5H-2

プログラムループおよび実行パスに基づいた自動並列化システムの検討

高橋 辰平 † 白戸 卓志 †† 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 †† 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

マルチコアプロセッサの普及にともないマルチスレッド実行による高速化が重要になっているが，アプリケーションによってはソースコードが必ずしも参照可能とは限らない．我々は，バイナリレベルでシングルスレッドコードからマルチスレッドコードへ変換するシステムの研究開発を行っている．

我々は，複雑な制御構造を持つプログラムに対して，プログラムの実行割合の最も高いパス (最頻度パス) を基にスレッド分割を行うパスベーススレッド分割手法において問題となる，実行時間の長いループによるスレッドサイズの偏りを，ループ分割により解決する手法 [1] 提案している．しかし，この手法では，ループ分割によりスレッド間のデータ依存が生じないループのみを分割の対象としている．そのため，データ依存が生じるループが最頻度パス内に存在する場合，スレッドサイズの偏りを解消することは出来ず，ループ分割による高速化の機会が減少している．

本研究では，これらの問題を解決するため，データ依存が存在するループも分割対象とする，パスベーススレッド分割手法を適用したシステム検討した．実装には，すでに開発されているパスベース自動並列化システム [2] およびループベース自動並列化システム [3] を統合することで，コストを抑え開発することが出来ないか．また，システム統合に必要な処理の検討した．

2 ベース手法

ここではまず，ループ構造を考慮したパスベーススレッド分割手法 [1] について説明を行う．その後，手法の問題点や対処法について述べる．

2.1 ループ分割を行うパスベーススレッド分割手法

パスベーススレッド分割手法では，最頻度パスにループ構造ある場合，ループによるスレッドサイズの偏りにより，スレッドのオーバーラップされ実行される部分が少なくなるため，高速化の妨げとなっている．そこで，ループを任意のイテレーション単位に分割しスレッドを生成することで，スレッドサイズの偏りが解消できれば，図 1 のように高速化が可能である．図 1 では，図 1(a) が従来の分割手法によるスレッドで，スレッドサイズに大きな偏りがあるのに対し，図 1(b) ではループ分割が適用され，スレッドサイズの偏りが改善されている．

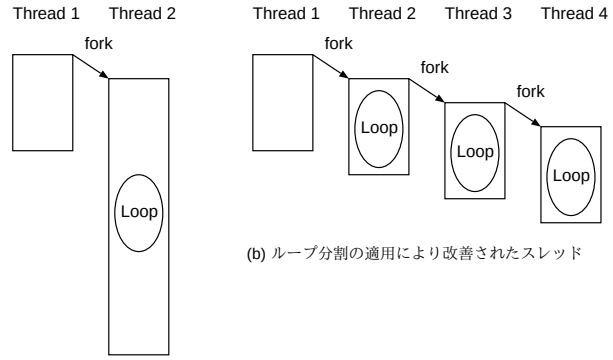

図 1: ループ細分により期待される効果

2.2 ベース手法の問題点と対処法

この手法では，スレッド間同期を起こさずにマルチスレッド実行を実現する従来のパスベーススレッド分割手法の利点を損なわないように，スレッド間の同期を起こさずにマルチスレッド化可能なループに限定してループ分割の対象としている．そのため，データ依存が生じるループがある場合，スレッドサイズの偏りを解消することは出来ず，図 1 のような効果が得られない．そこで，データ依存が存在するループについても分割の対象とする事が考えられるが，それにもいくつかの問題が生じる．

まず，全てのループがループ分割により高速化出来るわけではないこと．データ依存による同期待ち時間が長いループなどにおいては，ループ分割により逆にシングルスレッド実行より実行速度が低下する場合もある．そこで，ループの選別を行い分割対象とするループを決める必要がある．ループ選別については 3 節にて説明する．

次に，図 2 に示すようなデータ依存があるパスをループ分割することにより，ループ以外のスレッドにも依存が生じる問題がある．これにより同期間ちが生じる可能性がある他，依存変数の登録命令を追加する必要があるため次スレッドの起動が遅くなり，高速化の妨げとなる．そこで，パスベース自動並列化システム [2] に用いられているデータ依存命令の移動の手法を適用し，このようなデータ依存は可能な限り消去する必要がある．

A Study of Loops and Paths oriented Automatic Parallelizing System

†Shinpei Takahashi, ††Takashi Shiroto, ††Kanemitsu Ootsu,††Takashi Yokota and ††Takanobu Baba
Department of Infomation Science, Faculty of Engineering, Utsunomiya University (†)
Department of Information Systems Science, Graduate School of Engineering, Utsunomiya University (††)

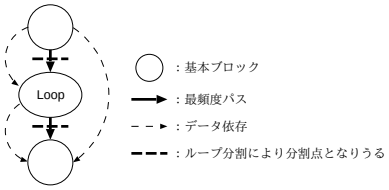

図 2: ループ分割の際に消去を試みるデータ依存

3 ループ選別

ループ分割による性能低下防ぐため，高速化が見込めるループのみを分割の対象とするループ選別が必要となる．

まず，ループベーススレッド分割手法 [3] では誘導変数が一定の変化量でないループは並列化不可であることから，誘導変数がイテレーション毎に一定の変化量であるかを解析する必要がある．次に，イテレーション数が多い方がループ分割の効果が大きく，また，イテレーション数が少なければスレッドサイズの偏りは大きくならないと考えられることから，ループのイテレーション数が一定以上である必要がある．なお，イテレーション数は，静的な解析が困難であるためプログラムを一度実行し解析する．最後に，ループ分割による高速化が見込めるかどうかを判断する必要がある．これは，図 3 に示すように，左側のシングルスレッド実行時間の見積りと，右側のマルチスレッド化によるオーバーヘッドと同期待ちを加算したマルチスレッド実行時間の見積りを比較して，シングルスレッド実行時間の見積りよりマルチスレッド実行時間の見積り方が短かければ，高速化が見込める．

これらの条件全てに当てはまるループのみをループ分割対象とする．

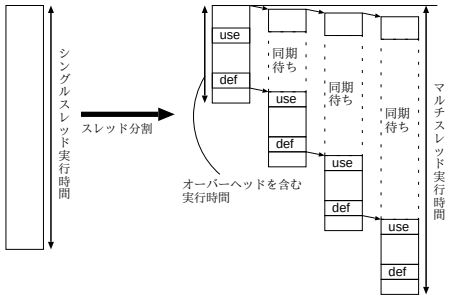

図 3: ループ選別の方法

4 改良手法システム化への検討

これまでに述べてきた手法を取り入れた，ループ分割を行うパスベーススレッド分割手法を，パスベース自動並列化システム [2] およびループベース自動並列化システム [3] の統合による実現を検討する．

システムの実行の流れを図 4 に示す．

本システムでは，少ない変更で統合が可能．また，パスベース自動並列化システムまたはループベース自動並列化システムに，より性能向上が得られる手法が適用されたとき，少ない変更で本システムにも適用可能と考え，パスベーススレッド分割とループベーススレッド分割を分けて行う．ループ選別は，ループを分割するか否かで，パスベーススレッド分割のデータ依存命令の移動の処理を変える必要があるため，パスベーススレッド分割より先に行う必要がある．ループ選別やループ分割については，最頻度パスは実行割合が高く全体への効果が大きい．また、新たな依存を考慮する必要が無いので，最頻度パス上のループのみを対象とする．プログラムの分割の手順は，まず，パスベースでスレッド分割を行い，その後ループベースで，最頻度パス内のループの分割を行う．

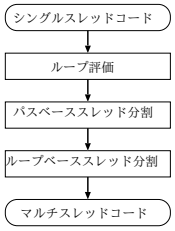

図 4: システムの処理の流れ

5 おわりに

本稿では，パスベース並列処理とループベース並列処理を統合することで，ループによるスレッドサイズの偏りを解消することにより性能向上を達成するシステムの検討を行った．

今後は，システムを実装し，性能評価を行う予定である．

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) および宇都宮大学若手萌芽的研究プロジェクトの援助による．

参考文献

[1] 小川 大仁，伊里 拓也，大津 金光，横田 隆史，馬場 敬信: “ループ構造を考慮したパスベーススレッド分割手法の検討”, 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.108, No.303, pp.1-6, 2008.

[2] 伊里 拓也，大津 金光，横田 隆史，馬場 敬信: “パスベーススレッド分割手法に基づく自動並列化処理の実装”, 情報処理学会 第 72 回全国大会 講演論文集, pp.1-201-1-202, 2010.

[3] 白戸 卓志，大津 金光，横田 隆史，馬場 敬信: “バイナリレベル変数解析に基づいた自動並列化処理の初期評価”, 信学技報, Vol.110, No.318, pp.31-36, 2010.