3J-8

Valgrind ベース自動並列処理系におけるループ並列化機能の初期実装

小渕 裕之 † 星 孝幸 †† 大津 金光 †† 大川 猛 †† 横田 隆史 ††
† 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

近年普及しているマルチコアプロセッサの性能を有効に活用するためには, マルチスレッド化したプログラムコードが必要となる. しかしながら, ユーザがマルチスレッドコードを作成することは困難であり, ソースコードを必要とするマルチスレッド化手法では, ソースコードが参照できないプログラムの並列化は不可能である. そこで, 我々は逐次のバイナリコードを自動で並列に処理するシステムを開発している [1].

本稿では, システムにおける並列化機能の初期実装として, 基本的な構造をしたループの並列化機能について述べる.

2 Valgrind ベース自動並列処理系

逐次バイナリコードから並列化コードを生成するために, バイナリ変換機能が必要となる. 我々は様々な命令セットのバイナリコードの並列化を目的として, 複数の命令セットに対応した動的バイナリ変換を可能とする Valgrind[2] システムをベースにした.

Valgrind はプロファイラのような動的計測ツールを構築するオープンソースのフレームワークである. Valgrind の構成を図 1 に示す. Valgrind は Valgrind core と User Plug-in Tool によって構成される. Valgrind core は Valgrind で解析する Guest Application に対してバイナリ変換を行う. Plug-in Tool は Valgrind core によってバイナリコードから生成された中間表現コードに対して, 計測用コードの挿入などを行う. ユーザは Plug-in Tool を開発することで, Valgrind に独自のバイナリ変換処理機能を追加することができる.

Valgrind におけるバイナリ変換は基本ブロック単位で行われる. はじめに, 解析するプログラムのバイナリコードを, Valgrind core において命令セット独立の中間表現コードに変換する. 次に, その中間表現コードを Plug-in Tool に渡し, Plug-in Tool は中間表現コードに対して計測用コードの挿入などを行う. そして, Plug-in Tool によって変更された中間表現コードを Valgrind core がターゲットマシンのバイナリコードに変換し, それを Software Code Cache に格納する. 格納された変換コードは Dispatcher を経由して実行される.

我々が開発している自動並列処理システムでは, プログラムの実行に多くの時間を要するループを並列化することで, プログラム全体の高速化を目指す. これを実現するために, 並列化機能を持った Plug-in Tool を新たに開発する. また, 並列化したコードの実行に必要なスレッド生成, および制御する機能を Valgrind core に追加する. 本稿では, システムを始めから実行しているスレッドを親スレッド, 並列実行のために生成したスレッドを子スレッドと呼ぶ. システムは並列化対象ループに実行が到達したら, Plug-in Tool によってそのループの並列化コードを生成し, 子スレッドで実行する. ここで, 生成した子スレッドにはレジスタコンテクストが設定されていないため, 並列化コードを実行する前に親スレッドから子スレッドへ必要な値をコピーする. これにより, 子スレッドで並列化コードを実行できる状態にする. 並列実行が終了したら, 今度は子スレッドから親スレッドへ必要な値をコピーし, 次の基本ブロックの処理に移る.

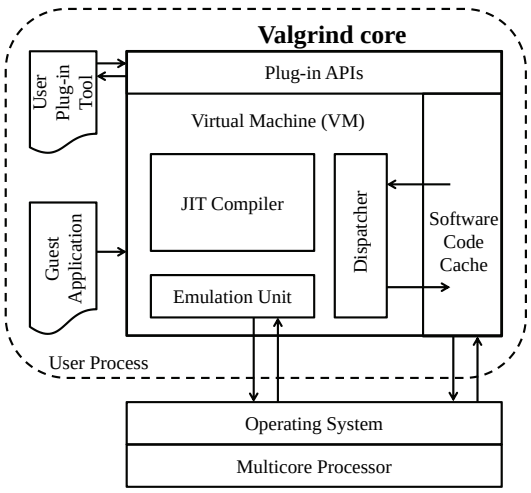

図 1: Valgrind の構成

3 基本ループのバイナリコード並列化

本研究では, システムにおけるループ並列化機能の初期実装として, 基本ループの並列化を行う Plug-in Tool を開発する. Plug-in Tool では以下の 4 つの処理を中間表現コードに対して行うことで, 並列化を実現する.

1. ループの検出
2. 検出したループ内の変数解析
3. ループの並列化可否判定
4. 並列化コードの生成

Preliminary Implementation of Loop Parallelization in Automated Parallel Processing System by using Valgrind
†Hiroyuki Obuchi, ††Takayuki Hoshi, ††Kanemitsu Ootsu, ††Takeshi Ohkawa and ††Takashi Yokota
Department of Information Science, Faculty of Engineering, Utsunomiya University (†)
Department of Information Systems Science, Graduate School of Engineering, Utsunomiya University (††)

ループを並列化するとき, そのループが並列化可能であるかを判定しなければならない. そのためには, ループを解析し, ループのイテレーション間の依存関係とループ内の変数がどのような計算に使われているかを把握する必要がある. ループ内に多くの分岐が存在するような複雑な構造をしたループや, ループ内で配列変数へアクセスするループの場合, ループの解析が難しくなる. そこで, 今回はループが一つの基本ブロックで構成され, ループ内で配列変数へのアクセスが存在しない単純なループを並列化の対象とした. 並列化の対象となる主なループは総和計算のようなリダクション計算を行うループである.

基本ループの並列化を行うために, まず一つの基本ブロックで構成されるループを検出する必要がある. 検出は Plug-in Tool に渡された基本ブロックの中間表現コードから, ブロックの先頭アドレスとブロックの分岐先のアドレスを取得し, 互いを比較することにより行う. 互いのアドレスが一致する場合は, その基本ブロックをループとして検出し, そうでない場合は今回の対象ループではないため並列化を行わない.

次に検出したループに対して変数解析を行う. ループ内のそれぞれの変数について以下の情報を取得する.

- 変数の種類
- 変数へのアクセスに使用するレジスタの番号とそのレジスタが指すアドレスからのオフセット
- 変数が使用される命令とその命令の位置

変数は, 使用される命令によってイテレーション間で依存する変数とそうでない変数, およびループ変数とリダクション変数に分類する. 変数の分類はループの並列化可否判定に必要となる. また, 親スレッドと子スレッド間での値のコピーや並列化コード生成において, ループ変数とリダクション変数を個別に指定するため, ループ変数とリダクション変数の分類が必要となる. レジスタの番号とオフセットは変数の位置を表し, 親スレッドと子スレッド間での値のコピーにおいて変数の位置を指定するために使用する.

ループの並列化可否判定はイテレーション間で依存する変数の有無, および変数解析で種類を判別できなかった変数の有無によって判定する. ループ内に依存変数や種類の判別ができなかった変数が存在しない場合に並列化を行う.

現状のシステムでは, Valgrind core のスレッド実行制御上の制約により, Plug-in Tool においてスレッドごとに異なるコードを生成することができないため, 各子スレッドで実行する変換コードは全て同じコードになる. 今回は各子スレッドで同じ変換コードを実行しても正しい計算結果が得られるように, ループのイテレーションをサイクリックに分割する. サイクリック分割では, スレッドごとの増分値と終了条件が共通するため, 各子スレッドでループ変数の初期値をあらかじめ設定することにより, 同一のコードを正しく並列実行することができる. サイクリック分割はループ内のループ変数の増分値を, 元の値に子スレッドの数を掛けた値に変更することによって実現する.

並列化コードを生成するためには, 元のコードにおけるループ変数の増分値を変更する必要がある. また, 並列実行を終えた際に各子スレッドで求めたリダクション計算の結果を集計し, 集計した値を親スレッドに渡さなければならないので, リダクション計算の結果を集計するコードの追加が必要となる. ループ変数の増分値の変更例を図 2 に示す. また, リダクション計算の結果を集計するコードは, ループの先頭へジャンプする命令の次に追加する. これにより, 各子スレッドでループの実行が終了した後にリダクション計算の結果を集計する処理が行われる.

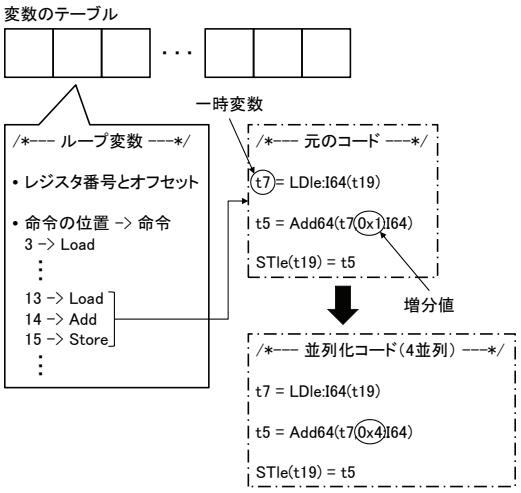

図 2: ループ変数の増分値の変更例

4 おわりに

本稿では, Valgrind を利用したバイナリレベル自動並列処理システムにおいて, 基本ループの並列化機能について述べた. 今後はより複雑なループを並列化できるように拡張する予定である.

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 (C)24500055，同 (C)24500054，同 (C)25330055，若手研究 (B)25730026) の援助による.

参考文献

[1] 星孝幸，大津金光，大川猛，横田隆史，馬場敬信: “動的計測ツールのバイナリ変換機能を利用した自動並列処理システムの開発”, 情報処理学会第 74 回全国大会講演論文集, pp.1-139˜1-140, 2012.

[2] Nicholas Nethercote and Julian Seward: “Valgrind:A Framework for Heavyweight Dynamic Binary Instrmentation”, Proc. of ACM SIGPLAN 2007 Conference on Programming Language Design and Implementation (PLDI 2007), pp.89-100, 2007.