2パス限定投機システムPALSの評価環境 – 言語処理系 –

福田 明宏† 十鳥 弘泰† 大津 金光† 横田 隆史† 馬場 敬信†
†宇都宮大学大学院工学研究科情報システム科学専攻

1 はじめに

我々はスレッドレベル並列性を抽出する投機的マルチスレッド処理方式として2パス限定投機方式[1]を提案した．一般のアプリケーションに対して2パス限定投機方式がどの程度有効であるのかを検証するためには詳細な評価環境の構築が必要不可欠である．そこで我々は同方式をハードウェア上で実現するプロセッサシステムである2パス限定投機システムPALSの開発を行っている．本稿ではPALSの評価環境の構築のためPALS向けのマルチスレッドコードを生成する言語処理系について述べる．

2 2パス限定投機方式

ループはプログラム中で最も頻繁に実行されるコード領域であり，速度向上のためにはループを高速化することが肝要となっている．2パス限定投機方式はループの1イテレーションに対してスレッド割り当てを行い，マルチスレッド処理を行うことで高速化を達成する実行モデルである．2パス限定投機方式ではまず，プログラム中の各ループ内の実行経路(パス)のうち最も実行頻度の高いパス(#1パス)に特化したコードと2番目に実行頻度の高いパス(#2パス)に特化したコードを生成する．ここでパスに特化したコードとはプログラム中の分岐を一切排除し，そのパスに沿った基本ブロックのプログラムコードのみを抽出したものである．これらのコードを投機スレッドコードと呼ぶ．プログラム実行時には2つのパスのどちらが実行されるかを予測しながら投機的に実行する．予測は1イテレーションごとに行い，2つの投機スレッドコードから予測された方を1つのスレッドに割り当てる．これらの処理を複数のスレッドで投機的かつ並列的に行うことで高速化を達成する．

同方式では投機的にスレッド生成および処理を行うため予測が誤っている可能性がある．そこで予測の成否を判定するためにassert命令を用いる．assert命令は投機スレッドコードを生成する際に分岐命令を置き換える．最初の投機に失敗した場合，投機に失敗したスレッドとそのスレッド以降に投機実行しているスレッド全てを破棄し，回復処理を行う．そして投機に失敗したスレッドにはもう一方の投機スレッドコードを割り当て，実行する．後続のスレッドでは新たに予測とスレッドの割り当てを行う．2回目の投機にも失敗した場合は同様にスレッドの破棄と回復処理をしたのち元のループ構造を保持した逐次コード(非投機スレッドコード)を実行する．非投機スレッドコードを実行しているスレッド以降に実行するスレッドは非投機スレッドコードの実行終了を待つ．

3 パスプロファイリング

PALSの実行のためにはループ中の実行頻度の高い上位2本のパスについて特化したコードを用意しなければならない．そのためプログラム中のパスの実行頻度に対してのプロファイリング情報(パス情報)が必要となる．そこでプログラム中のパスの実行頻度を調べるツールであるパスプロファイラ[2]を用いる．

3.1 パス情報の形式

図1にパス情報の形式を示す．開始アドレスは対象パスを含むループの開始位置を表している．分岐履歴には分岐不成立を0，分岐成立を1として記録している．継続アドレスは分岐履歴のオーバーフローあるいは間接分岐により1つのパス情報ではパスを表せなくなった場合に次に続くパスの開始アドレスを格納したものである．パス情報フラグは5つのフラグから成り，分岐履歴のオーバーフローの発生を示すOフラグ，分岐履歴のオーバフローなどにより1つのパス情報ではパスを表せなくなった場合にそのパスがまだ継続することを示すCフラグ，関数呼び出しが発生したことを示すRフラグ，レジスタ間接分岐を示すIフラグ，システムコールの発生を示すSフラグがある．検出数はそのパスを何回実行したかを示している．

bt_start_addr 開始アドレス	bt_history 分岐履歴	bt_next_addr 継続アドレス	bt_info パス情報フラグ	count 検出数

図1: パス情報の形式

4 言語処理系

言語処理系はPALS向けのマルチスレッドコードを生成するソフトウェアシステムである．本システムではまず，先に説明したパスプロファイラからパス情報を受け取り，それにより実行頻度の高い上位2本のパスを決定する．そして実行頻度の高い上位2本のパス情報を用いて，投機スレッドコードを生成する．さらに2本のパスどちらも投機に失敗した場合に実行する非投機スレッドコードを生成する．

4.1 パス情報の適用問題

パス情報には開始アドレスや継続アドレスといったアドレス情報が含まれている．一方で，本システムがコード生成の対象としているアセンブリコードでは番地割り当てが行われていない．そのためアドレス情報を参照することができず，投機スレッドコードを生成することができない．

そこで逆アセンブルコードを用いて，これらのアドレス情報をアセンブリコードで参照できるように変換を行う．図2にアセンブリコードと逆アセンブルコー

An Evaluation Environment for Two-Path Limited Speculation System PALS – Code Transrater –
†Akihiro Fukuda, Hiroyoshi Jutori, Kanemitsu Ootsu, Takashi Yokota and Takanobu Baba
Department of Information Systems Science, Graduate School of Engineering, Utsunomiya University (†)

ドの対応を示す．逆アセンブルコードはオブジェクトコードからアセンブリソースレベルにまで変換するため，オブジェクトコードで割り当てられていたアドレスを保持している．逆アセンブルコードとパス情報からパスの開始位置を割り出し，アセンブリコード中のラベルに対応付けすることでパス情報に含まれるアドレス情報の変換を行う．

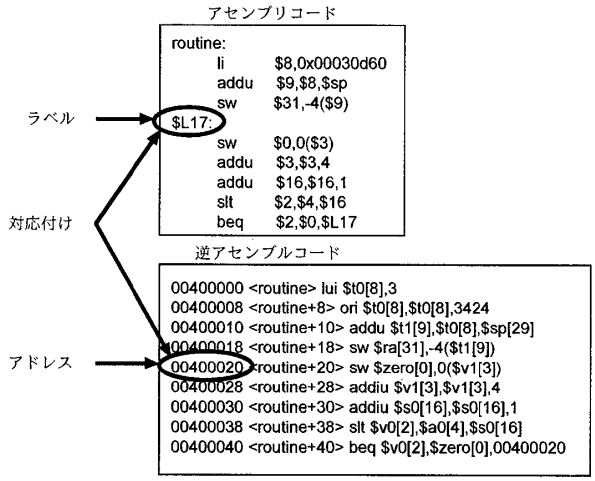

図 2: アセンブリコードと逆アセンブルコードの対応

4.2 スレッドコード生成

本システムではパス情報を元にアセンブリコードから投機スレッドコードを生成する．まず，パス情報の開始アドレスから対応づけられたラベルをコード生成対象ループの開始位置とする．そして，開始位置から順に命令を抽出していく．分岐命令を抽出した場合は分岐履歴に問い合わせ，分岐履歴が 0 であれば分岐不成立時に実行されるコードを抽出する対象とし，1 であれば分岐成立時に実行されるコードを抽出の対象とする．これらの処理を繰り返し，コード抽出対象がループの先頭の命令になった時はループの 1 イテレーションのパスのコードを抽出したことになるのでコード抽出を終了する．

非投機スレッドコードは元のループ構造を保持したものであるので元のコードをそのまま抽出する．

4.3 スレッド制御

2 パス限定投機方式はループイテレーションに対してのみスレッドを割り当て，マルチスレッド処理を行う．そのために PALS ではマルチスレッド処理開始や終了を告げる start2path 命令と stop2path 命令という特有の命令を定義している．本システムではマルチスレッド処理の対象ループに実行が移る直前に start2path 命令，直後に stop2path 命令を挿入し，PALS でのマルチスレッド処理を適切なタイミングで指示する．

PALS では拡張が容易という点から PISA[3] をベースとした命令セットを想定している．PISA の命令長である 64bit のうち未使用領域の 16bit の中に機能 bit として thread end bit を定義する．thread end bit はスレッドの終了を指示するために用い，命令に付加する．本システムでは投機スレッドコード，非投機スレッドコードにてループ先頭に戻る命令，すなわち 1 イテレーションの最後の命令のオペコードの後ろに/.end と記述することで thread end bit を付加する．

4.4 スレッド間通信

2 パス限定投機方式では複数のスレッドで並列に処理を行うためスレッド間通信が発生する．PALS では隣り合うスレッド間でレジスタの値の送受信を行うレジスタデータ通信を採用している．投機スレッドコードではパスが限定されており，分岐を考慮しなくてよいため送受信すべきレジスタを静的に決定できる．そのため本システムでは各投機スレッドコードで送受信すべきレジスタを明らかにし，レジスタデータ通信をサポートする．

レジスタの値送信は forward bit を使用して行う．forward bit は thread end bit と同様に命令長 64bit のうちの未使用領域に定義する．プログラム実行時に forward bit を付加した命令が実行されたらその命令で更新されるレジスタの値を後続のスレッドに送信する．本システムでは各投機スレッドコードにてレジスタの値が確定する命令のオペコードの後ろに/.fwd と記述することで forward bit を付加する．

レジスタの値受信は同期待ちすべきレジスタ群を示した bit mask により行う．本システムではパスごとにレジスタの依存解析を行い，スレッド間で依存関係になりうるレジスタを見つけ出し，bit mask を生成する．

5 おわりに

本稿では 2 パス限定投機方式を実現する 2 パス限定投機システム PALS の評価環境を構築するために PALS 向けのマルチスレッドコードを生成する言語処理系について述べた．

現在，本システムが簡易なプログラムから生成したコードを PALS のシステムシミュレータである pals にて正常に動作していることを確認した．今後は SPEC ベンチマークプログラムを始めとした様々なプログラムについても本システムの動作検証を行う．また，投機スレッドコードに命令スケジューリング等の最適化を施し，コードのチューニングによる高速化を目指していく予定である．

謝辞

本研究は，一部日本学術振興会科学研究費補助金 (基盤研究 (C)20500047，同 (C)21500049，同 (C)21500050) および宇都宮大学重点推進研究プロジェクトの援助による．

参考文献

[1] 横田 隆史ほか: “2 パス限定投機方式の提案”, 情報処理学会論文誌：コンピューティングシステム, Vol.46, No.SIG 16(AGS-12), pp.1-13, 2005.

[2] 増保 智久ほか: “動的最適化を支援する 2 レベルホットパス検出機構の設計”電子情報通信学会コンピュータシステム研究会 (CPSY)，信学技報, Vol.106, No.199, pp.13-18, (CPSY2006-15), 2006.

[3] Doug Burger, Todd M. Austin, “The SimpleScalar Tool Set, Version 2.0”. University of Wisconsin-Madison Computer Sciences Department Technical Report # 1324, 1997.6