5ZB-9

MediaBenchホットループ並列化のためのパスプロファイリング

増保 智久　道口 貴史　斎藤 盛幸　大津 金光　横田 隆史　馬場 敬信†
宇都宮大学工学部情報工学科‡

1　はじめに

我々はメタレベル最適化の概念に基づく計算機システムYAWARA[1]を提案している．本システムは，プログラムの振舞いや計算資源の動的な使用状況をもとに，最適な実行の仕方を自ら決定して自分自身に反映する能力をもたせ，システムの性能を上げることを目標としている．このためには，まずプログラムの振舞いを正確に把握することが重要となる．本研究はこの一環として，計算機の応用分野として重要なメディア処理に注目し，MediaBench[2]プログラム中で実行時間の多くを占めるホットループを対象としてそのパスの挙動を把握し，その結果をもとに最適な並列化戦略を検討する．なお，この研究で対象としているループとは，基本ブロック単位の制御フローで後方分岐している部分のことである．ループの末尾は後方分岐した基本ブロックとし，ループの先頭は分岐して飛んだ先の基本ブロックとしている．多重ループになっている場合は最内ループを対象としている．また，ここでのパスとは，ループの先頭から末尾へ至る制御フローに含まれる基本ブロック列のことである．

2　ホットループのパスプロファイリング

2.1　ホットループ検出

まず，ホットループ検出の手順としては，MediaBenchプログラムを配布ファイルで指定されたままの最適化オプションで，スレッドパイプラインモデルシミュレータSIMCA[3]用にコンパイルし，SIMCA上でプログラムカウンタを一定サイクル間隔でサンプリングしながら実行する．そして，ループに含まれるアドレスがサンプリングされた回数が多いループをホットループとして検出する．なお，サブルーチンコールを含んでいるループは，検出されてもパスプロファイリングが難しいためここでは対象にしない．

2.2　パスプロファイリング

2.1で検出されたループの中で，最もサンプリング回数が多いホットループを含む関数を対象として，基本ブロック単位の実行トレースログを取る．そして，ログのホットループに該当する部分を解析することで実行されたパスを抽出し，それらの実行頻度を調べる．

3　プロファイリング結果

図1はMediaBenchプログラムを5000サイクル間隔でサンプリングした結果である．最もサンプリング回数が多いループが#1Loop，#1Loopを除くすべてのループでサンプリングされた総数がother Loops，ループ部分以外でサンプリングされた回数がnon Loopである．棒グラフは，全サンプリング回数を100%としたときのそれぞれの割合を示している．図1は，そのまま大まかな実行時間の割合と見ることができる．今回

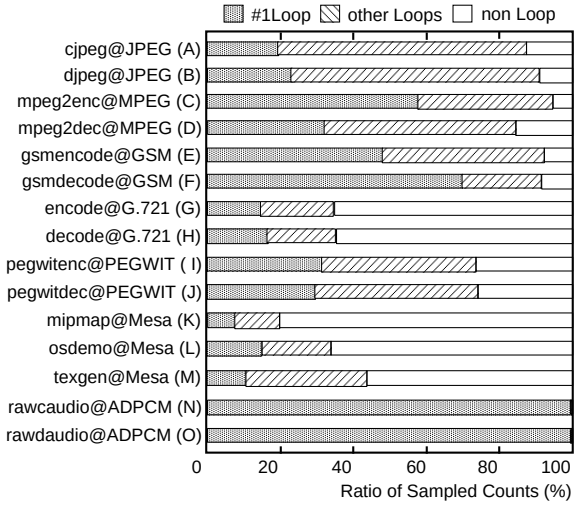


図 1. ループが占めているサンプリング回数の割合

は#1Loopのみに注目しているが，other Loopsの中にもホットループ候補が含まれていることに注意してほしい．なお，パスプロファイリングを行う#1Loopを，プログラムごとに参照しやすくするために，プログラム名@アプリケーション名 (#1Loopの参照名) として示した．

図2は図1の各プログラムの#1Loopをパスプロファイリングした結果である．実行頻度が最も高かったパスが#1Path，その次に高かったパスが#2Path，その他のすべてのパスがother Pathsである．棒グラフは，ループの全イテレーション回数を100%としたときのそれぞれの割合を示している．イテレーション回数は，ループの先頭の基本ブロックが実行されたら1回として数えている．また，パスの途中でループから脱出した場合は，other Pathsとして数えている．なお，各棒グラフの左側に図1の各#1Loopの参照名が示してある．

4　並列化戦略の検討

4.1　パス予測による制御投機

図2を見ると，#1Pathが実行頻度のほとんどを占める場合が多いことが分かる．また，図2のI,Jのように，#1Pathと#2Pathで実行頻度のほとんどを占める場合もあることが分かる．そこで，図2において上位2つまでのパスで実行頻度の8割以上を占めているループを**集中型**，それ以外のものを**分散型**と定義する．

集中型については，パス予測を応用した制御投機による並列化の可能性が考えられる．先行研究として，実行頻度が高い上位2つのパスに絞って予測を行う2パス予測方式を用いた投機的マルチスレッド実行[4]があり，これを参考にしてより効果的な並列化戦略を検討する．一方，分散型については，パス予測による制御投機では成功率が低くなると考えられるため，投機実行を避ける戦略が必要になる．また，ループ構造に着目し，実行され得るパスの本数を調べることによって，可能な限り集中型を事前に検出できるようにする．

Path Profiling for Parallelizing Hot Loops in MediaBench Programs

† Tomohisa Masuho, Takashi Douguchi, Moriyuki Saito, Kanemitsu Ootsu, Takashi Yokota and Takanobu Baba
‡ Department of Information Science, Faculty of Engineering, Utsunomiya University

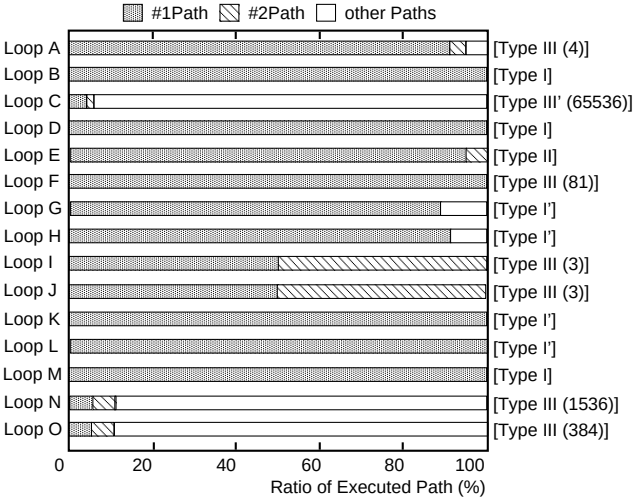
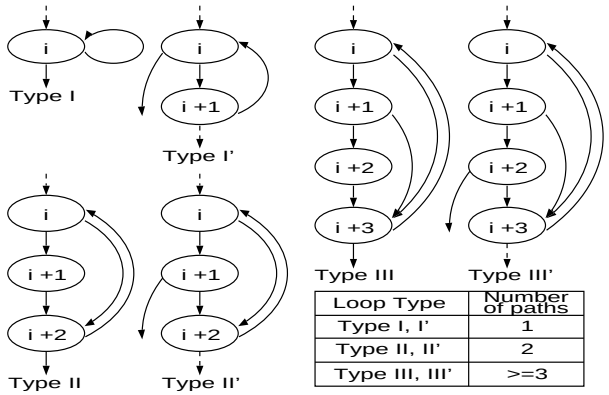


図 2. #1Loop 内のパスの実行頻度の割合



Loop Type	Number of paths
Type I, I'	1
Type II, II'	2
Type III, III'	>=3

図 3. ループ構造分類の例

4.2 パスの本数によるループ構造分類

ループの先頭から末尾へ至るパスの本数を調べ，1本→[Type I]，2本→[Type II]，3本以上→[Type III]とループ構造を分類する．さらに，パスの途中からループを脱出する可能性がある場合は，[']を付けて区別する．以上の分類に当てはまる簡単なループの例を，基本ブロックの制御フローグラフを用いて**図 3**に示す．また，図2の各棒グラフの右側にループ構造を分類した結果を示す．[Type III]の場合は具体的なパスの数を括弧の中に示してある．

4.3 並列化戦略

4.2で分類したループ構造ごとに，パスプロファイリングを利用した場合のスレッドレベルでの並列化戦略を検討する．

[Type I]は，#1Pathしかない集中型になると事前に分かるので，**(1)パスプロファイリングなしで#1Pathを非投機的マルチスレッド実行**する．

[Type II]は，#2Pathまでしかないので事前に集中型になることが分かるが，パスプロファイリングにより，**(2)#1Pathを投機的マルチスレッド実行**するか，**(3)#1Pathか#2Pathを予測して投機的マルチスレッド実行**するかを決める．なお，[Type II]は投機が失敗した場合はもう一方のパスを実行し直せばよい．

[Type III]は，other Pathsが実行される可能性があるので，パスプロファイリングによって集中型か分散型かを判定する．集中型ならば[Type II]のように投機的マルチスレッド実行すればよいが，投機が失敗した場合は，ループの1イテレーションをスレッドとした実行をし直さなければならない．一方，分散型ならば**(4)パス予測による投機実行はしないで，ループの1イテレーションをスレッドとした非投機的マルチスレッド実行**をする．

[Type I',II',III']は，パスの途中からループを脱出するother Pathsが有り得る．図2ではG,H,K,Lが[Type I']であり，[Type II']の該当はないが，これらの場合に見られるother Pathsは，パスの途中からループを脱出している割合であると読み取れる．[Type III']であるCのother Pathsは，他のパスの実行頻度と混ざっているため図2からは読み取れないが，パスプロファイリングでは区別可能である．パスの途中からの脱出頻度が低いと分かった場合は，[Type I']ならば**(5)#1Pathを非投機的マルチスレッド実行**し，[Type II',III']ならば[Type II,III]と同様の戦略を適用すればよい．しかし，多重ループにおける最内ループなどの場合には，パスの途中から頻繁にループを脱出する可能性があるため，パスをスレッドとしたマルチスレッド実行では効率が悪くなると考えられる．従って，脱出頻度が高いと分かった場合は，マルチスレッド実行をしないで**(6)シングルスレッド実行**するという戦略が考えられる．しかし，最適な判定のためには速度向上率などの情報も必要であり，この判断は難しい．

4.4 MediaBenchホットループへの適用結果

4.3で検討した並列化戦略を，図2のMediaBenchホットループに適用すると，**実行方式(1)**はB,D,M，**実行方式(2)**はA,E,F，**実行方式(3)**はI,J，**実行方式(4)**はC,N,O，**実行方式(5)**はG,H,K,Lとなる．**実行方式(6)**はパスプロファイリングだけでは判断できないので，該当なしとなった．今回は，図1の#1Loopのみに適用したが，other Loopsにもこの並列化戦略は適用可能であり，ループが占めている実行時間を対象として高速化が期待できる．

5 おわりに

本稿では，MediaBenchプログラム中のホットループをパス予測による制御投機で効果的に並列化する方法を検討した．今後は，予測成功率を調べるためのパスの出現パターンの解析と，検討した並列化戦略を適用した場合の速度向上率を評価する予定である．

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(B)14380135，同(C)16500023，若手研究14780186)の援助による．

参考文献

[1] Takanobu Baba, et al., "YAWARA: A Meta-Level Optimizing Computer System," Proc. IWIA 2004, pp.148-153, 2004.1.

[2] Chunho Lee, et al., "MediaBench: A Tool for Evaluating and Synthesizing Multimedia and Communications Systems," http://cares.icsl.ucla.edu/MediaBench/.

[3] J. Huang, "The SImulator for Multithreaded Computer Architecture (Release 1.2)," http://www.cs.umn.edu/Research/Agassiz/Tools/SIMCA/simca.html.

[4] Takashi Yokota, et al., "Prediction and Execution Methods of Frequently Executed Two Paths for Speculative Multithreading," Proc. PDCS 2004, pp.796-801, 2004.11.