch predictor. Our evaluation results show that this hybrid branch predictor improves computer performance by a maximum amount of 18.0% or an average of 5.6% over 4K-byte gshare on SPECint95 benchmarks in the future superscalar processor with a deep pipeline and wide issue width. In addition, we propose a technique to improve performance of the selector by adding branch history to the index of the selection table. Our evaluation results show that the predictor with our technique improves computer performance by a maximum amount of 21.5% or an average of 9.9% in the future superscalar processor over 4K-byte gshare.

1　はじめに

分岐予測は、制御依存を緩和することにより命令レベル並列性 (ILP) を増加させ、プロセッサ性能の向上を図る技術である。分岐予測精度は命令発行幅が広く深いパイプラインを持っている近年のスーパースカラ・プロセッサに大きな影響を与える。今後も、命令発行幅はより広く、パイプラインはより深くなる傾向にあり、分岐予測精度の向上はプロセッサ性能にさらに大きな影響を与える。

これまでに考案された分岐予測方式の多くは、過去の分岐の振る舞いと将来の分岐の振る舞いとの間に相関があることを利用している。なかでも 2 レベル分岐予測方式 [8][14][15] は、過去の分岐方向の履歴パターンと将来の分岐方向との間に存在する相関を利用し、高い予測精度を実現している。しかし、分岐予測精度はわずかな改善でもプロセッサ性能に大きな影響を与えるので [11]、予測精度の向上は今なお重要な課題である。

一方、データ依存を緩和させることによって ILP を増加させ、プロセッサ性能の向上を図る技術が近年盛んに研究されている。この技術の 1 つに、命令の実行結果を予測する値予測がある [4][6][7][12][13]。これまでに提案されている値予測の多くは、分岐予測と同様に、過去の値の振る舞いと将来の値の振る舞いとの間に相関があることを利用している。そのため、命令アドレスをインデクスとする**値履歴表** (VHT：Value History Table) を用意し、命令の実行結果の履歴を保持させる。

値予測の中で最も単純な手法である**最終値予測** [6][7] は、命令の最も最近の実行結果を予測値とする。**ストライド値予測** [4][10][13] は、一定の差分 (ストライド) が検出されていれば最近の実行結果にストライドを加えた値を予測値とする。**2 レベル値予測** [13] は、2 レベル分岐予測と同様の考え方で、過去の値の振る舞いのパターンに対する相関を利用して予測を行う。これらの値予測は非常に高い予測精度を示すというわけではないので、予測結果が信頼できるかどうかを判断し、信頼できる時のみ予測結果を使用する方法がよくとられる (例えば文献 [3])。

過去の研究によれば、現在のプロセッサでは値予測を用いることによる性能向上は小さいものの

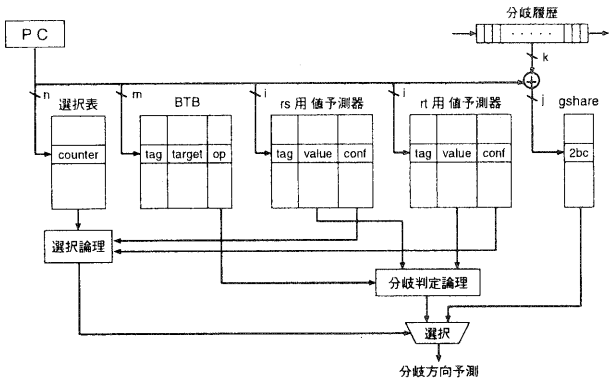


図 1: 値予測を利用した分岐予測器と gshare 分岐予測器とのハイブリッド分岐予測器

[7]、命令のフェッチバンド幅が拡大する将来のプロセッサでは、約 50%もの大きな性能向上が期待できる [5]。このことは、値予測が将来のプロセッサには必須の技術として搭載されることを示唆している。

値予測はこれまで、データ依存を緩和させることによって ILP を増加させることを目的として研究されていた。これに対して我々は、値予測を分岐予測に利用する機構を提案した [1]。この機構は、従来の分岐予測器と値予測を利用した分岐予測器とを組み合わせたものである。従来の分岐予測器では予測困難であった分岐の予測を値予測を利用した分岐予測器に割り当て、最終的な分岐予測精度を改善している。しかし、計算機性能がどの程度向上するのかは調べられていない。そこで本論文では、我々が提案した分岐予測器が計算機性能をどの程度向上させるのかを調査する。また、我々が提案した分岐予測器には分岐予測精度を向上させる余地があるので、分岐予測器の改良を行い、その効果を検証する。

以下、2章では背景として我々が文献 [1] で提案した分岐予測器の構成及び動作を述べる。3章ではこの分岐予測器をスーパースカラ・プロセッサに実装するための変更点を述べる。4章で計算機性能に与える影響を測定し、評価結果を述べる。5章では値予測を用いた分岐予測器の改良を行い、評価を行う。6章で本論文をまとめる。

2 背景

2.1 値予測を用いた分岐予測器

図 1に我々が文献 [1] で提案した分岐予測器の構成を示す。従来の分岐予測器として gshare[8] を用い、値予測器には最終値予測器を用いている。他の予測器を用いる場合も図 1と同様に構成できる。

値予測器は分岐命令のソースオペランドの予測に用いる。分岐命令のソースオペランドは 1 つあるいは 2 つなので、値予測器を 2 つ用意する。ソースオペランドが 2 つの分岐命令では両方の値予測器を使用し、1 つの分岐命令では `rs` 用の値予測器のみを使用する。各値予測器の VHT は 2^iのエントリを持つ。各エントリは命令アドレスによって参照され、命令を識別するためのタグ、最終値 (図 1では value)、及び、予測の信頼性を表すリセッティングカウンタ (図 1では conf) からなる。

ソースオペランドの予測値を利用して分岐方向を予測するには、分岐命令のオペコードが必要となる。そのため、分岐先バッファ (BTB：Branch Target Buffer) のエントリに分岐命令のオペコードを格納するフィールドを追加する。これによって、命令フェッチ時に分岐命令のオペコードを予測することができる。

値予測を利用した分岐予測器の動作を述べる。分岐命令アドレスをインデクスとして値予測器及び BTB にアクセスする。値予測器はソースオペランドの値を予測するが、もし、値予測器に予測対象の分岐命令が登録されていない場合、0 と予測する。また、ストライド値予測器を使っている場合で、最終値は保持しているがストライドを検出できていない場合は、保持している最終値を予測値とする。分岐判定論理はソースオペランドの値と BTB から得られた分岐命令のオペコードの予測結果を用いて分岐方向を判定する。

値予測を利用した分岐予測結果と gshare 分岐予測結果のどちらを最終的な出力とするかを決定する選択器は、2^nのエントリを持つ選択表と、選択を行う選択論理からなる。選択論理は、選択表から読み出された値と、BTB のヒット情報 (図 1では省略している)、及び (必要なら) 値予測器の信頼性カウンタの値を用いて選択を行う。BTB ミスの場合はオペコードを得ることができないので、必ず gshare の結果を出力する。BTB ヒットの場合は、選択表から読み出された値が条件を満たしていれば値予測を利用した分岐予測結果を出力する (値予

測器の信頼性カウンタの値を利用する場合はその値が閾値以上であることも条件となる)。

文献[1]によれば、値予測を利用した分岐予測器と4KBのgshare分岐予測器とのハイブリッド構成で、最大0.82%、平均0.42%分岐予測精度を改善できることが確認されている。しかし、提案されたハイブリッド分岐予測器が計算機性能に与える影響は測定されていない。そこで、4章において提案されたハイブリッド分岐予測器がIPCに与える影響を測定し、検証を行う。

また、分岐予測精度改善の上限は最大10.2%、平均でも約3%あることが測定により得られているが、実際の分岐予測精度改善量はその十分の一程度しかない。これは選択器の性能が悪いためであると考えられ、選択器の改良が必要である。これについては5章で検証する。

3 機構の構成

3.1 スーパースカラ・プロセッサへの実装

2.1節の分岐予測器のスーパースカラ・プロセッサへの実装においては、サイクル時間に悪影響を与えないように、値予測器及び選択表の参照までを1サイクル目で行い、2サイクル目で値予測を用いた分岐予測の結果を得るようにした。

スーパースカラ・プロセッサ上での値予測を用いた分岐予測器の予測動作を以下に述べる。まずフェッチ時ではgshare分岐予測の結果を用いて投機的にフェッチを行う。1サイクル後に値予測を用いた分岐予測の結果が判明するので、その結果がgshare分岐予測の結果と異なり、なおかつ選択器が値予測を用いた分岐予測の結果を選択した場合、1サイクルのペナルティを払って再フェッチを行う。

次に値予測を用いた分岐予測器の更新タイミングについて述べる。gshare分岐予測器のPHT、及び選択表の更新は命令コミット時に行う。gshare分岐予測器のインデクスに用いる履歴情報は予測を行った直後にgshare分岐予測の結果を用いて投機的に更新する。値予測器のVHTは分岐命令のソースオペランドのレディビットが全て1になった時に更新する。

4 評価

本章では、値予測を用いた分岐予測器がIPCに与える影響について評価を行う。

4.1 評価環境

シミュレータはSimpleScalar Tool Set[2] Version 3.0aの中のスーパースカラ・プロセッサ用シミュレータを基にして作成した。ベンチマーク・プログラムは、SPECint95の全8種類を使用した。ベンチマーク・プログラムのバイナリのコンパイルには

表1: ベンチマーク・プログラム

ベンチマーク	入力	静的分岐数	動的分岐数
compress95	30000 e 2231	528	10.5M
gcc	genoutput.i	16,470	13.0M
go	6 9 2stone9.in	6,041	9.2M
ijpeg	specmun.ppm	1,495	23.9M
li	train.lsp	679	24.2M
m88ksim	ctl.in	1,258	12.6M
perl	scrabbl.in	2,127	10.4M
vortex	vortex.in	1,428	8.8M

表2: プロセッサパラメータ

(a) 各プロセッサモデル共通のパラメータ

実行レイテンシ	iALU 1, iMULT/DIV 3/20, Ld/St 1, fpALU 2, fpMULT/DIV/SQRT 4/12/24
L1 I-cache	64KB 2-way アソシアティブ、ミスレイテンシ 6 サイクル
L1 D-cache	64KB 2-way アソシアティブ、ミスレイテンシ 6 サイクル
L2 unified cache	256KB 4-way アソシアティブ、ミスレイテンシ 18 サイクル

(b) プロセッサモデルによって変化するパラメータ (命令フェッチ幅8の場合)

命令発行幅	8
RUU	128 エントリ
LSQ	64 エントリ
機能ユニット数	iALU 6, iMULT/DIV 1, Ld/St 2,fpALU 6, fpMULT/DIV/SQRT 1
命令コミット幅	16

GNU GCC Version 2.7.2.3 (コンパイルオプション：-O6, -funroll-loops) を用いた。命令セットはMIPS R10000[9] を拡張したSimpleScalar/PISAとした。

表1に各ベンチマーク・プログラムの入力及び静的条件分岐数、動的条件分岐数を示す。シミュレーション時間が過大にならないようにしつつ、関数の出願頻度をほぼ維持するようにするため、入力のパラメータを調整している。ijpegでは最初の50M命令をスキップした後、450M命令を実行し、vortexでは最初の100M命令をスキップした後、80M命令を実行した。残りのベンチマーク・プログラムでは、命令のスキップを行わず、最後まで実行した。

4.2 評価モデル

評価対象とするプロセッサモデルを以下に示す。

- **MachineA**：命令フェッチ幅8, 分岐予測ミスペナルティ10とした。現在のハイエンドプロセッサを想定している。
- **MachineB**：命令フェッチ幅32, 分岐予測ミスペナルティ10とした。発行命令数を増加させることを重視する方向へ進んだプロセッサを想定している。

- **MachineC**：命令フェッチ幅8, 分岐予測ミスペナルティ40 とした。パイプライン段数を深くしてクロックサイクル速度を向上させることを重視する方向へ進んだプロセッサを想定している。
- **MachineD**：命令フェッチ幅32, 分岐予測ミスペナルティ40 とした。発行命令数、パイプライン段数をともに増加させる方向へ進んだプロセッサを想定している。

4.3 測定条件

各評価モデルで測定する時の条件を以下に示す。まず、各評価モデルで共通の条件について述べる。

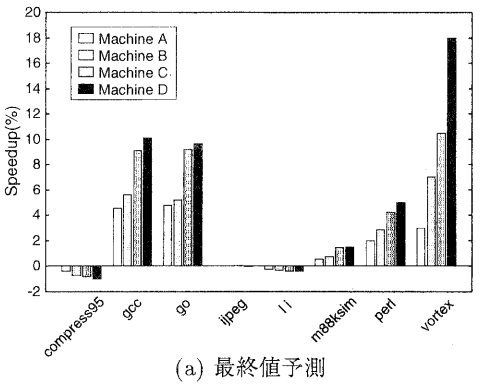
- 値予測として最終値予測、ストライド値予測、2レベル値予測を用いた。1つの値予測器のVHTのエントリ数は16Kエントリとした。また、2レベル値予測では、保持する履歴数を4とした。
- 選択器は16Kエントリの2ビットカウンタと値予測の信頼性カウンタを用いる方式とした。値予測器の信頼性カウンタの閾値は、文献[1]の測定で用いられていた値(最終値予測：3、ストライド値予測：8、2レベル値予測：6)とした。
- 従来の分岐予測器として、16KエントリのPHTを持ち、履歴長8のgshare分岐予測器を用いた。また、BTBは8Kエントリで連想度8とした。
- 他のプロセッサパラメータは表2(a)に示す値を用いた。

次に、評価モデルによって変化するプロセッサパラメータを表2(b)に示す。なお、表2(b)のパラメータはMachineA、MachineCのものである。MachineB、MachineDは命令フェッチ幅がMachineA、MachineCの4倍になるので、表2(b)に対応するパラメータも4倍にした。

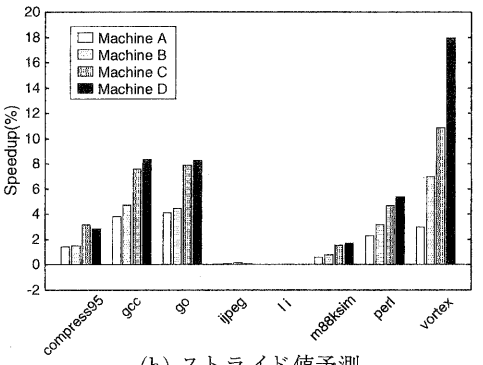
4.4 評価結果

図2に値予測を用いた分岐予測器によるIPCの改善率を示す。同図(a)～(c)はそれぞれ値予測に最終値予測、ストライド値予測、2レベル値予測を用いた場合である。各グラフの縦軸は、4KBのgshare分岐予測器で予測を行った場合のIPCに対する改善率を示す。横軸はベンチマークである。各ベンチマークにつき4本の棒グラフがあり、左から順に、MachineA, MachineB, MachineC, MachineDの測定結果である。また、図3にMachineDを用いた場合の分岐予測ミス率を示す。縦軸は分岐予測ミス率を示す。横軸はベンチマークである。各ベンチマークにつき4本の棒グラフがあり、左から順に、最終値予測、ストライド値予測、2レベル値予測を用いた場合の測定結果である。最後の1本は比較のためのもので、gshareの予測精度である。

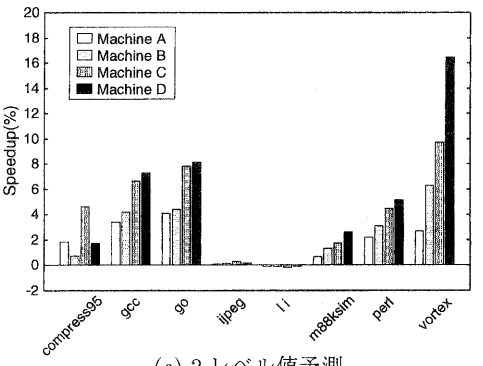
IPCの改善率はMachineAでは最大4.8%、平均1.9%であり、MachineDでは最大18.0%、平均



(a) 最終値予測



(b) ストライド値予測



(c) 2レベル値予測

図 2: ハイブリッド分岐予測器を用いた時のIPC

5.6%であった。

ベンチマーク毎の傾向を以下に示す。

- m88ksim,vortex：分岐予測ミス削減量に比べて、IPCの改善率が高い。理由は次のように考えられる。gshareの分岐予測ミス率が低いため、命令ウインドウ内の有効命令数が多い。それゆえ命令が発行されるまでの時間が長くなり、分岐予測ミスの発見が遅れるため、予測ミスした分岐1命令当りのストールが長い。したがって、

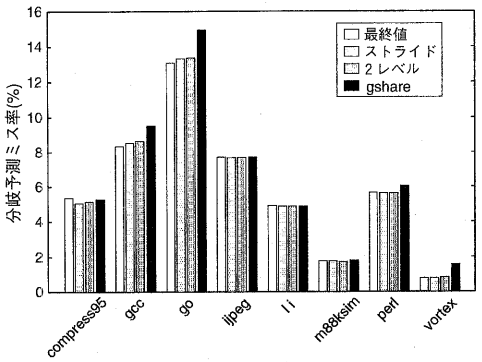


図 3: MachineD における分岐予測ミス率

分岐予測ミスの削減量がわずかであっても、分岐予測ミスによるストールの割合が大きく減少する。これにより、有効なフェッチが増え、実行サイクル数を大きく削減できる。

- gcc,go：gshare の分岐予測ミス率が高いため、値予測を用いることによる予測精度改善の余地が大きく、分岐予測ミスを大きく削減できている。これにより、分岐予測ミスによるストールの発生回数を大きく減らせるため、実行サイクル数を大きく削減でき、IPC の改善率も高い。
- perl：分岐予測ミス削減量は 0.40～0.43 ポイント、IPC の改善率は 5.1～5.4%(共に MachineD の値) である。m88ksim と gcc,go の中間の振る舞いをしている。
- compress95：値予測にストライド予測、2 レベル予測を用いた場合では IPC が改善しているが、分岐予測ミスペナルティを変えずにフェッチ幅を広くすると性能が低下する。命令フェッチ幅が広くなると命令が早く投入され、値予測器、分岐予測器、選択表の内容がより古い状態で予測を行うことになってしまい、予測精度を低下させる。compress95 ではその影響が大きく、上記のような現象が起こると思われる。
- ijpeg,li：分岐予測ミス削減量はほとんど 0 ポイントなので、命令フェッチ幅やパイプライン段数にかかわらず IPC の改善率はほとんど 0%である。

5 値予測を利用した分岐予測器の改良

本章では、値予測を利用した分岐予測器の中で、どちらの予測結果を用いるかを決めるための選択器の改良を行う。

値予測を利用した分岐予測器が持つ予測精度の改善可能性は最大 10.2%、平均でも約 3%あるにも関わらず、実際の予測精度改善量は最大 0.82%、平均 0.42%しかない。これは、選択器の性能が悪く、

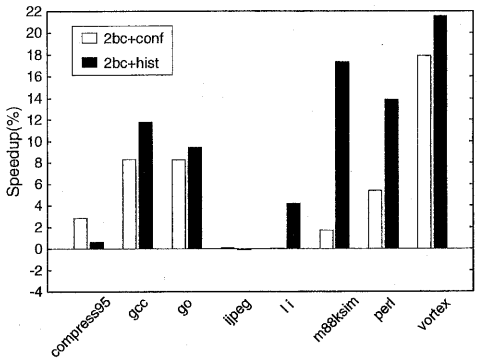


図 4: 選択表を改良した時の IPC (ストライド予測)

正しい予測結果をうまく選択できていないことを示している。文献 [1] で提案されている分岐予測器は選択表へのインデクスに命令アドレスを用いている。本章ではこの部分に着目し、命令アドレスと分岐履歴の XOR をとった値を選択表へのインデクスに用いる手法を提案する。

5.1 選択表の性質に基づく改良

分岐予測も値予測も過去の命令の振る舞いと将来の命令の振る舞いとの相関を利用している。したがって、ある分岐履歴パターンでは、従来の分岐予測器のみが正しく予測できる場面があり、逆に値予測器を用いた分岐予測器のみが正しく予測できている場面も存在する。選択表へのインデクスに分岐履歴情報を利用すれば、上記のようなケースにおいて最終的な予測精度をさらに高めることができると考えられる。

5.2 性能評価

本節では、選択表へのインデクスに分岐履歴を付加した場合と選択論理に値予測の信頼性を付加した場合とで IPC 改善率を比較した。

評価環境は 4.1節と同じとし、評価モデルは 4.2節の MachineD を用いた。また、測定条件は選択器以外は 4.3と同様にした。選択器は、16K エントリの 2 ビットカウンタと値予測の信頼性カウンタを用いる方式と 16K エントリの 2 ビットカウンタ単体でインデクスに分岐履歴情報を利用する方式の 2 種類を用いた。選択表へのインデクスに用いる分岐履歴の履歴長は 14 とした。これは測定によって最適なものを選択した。また、gshare の分岐履歴と同じタイミングで、gshare の予測結果を用いて投機的に更新するようにした。

図 4に評価結果を示す。紙面スペースの都合上、値予測にストライド値予測を用いた場合のみを示す。各グラフの縦軸は 4KB の gshare 分岐予測器で予測を行った場合の IPC に対する改善率を示す。横軸はベンチマークである。各ベンチマークにつ

き2本の棒グラフがあり、左から順に、選択表と値予測の信頼性カウンタを用いる場合、選択表のインデクスに分岐履歴情報を利用する場合の測定結果である。

選択表のインデクスに分岐履歴情報を利用した場合、IPCの改善率は最大21.5%、平均9.9%であった。compress95,ijpeg以外のベンチマークでは、選択表のインデクスに分岐履歴情報を利用することでIPCの改善率が増加した。特にli,m88ksim,perlで著しく増加しており、liではIPC改善率が−0.4〜0.2%から2.5〜4.2%に、perlでは5.1〜5.4%から13.9〜14.6%に、m88ksimでは1.5〜2.6%から7.0〜17.4%に増加した。このことから、選択表へのインデクスに分岐履歴情報を付加することは効果的であるといえる。

6 まとめ

命令の実行結果を予測する値予測は、データ依存を緩和し性能を向上させるために、将来のスーパースカラ・プロセッサには必須の技術になる。我々は値予測器の予測結果を分岐予測に利用する機構を提案してきた。この機構は、分岐命令のオペコードとソースオペランドの値を予測し、分岐方向を予測するものであり、従来の分岐予測器とのハイブリッド構成で使用することで最終的な予測精度を改善する。本論文は、値予測を利用した分岐予測器と従来の分岐予測器とのハイブリッド分岐予測器が計算機性能にどの程度影響を与えるか測定した。選択器として2ビットカウンタと値予測器の信頼性カウンタを利用する方式を用いて、SPECint95ベンチマークで評価した。その結果、今日のスーパースカラ・プロセッサでは最大4.8%、平均1.9%計算機性能の向上が見込めることを確認した。そして、命令発行幅が増え、パイプライン段数が深い将来のスーパースカラ・プロセッサでは最大18.0%、平均5.6% 計算機性能の向上が見込めることを確認した。

値予測を利用した分岐予測器に潜在する性能を引き出すには、どちらの分岐予測結果を用いるのかを正しく選択できることが必要である。そこで、選択表のインデクスに分岐命令アドレスと分岐方向の履歴情報を用いる手法を提案した。測定の結果、将来のスーパースカラ・プロセッサで最大21.5%、平均9.9%計算機性能を引き上げることができ、値予測器の信頼性カウンタを利用する方式より有効であることがわかった。

謝辞

本研究の一部は、文部省科学技術費補助金基盤研究(C)課題番号1068034、同じく課題番号11680351、及び財団法人カシオ科学振興財団研究助成の支援により行った。

参考文献

[1] 戸田聡、布施裕基、片山清和、安藤秀樹、島田俊夫、"値予測を用いた分岐予測," *2000*年並列処理シンポジウム *JSPP2000*, pp.237-244, 2000年6月.

[2] D. Burger and T. M. Austin, "The SimpleScalar Tool Set, Version 2.0," Technical Report, University of Wisconsin-Madison, CS-TR-97-1342, June 1997.

[3] B. Calder,G. Reinmann,and D. M. Tullsen, "Selective Value Prediction," In *Proc. 26th Int. Symp. on Computer Architecture*, pp.64-74, May 1999.

[4] F. Gabbay and A. Mendelson, "Speculative Execution based on Value Prediction," EE Department TR #1080, Technion - Israel Institute of Technology, November 1996.

[5] F. Gabbay and A. Mendelson, "The Effect of Instruction Fetch Bandwidth on Value Prediction," In *Proc. 25th Int. Symp. on Computer Architecture*, pp.272-281, July 1998.

[6] M. H. Lipasti, C. B. Wilkerson, and P. J. Shen, "Value Locality and Load Value Prediction," In *Proc. Seventh Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, pp.138-147, Octorber 1996.

[7] M. H. Lipasti and P. J. Shen, "Exceeding the Dataflow Limit via Value Prediction," In *Proc. MICRO-29*, pp.226-237, December 1996.

[8] S. McFarling, "Combining Branch Predictors," *WRL Technical Note*, TN-36, Digital Equipment Corporation, June 1993.

[9] MIPS Technologies, Inc., "MIPS R10000 Processor User's Manual, Version 2," October 1996.

[10] 西本晴子, 勝野昭, 木村康則, "ロードアドレス予測による命令並列度の向上," 情報処理学会研究会報告, 96-ARC-119, pp.49-54, 1996年8月.

[11] 野口良太、森敦司、小林良太郎、安藤秀樹、島田俊夫、"分岐方向の偏りを利用し破壊的競合を低減する分岐予測機構," 情報処理学会論文誌, vol. 40, no. 5, pp.2119-2131, 1999年5月.

[12] Y. Sazeides and J. E. Smith, "The Predictability of Data Values," In *Proc. MICRO-30*, pp.248-258, December 1997.

[13] K. Wang and M. Franklin, "Highly Accurate Data Value Prediction using Hybrid Predictors, " In *Proc. MICRO-30*, pp.281-290, December 1997.

[14] T-Y. Yeh and Y. Patt, "Two-Level Adaptive Branch Prediction," In *Proc. 24th Int. Symp. and Workshop on Microarchitecture*, pp.55-61, November 1991.

[15] T-Y. Yeh and Y. Patt, "A Comparison of Dynamic Branch Predictors that use Two Levels of Branch History," In *Proc. 20th Int. Symp. on Computer Architecture*, pp.257-266, May 1993.