2レベル・ストライド値予測機構の可能性検討

吉瀬謙二[†,☆] 坂井修一[†] 田中英彦[†]

演算結果を予測することで真のデータ依存関係を解消する値予測の手法を用い，プロセッサで利用できる命令レベル並列性の向上を目指す．値予測の機構として，値予測を行う命令までの制御流の変化を考慮することで，ストライド値予測機構の欠点を補う2レベル・ストライド値予測機構を新たに提案し，その可能性を検討する．2レベル・ストライド値予測機構におけるいくつかのパラメータを変化させ予測ヒット率とミス率を測定する．また，スーパースカラ・プロセッサのシミュレータを用いた評価より，プロセッサの性能に与える影響を議論する．

Performance Potential of Two-level Stride Value Predictor

KENJI KISE,[†,☆] SHUICHI SAKAI[†] and HIDEHIKO TANAKA[†]

This paper presents two-level stride value predictor, which uses the branch history register to make up for the weak point of the stride value predictor. The purpose of this paper is to verify the performance potential of the two-level stride value predictor. Some parameters in the predictor are made to change, and a prediction hit rate and miss rate are measured. An influence on the processor performance is examined from the evaluation result with a superscalar simulator.

1. はじめに

プログラムに内在する並列性は，制御依存関係とデータ依存関係により制限されることが知られている．制御依存関係を解消するための手法として分岐予測を用いた投機処理があり，精度の高い分岐予測機構の提案は重要な研究課題の1つになっている[2),7)]．一方のデータ依存関係は，レジスタ数の不足が原因で生じる出力依存関係，逆依存関係と，ハードウェア資源に依存しない真の依存関係に分類できる[4)]．この中で，出力依存関係と逆依存関係はレジスタ名前替えで解決できる．残る真の依存関係を解消することは困難と考えられてきたが，近年，真の依存関係を解消する技術として，値予測を用いた投機処理の手法が提案された[5),6),10),11)]．これは，実際に計算を行ってデータ値を得る代わりに，生成されるデータの値を予測することで処理を進めておくという投機処理技術である．この値予測により，命令レベル並列性の上限を引き上げることが可能となる[3)]．

予測により正しい値を得ることができれば真の依存関係を解消できるが，予測に失敗すれば誤った値を利用した命令の再処理が必要となり，プロセッサ性能にペナルティを与えてしまう．いくつかの研究[3),5),11)]において，ミス率の低減を考慮した値予測機構が提案されているが，これらの値予測機構を用いても，全実行命令に対して2〜5%程度の頻度で予測ミスが発生する．値予測による性能向上を維持するためにはミス率の低減が重要な課題となっている．

本稿では，値予測を行う命令までの制御流の変化を考慮することで，ストライド値予測機構の欠点を補う2レベル・ストライド値予測機構を提案する．提案手法では，ストライド値予測で予測ミスが発生した際に，予測ミスを引き起こした命令の演算結果とその命令に至るグローバルな分岐履歴をテーブルに保存し，この情報を用いてストライド値予測で予測ミスとなる領域を正しく予測することを試みる．このため，予測ミスを削減できるとともにヒット率の向上を期待できる．2レベル・ストライド値予測機構の評価として，いくつかの重要なパラメータを変化させ，ヒット率とミス率を議論する．また，スーパースカラ・プロセッサのシミュレータを用いた評価より，プロセッサ性能に与える影響を議論する．

† 東京大学大学院工学系研究科
Graduate School of Engineering, The University of Tokyo
☆ 現在，電気通信大学大学院情報システム学研究科
Presently with Graduate School of Information Systems, The University of Electro-Communications

以下に本稿の構成を示す．次章で関連研究をまとめ，3章で2レベル・ストライド値予測機構を提案する．4章では評価環境をまとめ，5章では，提案する値予測機構のヒット率とミス率を評価する．6章では，提案する値予測機構がプロセッサ性能に与える影響を議論する．7章で全体のまとめを行い，今後の課題を述べる．

2. 関連研究

従来の研究は，実行レイテンシの長いロード命令に焦点を絞り，ロードするデータ値を予測する値予測機構[6),13)]と，演算結果を生成するすべての命令の結果の値を予測する値予測機構[1),3),5),8),10),11)]とに分類できる．本研究は後者の分類に入り，ストア，分岐，特殊命令（たとえば，DEC Alpha の PAL 命令）を除くすべての命令の演算結果を予測の対象とする．このように多くの命令を予測対象とした値予測機構には次のものがある．

2.1 Last-value 予測機構

Last-value 予測機構[5),6)]は，予測する命令の前回の演算結果を今回の予測値として利用する．文献5)では，予測ミスの回数を削減するために，動的に変化するカウンタからなる分類テーブル（Classification Table）を用いて予測精度の高い命令のみ予測する Last-value 予測機構を提案している．

2.2 ストライド値予測機構

ストライド・ベースの値予測機構は，予測する命令の過去2回の演算結果の差分 $Stride$ と，最も近い過去に得られた値 $Value$ から，$Value + Stride$ により予測値を計算する．ストライド・ベースの値予測機構を図 **1** に示す．値履歴テーブル（Value History Table: VHT）は前回の演算結果やストライド等を格納するテーブルである．値履歴テーブルの更新の仕方に選択の余地があるが，3つの状態を用いて予測ミスを削減する文献11)のアルゴリズムを説明する．

VHT のエントリは図1に示す Tag，Value，Stride，State（状態）の4つのフィールドを持つ．状態は，Init，Transient，Steady のどれかの値を持つ．

状態の変化と予測アルゴリズムを図 **2** に示す．最初に値を生成する命令に出会ったときには予測を行わない．値が生成されたときに VHT のエントリが割り当てられ，(i) 演算結果を Value フィールドに格納し，(ii) 状態を Init に変更する．

状態が Init だった場合には予測をしない．しかし，その命令が値（D1）を生成した場合にはストライドの開始となった可能性があるので，(i) ストライドを $S1 = D1 - Value$ より計算し，(ii) D1 を Value フィールドに，S1 を Stride フィールドに格納し，(iii) 状態を Transient に変更する．

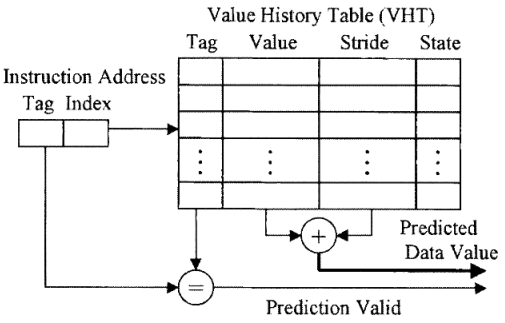

図 1 ストライド値予測機構のブロック図
Fig. 1 Block diagram of stride value predictor.

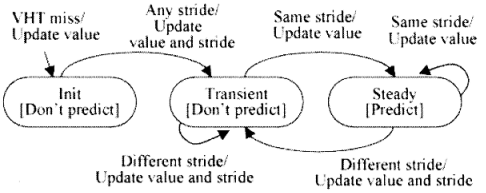

図 2 ストライド値予測における状態遷移
Fig. 2 State transition of stride value predictor.

状態が Transient のときに命令の別のインスタンスが実行されたときも予測を行わない．そのインスタンスが値（D2）を生成した場合には，(i) ストライドを $S2 = D2 - Value$ で計算し，(ii) D2 をエントリの Value フィールドに格納し，(iii) もし S2 の値が前のストライドと等しかった場合には，状態を Steady に変更する．ストライドが等しくなければ S2 を Stride フィールドに格納する．

状態が Steady のときには Stride と Value を用いて値を予測する．計算したストライドが前回のストライドと等しい場合には，Value フィールドを更新する．もし異なったストライドが得られたときには，状態を Transient に変更するとともに Value，Stride の値を更新する．

2.3 グローバル・コンテキストを利用した値予測機構

文献1)，8)では演算結果を予測する命令以外の命令の挙動（グローバル・コンテキスト）を利用する値予測機構を議論している．グローバル・コンテキストとして，グローバルな分岐履歴を利用する値予測機構を次にまとめる．

Per-path stride per-path value（PS-PLV）予測機

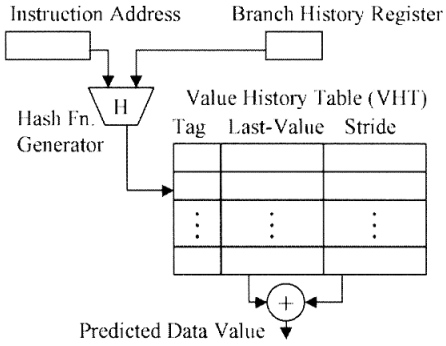

図 3 Per-path stride per-path value (PS-PLV) 予測機構のブロック図
Fig. 3 Block diagram of per-path stride per-path value predictor.

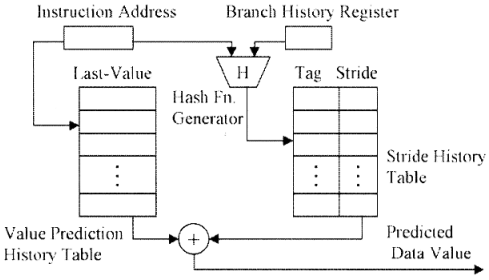

図 4 Per-path stride (PS) 予測機構のブロック図
Fig. 4 Block diagram of per-path stride predictor.

構[8]はストライド・ベースの値予測機構で，命令アドレスと分岐履歴レジスタを用いて値履歴テーブルのインデックスを生成する．これにより，実行パスごとに異なった last-value とストライドを利用した予測が可能となる．PS-PLV 予測機構のブロック図を図 **3** に示す．ただし，テーブル更新のためのいくつかのフィールドを値履歴テーブルから省略した．

Per-path stride (PS) 予測機構[8]はストライド・ベースの値予測機構で，last-value とストライドを別々のテーブルに保存し，ストライドを選択するために分岐履歴を利用する．これにより，実行パスごとに異なったストライドを利用した予測が可能となる．図 **4** に PS 予測機構のブロック図を示す．ただし，テーブル更新のためのいくつかのフィールドを図から省略した．

文献 8) では，PS-PLV 予測機構と PS 予測機構の予測成功率を評価しており，ストライド値予測機構と比較してそれぞれ 6.9%と 8.4%の予測成功率の向上を確認している．

文献 1) では，命令アドレスと分岐履歴レジスタに加えて，予測を行う命令の過去の実行結果 (Last-value) を利用してテーブルのインデックスを作成する 2 レベルの値予測機構を検討している．この値予測機構のブロック図を図 **5** に示す．

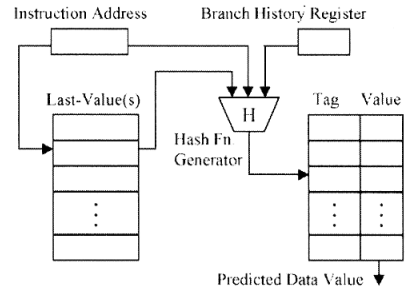

図 5 Last-value と分岐履歴を利用する 2 レベル値予測機構
Fig. 5 Two level value predictor with branch history and last-value.

2.4 その他の値予測機構

文献 11) では，動的に生成される値の多くは最近のユニークな 4 個以内の値であるという結果に注目し，パターン・ベースの 2 レベル値予測機構を提案している．この 2 レベル値予測機構では，VHT に 4 つの異なった値を格納しておき，次の 2 段階で値を予測する．最初のレベルでは，予測するインスタンスの命令アドレスを用いて VHT のエントリを引き，そこに格納されている 4 つの値を出力する．2 レベル目で，4 つの中から 1 つの値を選択することで予測値を得る．2 レベル目の選択は，過去の p 回の予測結果とカウンタを用いて決定するが詳細は省略する．

文献 11) では，Last-value 予測とストライド値予測のハイブリッド値予測機構，ストライド値予測と 2 レベル値予測のハイブリッド値予測機構を提案評価しており，複数の予測アルゴリズムを利用することで予測範囲が広がることを確認している．

文献 10) では，過去の連続した有限個の値の履歴 (コンテキスト) と過去の実行履歴を比較し，高い確率で実行されるパターンに基づいて予測を行うコンテキスト・ベースの値予測機構を提案している．この値予測機構では，次のように $a\ a\ a\ b\ c\ a\ a\ a\ b\ c\ a\ a\ a$ と値が変化したとき (a, b, c は 32 ビットまたは 64 ビットの値を表すシンボル)，実行確率の高いパターン $a\ a\ a\ b$ とコンテキスト $a\ a\ a$ の比較により値 b を予測する．文献 10) ではコンテキスト・ベースの値予測機構がストライド値予測機構や Last-value 予測機構より高い予測成功率を達成する可能性があるとしているが，ハードウェア量や実装に関する議論は今後の課題としている．

2.5 予測機構のカスケード接続

文献 9), 14) において，値予測とその他の予測機構

表1 全実行命令に対する予測ヒット率とミス率
Table 1 The ratio of prediction hit and miss to the executed instructions.

	LV+CT	Stride	Two-level	Hybrid
Hit	21.7%	29.8%	29.4%	39.9%
Miss	1.7%	2.9%	3.9%	5.9%

との組合せが議論されている．文献14)では，値予測が利用できなかったロード命令に関してはデータアドレスの予測を用いて投機的に処理を進める協調型予測器を議論している．文献9)では，固定された順序により，値予測，メモリリネーミング，データアドレス予測，依存関係予測の4つの投機技術を利用する協調型予測器を議論している．本稿で議論するグローバルな分岐履歴を利用する値予測機構は，複数の予測機構の利用という意味で協調型予測器といえるが，たとえば，分岐予測の確信度により値予測の動作を変化させるといったより密接な協調は今後の課題といえる．

2.6 値予測機構とプロセッサの性能向上

本稿で用いる値予測のヒット率とミス率という言葉を定義する．値予測機構が正しく値を予測した回数を実行命令数で割った値としてヒット率を定義する．値予測機構が間違った値を予測した回数を実行命令数で割った値としてミス率を定義する．ヒット率だけでなくミス率を示す理由は，予測を行わない命令が存在するためであり，これらのヒット率とミス率を足し合わせた割合が全実行命令に対して値予測を行った割合となる．

分類表を用いたLast-value予測（LV+CT），ストライド値予測（Stride），2レベル値予測（Two-level），ストライドと2レベルのハイブリッド値予測（Hybrid）のヒット率とミス率を**表1**にまとめる．これらの値は文献3)のデータを用いて計算した．表1から，複雑な値予測機構を用いることでヒット率とミス率がともに増加していくことが分かる．ハイブリッド値予測機構を用いた場合には，全実行命令の約40%の演算結果を正しく予測できる反面，平均で17命令実行するたびに1回の値予測ミス（ミス率5.9%）が発生することになる．

表1に示したヒット率とミス率は値予測機構を評価する際の重要な情報となるが，値予測機構はハードウェア・コストとプロセッサ性能の向上率を用いて評価しなければならない．プロセッサの性能向上は，予測のヒット率，ミス率，ヒットした場合の利得，ミスしたときのペナルティに依存するので，値予測を用いてプロセッサの性能を向上させるためには，性能向上につながる命令の演算結果を正しく予測するとともに，予測ミスの回数を削減し，予測ミスのペナルティを削減することが重要となる．

3. 2レベル・ストライド値予測の提案

本章では，ストライド値予測機構の予測ミスと，予測する命令までの条件分岐結果（分岐履歴レジスタ）の関係を示した後に，分岐履歴レジスタを用いてストライド値予測機構を拡張する2レベル・ストライド値予測機構を提案する．

従来から，分岐予測において分岐履歴レジスタを利用した2レベルの予測機構が提案されている[7]．また，近年，分岐履歴レジスタを用いた値予測が文献1)，8)，12)，15)において検討されている．本章で提案する2レベル・ストライド値予測は文献12)を基本としている．2.3節で述べた文献1)，8)の値予測と本章で提案する2レベル・ストライド値予測は，独立に提案された，分岐履歴レジスタを利用するという点が共通する方式の値予測であるが，本方式は文献1)，8)にない以下の特徴を持つ．文献1)，8)で検討されている値予測機構がヒット率の向上を目指しているのに対して，提案手法ではヒット率の向上とともにミス率の削減を目的とする．手法として，ストライド値予測で予測ミスが発生した際に，予測ミスを引き起こした命令の演算結果とその命令に至るグローバルな分岐履歴をテーブルに保存し，この情報を用いてストライド値予測で予測ミスとなる領域を正しく予測する方式を提案する．

3.1 値予測ミスと分岐履歴レジスタ

ストライド・ベースの値予測では，演算結果が一定間隔で変化を続ける場合に予測が成功する．しかし，典型的なループ構造（for文）に見られるように，実際のプログラムでは，演算結果が一定の間隔で変化を続けるわけではなく，なんらかの条件により，値が初期化されることが多いと考えられる．2.2節で示したストライド値予測機構を用いて1〜5の値を繰り返す例を予測した結果を図**6**に示す．図の下段は予測結果NP（No Predict），H（Hit），M（Miss）を表している．この例の場合，値が1に初期化される際に予測ミスが発生する．また，値が初期化されるまでの間隔が短くなるに従って正しく予測できる割合が減少する．もし，この初期化のタイミングと初期値を正しく予測できたとすれば，1〜5の値を繰り返す例のように，一定間隔でストライドが変化する場合に，100%に近い予測成功率を達成できる．

我々は，初期化のタイミングを予測するために，予測する命令に至るまでのグローバルな条件分岐命令の

```
Value Sequence:1  2  3  4 5 1 2  3  4 5 1...
               NP NP NP H H M NP NP H H M
```

図6 定期的にストライドが変化する場合の予測結果
Fig. 6 Prediction result of the value sequence where the stride differs periodically.

```
for(i=0; i<10; i++)
  for(j=0; j<5; j++) sum += j;

(1)   mov 0,%o1         ; sum=0
(2)   mov 9,%o2         ; i=9
(3)   mov 0,%o0         ; j=0
   .LL13:
(4)   add %o1,%o0,%o1   ; sum+=j
   .LL12:
(5)   add %o0,1,%o0     ; j++, Predict this!
(6)   cmp %o0,4         ; j<4 ?
(7)   ble,a .LL12       ; conditional branch
(8)   add %o1,%o0,%o1   ; sum+=j,delayed slot

(9)   addcc %o2,-1,%o2 ; i--
(10)  bpos,a .LL13      ; conditional branch
(11)  mov 0,%o0         ; delayed slot, j=0
```

図7 分岐履歴を用いることで予測ミスを改善できる例
Fig. 7 Example that removes prediction miss with the branch history.

履歴に注目した．すなわち，予測値が一定間隔で変化しているときと，初期化されるときで異なった制御の流れをとり，この制御流の変化を検出することで初期化のタイミングを予測できるという仮定により予測を行うことを提案する．この仮定を図**7**の例を用いて説明する．

図7上のソースコードにおいて，インナーループの誘導変数jの値予測を考える．SPARCアーキテクチャをターゲットとして，ソースコードをコンパイルした結果を図7下に示す．%o0，%o1，%o2はレジスタを表し，それぞれの命令の最後のレジスタがデスティネーションのレジスタを示している．ただし，分岐命令で利用するフラグを設定する(6) cmpと分岐命令にはデスティネーションのレジスタは存在しない．命令(5)がソースのj++に対応し，この命令が生成する値の予測を検討する．命令(5)は図6に示したように1，2，3，4，5という値を繰り返し生成する．また，このプログラムを実行したときの条件分岐命令（命令7と10）の結果は分岐成立を T，不成立を N で表現すると，$TTTTTNT\ \ TTTTTNT\ldots$ というビット列となり，予測する値が1に初期化される時点までの分岐結果は $\ldots NT$ となっていることが分かる．このコードの場合には値予測の際に最近の条件分岐の結果2ビットからなるビット・パターン（2ビットの分岐履歴レジスタ）が NT だった場合に値1を予測することで予測ミスを回避できる．このように，制御流の変化を検出することで初期化のタイミングを予測できるという仮定により2レベル・ストライド値予測機構を提案する．

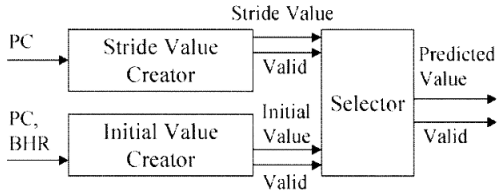

図8 2レベル・ストライド値予測機構のブロック図
Fig. 8 Block diagram of two-level stride value predictor.

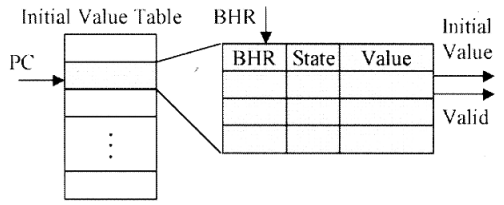

図9 4ウェイの初期値テーブル
Fig. 9 Initial value table of 4-way set associative.

3.2 2レベル・ストライド値予測機構

2レベル・ストライド値予測機構は，図**8**に示すように，ストライド値生成部，初期値生成部，セレクタ部の3つの要素からなる．ストライド値生成部は，図1に示したストライド値予測機構を拡張したもので，ストライドを用いて予測値を生成する．初期値生成部は，分岐履歴レジスタ（BHR）とプログラム・カウンタを用いて初期値を予測する．セレクタ部は，ストライド値生成部と初期値生成部の出力から，適切な予測値を選択するとともに，予測の確信度を評価する．この確信度が高いと評価された場合のみ予測を有効にすることで，予測ミスを削減する．

初期値生成部は，命令アドレスと分岐履歴レジスタを用いて初期値を出力する初期値テーブルからなる（図**9**）．初期値を得るために，まず，命令アドレスからインデックスを計算し，テーブル中のエントリを選択する．それぞれのエントリは，BHR，State，初期値，というフィールドを n 組持っている（n ウェイの初期値テーブルと呼ぶ）．次に，これらの n 組の中から，分岐履歴レジスタが一致するウェイを検索し，一致した場合に初期値を出力する．ただし，初期値の予測成功率を上げるために，それぞれのウェイには前回の予測成功・失敗を保存するStateというフィールドを用意し，前回の予測が失敗した場合には，予測を無効にする．

セレクタ部は，まず，ストライド値生成部，初期値生成部が生成した値から適切な値を選択する．もし，ストライド値生成部のみが値を生成した場合には，ストライド値を予測値とする．もし，両方が値を生成した場合には，初期値を予測値とする．それ以外の場合には，予測を無効とする（初期値生成部のみ値を生成した場合には予測が無効となる点に注意）．次に，セレクタ部は予測の確信度評価を用いた絞り込みを行う．予測ミスを抑えるために，予測する命令の過去の履歴が，ある値（閾値）以上連続して成功している場合にのみ予測を有効とする．

次に，状態の更新アルゴリズムを説明する．これらの状態更新はすべて命令のリタイア時に行われる．

ストライド値生成部は，次の変更点を除いて，2.2節で説明したアルゴリズムによりテーブルの更新と予測を行う．ストライド値生成部の予測がミスした場合でも，初期値生成部が行った予測がヒットした場合には状態を Transient に変更せず，Steady のままで予測を継続するように変更する．

初期値生成部は，ストライド値生成部が予測を行い，かつストライド値生成部の予測がミスした場合に，計算により得られた結果を初期値として初期値テーブルに登録する．このとき，初期値テーブルのすべてのウェイが有効だった場合には，LRU でウェイを入れ替える．また，初期値生成部の予測が有効だった場合には，演算結果と初期値生成部の予測値とを比較し，初期値テーブルの State フィールドを更新する．

セレクタ部で確信度の評価を利用する場合には，予測値と演算結果を比較し，それぞれの命令ごとに，連続して予測が成功した回数を専用のテーブルに保存する．

4. 評価環境

4.1 ベンチマーク・プログラム

SPEC95 の整数系ベンチマークを用いて 2 レベルストライド値予測機構を評価する．ベンチマーク・プログラムと本評価で用いる入力セット，実行命令数，値予測の対象となる命令数（割合）を**表 2** にまとめる．それぞれのベンチマーク・プログラムの入力セットは実行命令数が 2 億程度に収まるように調整した．これらのプログラムは Alpha AXP アーキテクチャをターゲットとし，DEC C コンパイラ（最適化オプション -O4）を用いてコンパイルした．値予測の対象とならない命令は分岐，ストア，PAL 命令であり，それ以外のすべての命令は値予測の対象となる．予測の対象となる命令は，全実行命令の 65〜88%と非常に高い．

表 2 SPECint95 のベンチマークプログラム
Table 2 Benchmark from SPECint95.

Program	Input Set	Executed Instr.	Predictable Instr.
go	9 9	139M	109M (78.6%)
m88ksim	train/ctl.in	127M	97M (75.9%)
gcc	genrecog.i	152M	107M (70.2%)
compress	30000 q 2131	142M	112M (78.8%)
li	train.lsp	208M	137M (65.8%)
ijpeg	specmun.ppm	172M	152M (88.2%)
perl	admits, 1/5 dictionary	154M	108M (70.3%)
vortex	train/vortex.in	185M	132M (70.8%)

4.2 プロセッサ・モデル

予測する時刻とテーブルを更新する時刻とのサイクル数のずれにより，予測機構のヒット率とミス率が変化する．このため，評価の際には用いるプロセッサ・モデルに注意する必要がある．5 章の評価では，この問題を回避するために，スカラ・プロセッサのシミュレータを用いて 2 レベル・ストライド値予測機構のヒット率とミス率を評価する．6 章において，プロセッサ性能を測定する際には，標準的なアウトオブオーダ実行のプロセッサ・モデルを用いて評価する．評価に用いるスーパースカラ・プロセッサの詳細は 6.1 節で説明する．

5. ヒット率とミス率の評価

スカラ・プロセッサのシミュレータを用いて，2 レベル・ストライド値予測機構のヒット率とミス率を測定する．ここでは，初期値テーブルのウェイ数，初期値生成部における分岐履歴レジスタのビット数，セレクタ部において確信度を評価する際の閾値（確信度評価の閾値と呼ぶ），という 3 つのパラメータを変化させ，2 レベル・ストライド値予測機構のヒット率とミス率を評価する．測定結果で示すヒット率とミス率は，8 つのベンチマークプログラムの調和平均を用いる．以降の評価では，予測する命令間の競合を避けるために，初期値テーブルのエントリ数，ストライド値生成部における値履歴テーブルのエントリ数を非常に大きな値（64 K）に設定して評価する．

初期値生成部における分岐履歴レジスタのビット数を 8，16，24，32 ビットに，初期値テーブルのウェイ数を 1，2，4，8，16，32，64 に設定した場合のヒット率とミス率を**表 3** に示す．ここでは，セレクタ部における確信度評価を利用していない．

表 3 の結果から，初期値テーブルのウェイ数を増やすことでヒット率が向上することを確認できる．また，分岐履歴レジスタのビット数を増やすことで，ミ

表3 2レベル・ストライド値予測機構のヒット率とミス率
Table 3 Hit and miss ratio of two-level stride value predictor.

	1 way	2 way	4 way	8 way	16 way	32 way	64 way
8 bit	38.9%,2.48%	39.7%,2.40%	40.4%,2.37%	41.1%,2.31%	41.5%,2.29%	41.6%, 2.28%	41.6%,2.28%
16 bit	38.9%,2.39%	39.7%,2.30%	40.6%,2.17%	41.6%,2.04%	42.4%,1.93%	43.0%, 1.89%	43.4%,1.88%
24 bit	38.9%,2.36%	39.5%,2.26%	40.4%,2.13%	41.4%,2.00%	42.3%,1.84%	43.0%, 1.77%	43.5%,1.74%
32 bit	38.9%,2.32%	39.4%,2.23%	40.2%,2.12%	41.2%,1.95%	42.1%,1.77%	42.8%, 1.67%	43.4%,1.63%

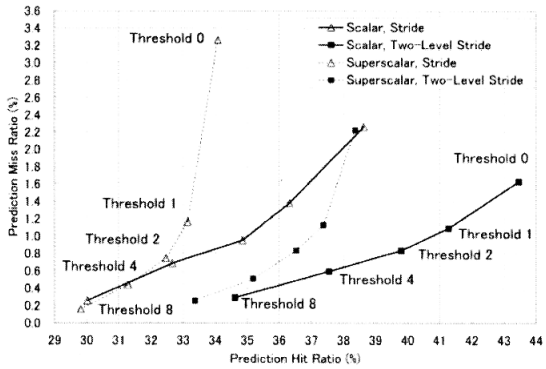

図10 確信度評価の閾値を変化させたときのヒット率とミス率
Fig. 10 Hit and miss ratio with different threshold of the confidence estimator.

ス率が低下することを確認できる．初期値テーブルのウェイ数を固定させた場合，分岐履歴レジスタのビット数を大きくすることでヒット率がわずかに低下する．これは，分岐履歴レジスタのビット数を大きくすることにより，ウェイの競合が深刻になるためと考えられる．表3の評価においては，初期値テーブルのウェイ数64，分岐履歴レジスタのビット数32とした設定が，ヒット率，ミス率ともに良い結果を出した．以降，この設定を用いて評価を続けていく．

3つ目のパラメータ，確信度評価の閾値を0，1，2，4，8に変化させた場合のヒット率とミス率を図**10**にまとめる．確信度評価の閾値0という設定は，確信度評価を用いないことを意味している．初期値テーブルのウェイ数64，分岐履歴レジスタ32ビットに設定した2レベル・ストライド値予測機構について評価した．比較のために，同じ条件で確信度評価を用いた場合のストライド値予測機構の評価結果を加えてある．図10には，スーパースカラ・モデルを用いて測定した結果を破線で示しているが，こちらに関しては6.2節で議論する．ここでは，実線で示したスカラ・モデルの測定結果を検討する．

値予測機構は，確信度評価の閾値で指定した回数以上連続して予測が成功している命令の予測を有効とする．このため，確信度評価の閾値を大きくすることは，値予測を行う割合の低下につながり，図10に示したヒット率とミス率の低下を引き起こす．図10の結果から，同じ閾値を用いた場合には，2レベル・ストライド値予測機構の方が4.5〜4.9%ヒット率が高いことが分かる．また，閾値8のとき微妙にミス率が増加している点を除いて，2レベル・ストライド値予測機構の方がミス率に関しても良い結果を出している．閾値が小さいときに，2レベル・ストライドによるミス率改善が顕著になり，閾値0のときには0.6%の予測ミスを削減できている．表1に示したように，複雑な値予測機構を用いた場合にはヒット率とともにミス率が増加する傾向にあるが，2レベル・ストライド値予測機構はヒット率，ミス率ともに改善できるという特徴がある．

6. プロセッサ性能に与える影響

2レベル・ストライド値予測機構をスーパースカラ・アーキテクチャに組み込む場合には，ストライド値生成部の構成を一部変更する必要がある．予測値を履歴テーブルから得られる $Value + Stride$ で計算する代わりに，予測する命令と同一の命令がリオーダバッファ内に存在する場合には，リオーダバッファから $Value$（演算結果または予測値）を出力するように変更する．この拡張を行わない場合には，履歴テーブルが更新される前に，同じアドレスの命令が複数フェッチされた場合に，それらの予測値が同じ値となり，予測のヒット率が著しく低下する．

6.1 スーパースカラ・プロセッサ

クロック単位で動作する実行ベースのシミュレータを用いて，スーパースカラ・プロセッサにおける値予測機構のヒット率，ミス率と，2レベル・ストライド値予測機構を用いた場合の並列性を測定する．評価に用いるプロセッサの設定を**表4**に示す．これらのパラメータは数年先のプロセッサを想定して設定した．分岐予測に関しては，条件分岐命令のためにハイブリッド分岐予測，レジスタ間接分岐命令のために過去のレジスタ間接分岐命令の履歴を利用するパス・ベースのBTB[2]を利用した．また，機能ユニット数，フォワーディングバス，結果バス，レジスタファイルのポート数，キャッシュのポート数というハードウェアに関し

表4 スーパースカラ・プロセッサの設定
Table 4 Superscalar processor configuration.

Fetch, Retire Width	16 instructions
Branch Predictor	Hybrid of gshare (20) and 2BC, Path-based BTB, 64 entry RAS
Data Cache	2-way set associative, Line 64B, 128 KB, 20 cycle miss penalty
Instruction Window	128 entry
I-Cache, L2 Cache	Ideal

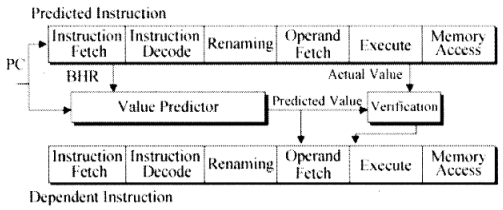

図11 必要としている命令への予測値の供給
Fig. 11 The predicted data is supplied to the destination instruction.

ては競合が発生しない十分な資源を確保できると想定した.

値予測に失敗した場合には,間違った値を利用した命令の再実行が必要となる.本評価では,値予測にミスした場合,正しいオペランドを用いて予測ミスした命令の後続命令を実行ステージから再処理することで予測ミスから回復する.このために,命令が発行された後にも,リタイアするまでは,リザベーション・ステーションのエントリを解放しない.予測ミスの場合にはリザベーション・ステーションに格納されている情報を用いて,命令の再実行が可能となる.文献5)で議論されているように,予測ミスした命令にデータ依存関係を持つ命令だけを実行ステージから再処理することで予測ミスから回復することができるが,本実装では,予測ミスした命令の後続命令をすべて再実行するとした.

図11にシミュレータのパイプライン構成と,値生成と結果比較のタイミングを示す.予測値を生成する命令は,命令フェッチからオペランド・フェッチ・ステージまでの4サイクルの間に予測値を生成すればよい.値生成のための処理をパイプライン化することで,プロセッサのサイクル時間に大きな影響を与えることなく予測値を生成できると考えている.本評価では,予測値と演算結果の比較を実行ステージ(ロード命令の場合にはメモリアクセス・ステージ)で行うとした.

以降の評価で示す命令レベル並列性は,実行命令数をシミュレータで測定した実行サイクル数で割ることで計算する.

6.2 スーパースカラにおける値予測機構のヒット率

プロセッサの性能を見る前に,スーパースカラ・プロセッサにおける値予測機構のヒット率とミス率を確認しておく必要がある.6.1節で設定したスーパースカラ・プロセッサを用いて測定したヒット率とミス率を,図10の破線に示す.ストライド値予測機構,2レベル・ストライド値予測機構ともに,スカラ・プロセッサにおける評価結果からの性能低下を確認できる.特に,閾値が0のときに性能低下が大きく,ストライド値予測の場合でヒット率が4.5%,2レベル・ストライド値予測の場合でヒット率が5.0%低下する.本評価では,命令がリタイアするときに,値予測で必要となる履歴テーブルの更新を行っている.このために,予測を行う時刻とテーブルを更新する時刻とのずれが生じ,この性能低下を引き起こしている.テーブル更新のタイミングによる性能低下は,テーブル更新の一部を投機的に更新するといった改良で削減できる可能性がある.これらの改良は,今後の課題である.次に示す並列性は,図10の破線で示したヒット率,ミス率における並列性である.

6.3 値予測による命令レベル並列性の変化

値予測機構による命令レベル並列性の変化を図12に示す.図12では,8本のプログラムとそれらの調和平均について,それぞれ11個の値を示している.それらは,左から,値予測を用いなかった場合の並列性(ベースライン),次の5本がストライド値予測機構を用いた場合の並列性で,確信度評価の閾値を0, 1, 2, 4, 8と変化させたときの並列性をそれぞれ示している.最後の5本が2レベル・ストライド値予測機構を用いて確信度評価の閾値を0, 1, 2, 4, 8と変化させたときの並列性である.

図12の結果を検討する.まず,m88ksim, vortexにおいて値予測により大幅な性能向上が得られることが分かる.ただし,これらのプログラムは,値予測を用いなかった場合にも高い並列性を示しており,これらの並列性のさらなる向上が調和平均ではそれほど大きな性能向上につながらない点に注意する必要がある.m88ksim, vortex以外のプログラムでは,確信度評価の閾値により性能が低下する場合がある.特に,閾値0で確信度評価を利用しない場合には,ストライド値予測機構,2レベル・ストライド値予測機構ともにいくつかのプログラムで性能低下が起こっている.平均で見た場合にも,ストライド値予測機構で閾値0の場合にはベースラインより性能が低下する.

値予測による性能向上が著しいプログラムには,値予測のヒット率と分岐予測の成功率が高いという特徴

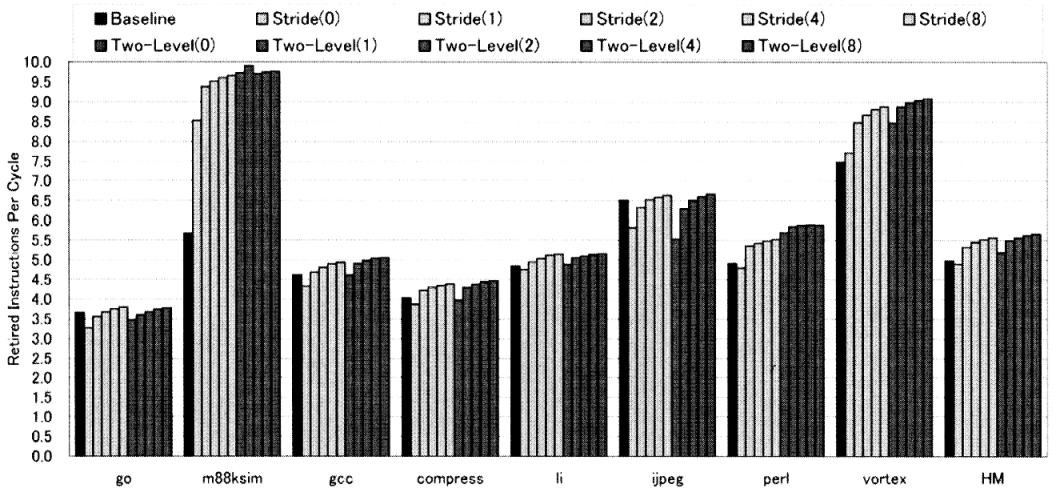

図 12 ストライド値予測機構と 2 レベル・ストライド値予測機構による命令レベル並列性の変化
Fig. 12 ILP with stride or two-level stride value predictor.

がある．2 レベル・ストライド値予測で確信度評価の閾値 8 の設定におけるデータを用いて議論するが，最も著しい並列性向上を示した m88ksim の場合には，値予測のヒット率は 67.7%と平均の 34.6%と比較して非常に高い．また，分岐予測に関しても成功率は 98.4%と平均成功率 95.2%に対して非常に高い．m88ksim に次いで並列性向上の大きい vortex の場合には，値予測のヒット率は 39.7%と平均より 5%高く，分岐予測に関しても成功率 99.2%と非常に高い．これらは，値予測により著しい性能向上をもたらすプログラムの傾向である．より詳細な分岐予測と値予測の相関関係の検討は今後の課題である．

プロセッサ性能と確信度評価の閾値の関係を見てみると，m88ksim の一部を除いて，確信度評価の閾値を大きくした方が性能が高いことが分かる．この傾向は，ストライド値予測機構，2 レベル・ストライド値予測機構ともに確認できる．この結果は，ヒット率の増加以上に，ミス率の改善が性能向上につながることを意味している．ただし，確信度評価の最適な閾値は，予測のヒット率，ミス率，ヒットした場合の利得，ミスしたときのペナルティにより決まるので，予測ミスによるペナルティを小さくすることができれば，閾値を小さくした場合でも，高い性能を達成できる可能性がある．

最後に，ストライド値予測機構と，2 レベル・ストライド値予測機構とを比較する．調和平均で見た場合には，すべての閾値で，2 レベル・ストライド値予測機構を用いた方が性能が高い．しかし，ストライド値予測機構との性能差は，命令レベル並列性で見て，確信度評価の閾値 0 のとき 0.41，閾値 8 のとき 0.11 とそれほど高くない．閾値が 8 のときに最も高い並列性を得ているが，このときのベースラインからの向上率で比較すると，ストライド値予測機構による性能向上率が 12%，2 レベル・ストライド値予測機構による性能向上率が 14%と，その差は 2%という結果になった．図 10 に示したように，2 レベル・ストライド値予測機構によるヒット率とミス率の改善は，確信度評価の閾値を小さくした場合に大きく現れる．さらに，スーパースカラ・モデルを用いることで生じるテーブルの更新タイミングによる低下が，値予測による利得を削減している．予測ミスからの回復機構と履歴テーブルの更新手法の改良により，2 レベル・ストライド値予測機構による性能向上率をさらに引き上げることができると考えている．

7. まとめと今後の課題

予測する命令までの制御流の変化を利用する 2 レベル・ストライド値予測機構を提案し，その可能性を議論した．2 レベル・ストライド値予測機構における 3 つのパラメータ，初期値テーブルのウェイ数，初期値生成部における分岐履歴レジスタのビット数，セレクタ部における確信度評価の閾値を変化させ予測ヒット率とミス率を測定した結果，確信度評価の閾値 0 のときに最も高いヒット率 43.4%を達成した．これはストライド値予測機構より 4.8%高いヒット率となる．また，閾値が 0 のとき，ヒット率だけでなくミス率に関しても 0.6%の改善を確認した．

スーパースカラ・プロセッサのシミュレータを用い

た評価より，2レベル・ストライド値予測機構がプロセッサの性能に与える影響を検討した．並列性の向上率では，値予測機構における確信度評価の閾値を8とした場合に最も高い性能向上を示し，ストライド値予測機構を用いることで12%，2レベル・ストライド値予測機構を用いることで14%の性能向上を確認した．本評価で用いたスーパースカラ・プロセッサでは，2レベル・ストライド値予測機構の有効性を示すには至らなかった．ただし，2レベル・ストライド値予測機構によるヒット率とミス率の改善は，確信度評価の閾値を小さくした場合に大きく現れる．さらなる予測ミスペナルティの削減と最適な確信度評価の閾値の検討は今後の課題である．

本評価では，予測する命令間の競合を避けるために初期値テーブルのエントリ数，ストライド値生成部における値履歴テーブルのエントリ数を64Kと非常に大きな値に設定した．また，初期値テーブルのウェイ数に関しても64ウェイと大きな値を想定して，2レベル・ストライド値予測機構の可能性を検討した．ストライドや初期値を格納するフィールドのビット数，テーブルのエントリ数とテーブルの構成方式を含め，必要となるハードウェア量と性能の関係を議論する必要がある．ハードウェア量に関しては，予測ミスした場合の回復機構で必要となるハードウェア量も同時に検討する必要がある．これらの検討は今後の課題である．

文献9)，14)で議論されているように，値予測と分岐予測に加えて様々な予測機構がプロセッサ上に実装されていく可能性がある．これらの複数予測機構によるカスケード方式の有効性に関する検討は今後の課題である．

謝辞 本研究は我々研究室内の大規模データパス・プロジェクトの一部として行われたものであり，多くの貴重なご意見をくださったプロジェクトメンバ，ならびに査読者の方々に感謝いたします．本プロジェクトの一部は（株）半導体理工学研究センターとの共同研究による．

参考文献

1) Codrescu, L. and Wills, S.: Improving Value Prediction Accuracy with Global Correlation, Technical report, School of Electrical and Computer Engineering, Georgia Institute of Technology (1999).
2) Driesen, K. and Holzle, U.: Accurate Indirect Branch Prediction, *Proc. International Symposium on Computer Architecture*, pp.167–178 (1998).
3) Gonzalez, J. and Gonzalez, A.: The Potential of Data Value Speculation to Boost ILP, *Proc. International Conference on Supercomputing*, pp.221–228 (1998).
4) Hennessy, J.L. and Patterson, D.A.: *Computer Architecture a Quantitative Approach*, Morgan Kaufman Publishers (1995).
5) Lipasti, M.H. and Shen, J.P.: Exceeding the Dataflow Limit via Value Prediction, *Proc. 29th Annual International Symposium on Microarchitecture*, pp.227–237 (1997).
6) Lipasti, M.H., Wilkerson, C.B. and Shen, J.P.: Value Locality and Load Value Prediction, *Proc. International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pp.138–147 (1996).
7) McFarling, S.: Combining Branch Predictors, Technical Report, TN-36, Compaq Computer Corp. Western Research Laboratoty (1993).
8) Nakra, T., Gupta, R. and Soffa, M.L.: Global Context-Based Value Prediction, *5th International Symposium on High-Performance Computer Architecture*, pp.4–12 (1999).
9) Reinman, G. and Calder, B.: Predictive Techniques for Aggresive Load Speculation, *Proc. 32th Annual International Symposium on Microarchitecture*, pp.4–12 (1999).
10) Sazeides, Y. and Smith, J.E.: The Predictability of Data Values, *Proc. 29th Annual International Symposium on Microarchitecture*, pp.248–258 (1997).
11) Wang, K. and Franklin, M.: Highly Accurate Data Value Prediction using Hybrid Predictors, *Proc. 30th Annual International Symposium on Microarchitecture*, pp.281–290 (1997).
12) 吉瀬謙二，坂井修一，田中英彦：マルチレベル・ストライド値予測機構による命令レベル並列性の向上，並列処理シンポジウムJSPP'99論文集，pp.119–126 (1999).
13) 佐藤寿倫：アドレス名前替えによるロード命令の投機的実行，並列処理シンポジウムJSPP'98論文集，pp.15–22 (1998).
14) 佐藤寿倫：データ値予測とアドレス予測を組み合わせたデータ投機実行，並列処理シンポジウムJSPP'99論文集，pp.111–118 (1999).
15) 小池汎平，山名早人，山口喜教：投機的制御/データ依存グラフとJava Jog-time Analyzer，情報処理学会論文誌，Vol.40, No.SIG1 (PRO 2), pp.32–41 (1999).

(平成11年8月30日受付)
(平成12年2月 4 日採録)

吉瀬 謙二（学生会員）

1972年生．1995年名古屋大学工学部電子工学科卒業．2000年東京大学大学院情報工学専攻博士課程修了．工学博士．同年，電気通信大学助手．計算機アーキテクチャの研究に従事．

坂井 修一（正会員）

1958年生．1981年東京大学理学部情報科学科卒業．1986年同大学院情報工学専門課程修了．工学博士．同年，電子技術総合研究所入所．1991年4月より1年間米国MIT招聘研究員．1993年3月より1996年2月までRWC超並列アーキテクチャ研究室室長．1996年10月より1998年3月まで筑波大学助教授(電子・情報工学系)．1998年4月より東京大学助教授（工学系研究科）．計算機システム一般，特にアーキテクチャ，並列処理，スケジューリング問題，マルチメディア等の研究に従事．情報処理学会論文賞（1990年度），日本IBM科学賞（1991年），市村学術賞（1995年），ICCD Outstanding Paper Award（1995年）等受賞．IEEE，ACM，電子情報通信学会会員．

田中 英彦（正会員）

1943年生．1965年東京大学工学部電子工学科卒業．1970年同大学院博士課程修了．工学博士．同年東京大学工学部講師．1971年助教授，1978〜1979年ニューヨーク市立大学客員教授，1987年東京大学教授現在に至る．計算機アーキテクチャ，並列処理，人工知能，自然言語処理，分散処理，CAD等に興味を持っている．「非ノイマンコンピュータ」，「情報通信システム」（著），「計算機アーキテクチャ」，「VLSIコンピュータI，II」，「ソフトウエア指向アーキテクチャ」（共著），New Generation Computing編集長，電子情報通信学会，人工知能学会，ソフトウェア科学会，IEEE，ACM各会員．