Vol.2012-ARC-199 No.3
2012/3/27

並列化メディアアプリケーションを対象とした メニーコアアーキテクチャシミュレーションの高速化の検討

阿部洋一[†] 石塚亮[†] 大胡亮太[†] 田口学豊[†]
木村啓二[†] 笠原博徳[†]

本稿ではプログラム上のループ処理に対する統計的サンプリングによる，並列化メディアアプリケーションを対象としたメニーコアシミュレーションの高速化手法を検討する．筆者などによるこれまでのループ全体を 1 つの母集団とみなしてサンプリングを行う手法では，イテレーション間のコスト変動が大きい場合に，サンプル数の増加や推定誤差が大きくなってしまうという問題があった．本稿では，この問題を K 平均法によるクラスタリングを用いてサンプル数，サンプル位置の特定を行うことで解決を試みる．動画圧縮アプリケーションの MPEG2 エンコーダを実サーバー上で動作させて得た計測結果を元に検証したところ，クラスタリングをすることによって，クラスタリングを行わない場合と比較して最大で 11.4 倍の速度向上が見込まれることが分かった．推定誤差については，検証した全ての条件で目標とする誤差の範囲に収められる見込みとなった．

An Examination of Accelerating Many-core Architecture Simulation for Parallelized Media Applications

Yoichi Abe[†], Ryo Ishizuka[†], Ryota Daigo[†], Gakuho Taguchi[†], Keiji Kimura[†], and Hironori Kasahara[†]

This paper examines an acceleration technique of many-core architecture simulator by statistical sampling on loop blocks in a program. The authors' previous work has a problem that the number of samples increases and estimation error becomes greater, since the previous work assumes that the cost of iterations is not varied so much. This paper proposes using clustering by K-means for making decision about sampling in order to solve the problem. MPEG2 encoder, that is video compression application, is used for examination. The preliminary evaluation shows the proposed technique achieves 11.4 times speed up compared with the previous work. In addition, the estimation error becomes under the target range.

1. はじめに

アーキテクチャシミュレータを用いることによって，コンピュータアーキテクチャの研究開発における検証を容易に行うことができ，コストを削減することができる．しかしながら，ソフトウェアによるシミュレーションは評価に非常に長い時間がかかってしまう．特に，複数のプロセッサコアについて検証を行うメニーコアのアーキテクチャシミュレーションの場合，コア数の増加がさらなるシミュレーション時間の増加につながり，メニーコア研究開発の妨げとなっている．

このようなアーキテクチャシミュレーションの課題に対し，シミュレーション結果について高い精度を保ちつつ，高速化を行うような手法の研究が進められている．このような研究の一つに，プログラム全体のうち一部分のみ詳細なシミュレーションを行い標本抽出（サンプリング）することで高速化を図る手法として，SimFlex1)とSimPoint2)が挙げられる．

一方，筆者等はこれまでに，シミュレーションに用いる並列プログラムのループ処理に対してサンプリングを行い，全体の実行サイクル数の推定をする手法を提案してきた．本手法では，始めに，対象プログラムを並列化せずに任意の実機上にて実行し，サンプリングを行うループの 1 イテレーションごとの実行サイクル数を計測する．その結果を用いて，推定する全体の実行サイクル数が期待する誤差に収まるような最小のサンプル数を統計的手法によって計算する．このようにして求めたサンプル数分のイテレーションについて詳細なシミュレーションを行う．また，それ以外のイテレーションについては簡易な機能シミュレーションを行うことでシミュレータの状態を保つ．

本手法により，イテレーション間の実行サイクル数の変化があまり大きくない，科学技術計算のようなアプリケーションを対象とする場合に，高精度かつ高速なシミュレーションが可能となることが見込まれている3)．しかし，動画像処理などのメディアアプリケーションのような，イテレーション間で処理量が大きく変わりやすいプログラムの場合では，大きな高速化は見込めず，精度についても期待する範囲に収まらないことがある．本稿では，プロファイル情報をクラスタリングし，クラスタ毎にサンプル数の決定を行い，サンプリングを行うことで並列化メディアアプリケーションを対象としたアーキテクチャシミュレーションを高速化することを検討する．また，検討手法に関する予備評価について述べる．

以下，2 章ではサンプリングとクラスタリングを利用したシミュレーション高速化手法について，3 章では予備評価について，最後の 4 章でまとめをそれぞれ述べる．

[†] 早稲田大学
Waseda University

2. シミュレーション高速化手法

本章では，本稿で提案するシミュレーション高速化手法について述べる．2.1 節では実機上の実行情報を用いたサンプリングによる高速化手法について述べる．2.2 節ではクラスタリングのシミュレーション高速化手法への導入について提案を行う．

2.1 サンプリング回数決定手法

本手法は，プログラムの構造に着目し，ループ処理のうち一部のイテレーションを詳細に実行する（サンプリング）という点が特徴である．また，ループのイテレーション間の実行サイクル数の変化はプログラム，あるいは入力データに大きく依存し，マイクロアーキテクチャの違いによる差異は小さい，という前提を持つ．

始めに，シミュレーションに用いるプログラムを並列化せずに任意の実機上にて実行し，サンプリングを行うループの 1 イテレーションごとの実行サイクル数を計測する．サンプリング対象ループは並列ループ，またはその並列ループを内側に持つような大きなループを選択する．次に，計測結果から，推定する全体の実行サイクル数が期待する誤差に収まるような最小のサンプル数を統計的手法によって算出する．この計算には，計測結果として得られた 1 イテレーションごとの実行サイクル数の相加平均 μ，標準偏差 σ と，標準正規分布の上側 P%点，許容する誤差率を用いる．このうち標準正規分布の上側 P%点は P=2.5 の時の 1.96 を，許容する誤差は 5%，すなわち 0.05 を用いるとする．以下の(1)がその計算式である．

$$\text{サンプル数} n \geq \left(\frac{1.96}{0.05} \times \frac{\text{標準偏差}\sigma}{\text{相加平均}\mu} \right)^2 \quad (1)$$

以上のようにして求めたサンプル数分のイテレーションについて詳細なシミュレーションを行い，シミュレータ上での実行サイクル数を得る．

また，全体の実行サイクル数の推定は以下の式(2)によって行われる．

$$\text{推定実行サイクル数} = \text{詳細シミュレーションサイクル数} \times \frac{\text{総イテレーション数}}{\text{サンプル数} n} \quad (2)$$

2.2 クラスタリングを利用したサンプリング回数決定手法

前節で述べた高速化手法は，イテレーション間の実行サイクル数の変化が小さいプログラムを対象とする場合は速度向上率，推定誤差ともに良い結果を示すことが見込まれている 3)．特に，科学技術計算を行う一部のベンチマークアプリケーションについて，数百回の回転数のうち数回の詳細シミュレーションを行えば十分に小さな誤差で全体の実行サイクル数が推定できるという見通しである．しかしながら，動画像の圧縮処理のようにイテレーション間の実行サイクル数の変化が大きく，またその変化が不規則なプログラムを用いる場合，算出されるサンプル数は小さくならず，サンプリングを行うイテレーション位置によっては誤差も大きくなってしまう．そこで，本稿ではこのようなプログラムを対象にシミュレーションを行う際にプロファイル結果をクラスタリングし，クラスタ毎にサンプル数を計算する手法を提案する．

クラスタリングとはデータの集合を，類似度などを元に集合（クラスタ）に分類する操作である．そのアルゴリズムは目的・用途によって様々なものがある．よく知られた手法の一つに K 平均（K-means）法がある．このアルゴリズムは，各データへのクラスタの割り当て，各クラスタの代表値の算出という操作を繰り返して代表値を改善していき，代表値が変化しなくなるまで続けるというものである．

本稿にて検討する手法は，実機上の実行によって得られたデータをいくつかのクラスタに分類し，クラスタ毎に式(1)によってサンプル数を算出，詳細シミュレーションを行うサンプル位置を決定する．クラスタリングを用いたサンプル情報の決定方法のイメージを図 1 に示す．

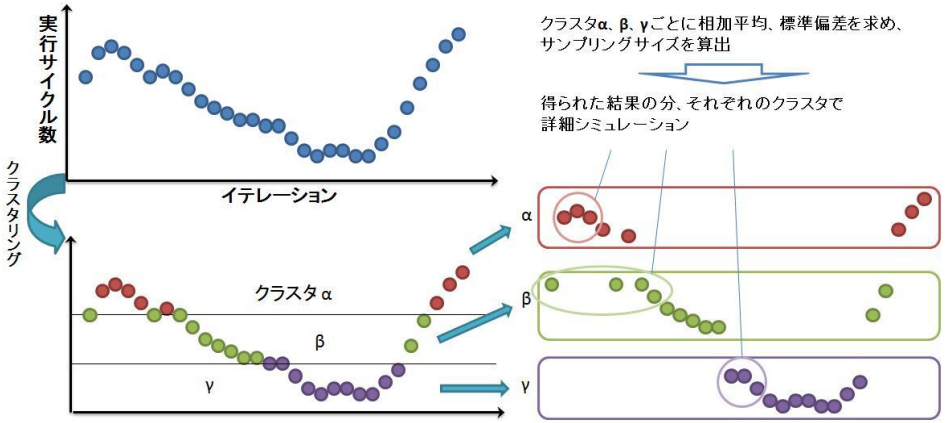

図 1 サンプル情報決定へのクラスタリングの導入イメージ

シミュレーション後，クラスタ毎に以下の式(3)によってクラスタ単位の推定実行サイクル数を算出して，得られた結果を総計し，全体の推定実行サイクル数とする．

$$\text{クラスタ単位の推定サイクル数} = \text{該当クラスタの詳細シミュレーションサイクル数} \times \frac{\text{該当クラスタのノード数}}{\text{サンプル数} n_i} \quad (3)$$

3. 予備評価

本章では，本稿で提案する手法の予備評価として，動画像ファイルの圧縮処理を行うアプリケーションである MPEG2 エンコーダを実機上で逐次で実行し，処理の一番大きな部分のループの 1 イテレーションごとの実行サイクル数を計測する．得られたデータを K 平均法によりクラスタリングし，それぞれのクラスタ毎にサンプリングに必要なサンプル数を算出して合計，クラスタリングをしない場合のサンプル数との比較を行う．また，実機上の実行で得られた情報をサンプリングの対象として推定誤差の計算を行い，クラスタリングをしない場合との比較を行う．

3.1 評価アプリケーション

MPEG2 エンコーダは MPEG2 の規格に沿った圧縮処理を行うプログラムである．元動画は 1 フレームずつ順番に処理されるが，この処理の 1 単位はピクチャと呼ばれる．入力フレーム列は I, P, B の 3 種類のピクチャタイプの規則的な並びとして扱われる．図 2 に，その並び方の一部を示す．また，図 3 に 1 ピクチャに対して行われる処理の流れを示す．ピクチャタイプによって，この施される処理の内容が一部異なるため，ピクチャタイプの違いは計算量の違いに大きく表れる．今回は，この 1 ピクチャごとの処理にかかる実行サイクル数の計測を行う．

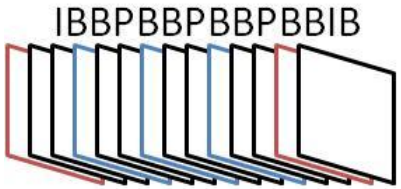

図 2 MPEG2 エンコーダのフレームの並び

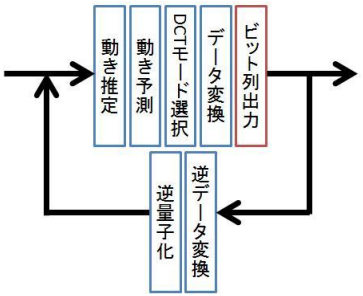

図 3 MPEG2 エンコーダの圧縮処理の流れ

3.2 入力動画

メディアアプリケーションは，入力データの違いが計算量に大きく影響を与えることが多い．今回の評価については，NHK システム評価用標準動画像解析度 SIF(352x240)4)から 5 種類の動画を選択して，各々タイトル部分を除いた 450 フレーム分について評価を行った．表 1 に，評価用の入力として選択した動画の概要を示す．

3.3 評価環境

今回の評価を行うに当たって，ネイティブコンパイラは gcc version 4.3.2 を用いた．また，表 2 に今回の評価を行う実サーバーの仕様を示す．

表 1 評価アプリケーションの入力動画

番号	タイトル	内容
02	花かご	色鮮やかな花の画像
06	交差点	交差点での車の往来の風景
07	市場	花売りなど市場のルーズショット
16	シャチのジャンプ	水族館のショーでジャンプするシャチと観客の風景
20	サッカー	サッカーの試合のルーズショット

表 2 評価用サーバーの仕様

CPU	Intel Xeon E5440
CPU 数	8
CPU Clock	2.83GHz
L2 Cache	6MB
Main Memory	7.8GB

3.4 評価結果

1 イテレーションあたりの実行サイクル数の推移を図 4 に示す．図 4 より，各動画でイテレーション間のコスト変動が大きいことが分かる．また，その変動の仕方も動画の種類によって大きく異なることが分かる．これら 5 種類の動画に関する計測結果に対し，それぞれクラスタ数 3, 5, 7 の 3 種類の条件にてクラスタリングを行った．

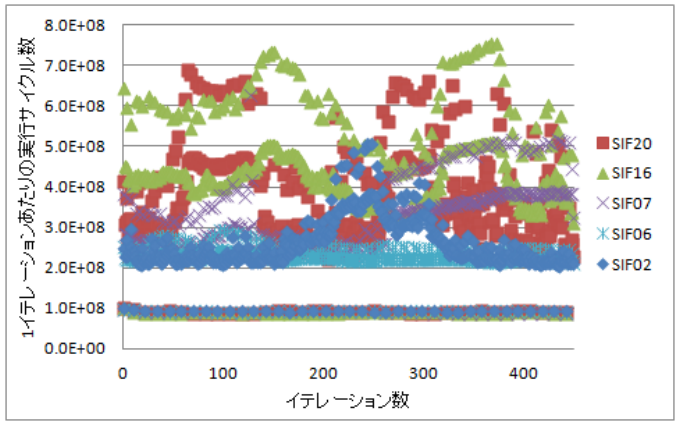

図 4 1 イテレーションあたりの実行サイクル数の推移

2.2 節にて提案した手法によって必要なサンプル数を計算した結果について，全フレーム数である 450 に対する比率を図 5 に示す． さらに，クラスタリングの結果と算出したサンプル数を元に，サーバー上での実行結果から推定サイクル数を計算した．実際の実行結果との誤差を図 6 に示す．

図 5 より，クラスタリングを行うことによりサンプル数を削減することができ，クラスタ数 7 の場合に 1.78%~5.33%となっていることが分かる．一方クラスタリングを行わない場合は，12.4%~48.7%と必要なサンプル数が大きいことが分かる．

また図 6 より，クラスタリングを行った場合は低い誤差で総コストを見積もることができ，クラスタ数 7 では最大 3.02%の誤差となっていることが分かる．その一方でクラスタリングなしの場合は最大 12.8%の高い誤差となっている．以上の結果より，クラスタリングを行うことで，少ないサンプル数で高い精度の推定値が得られることが確認できた．

4. まとめ

クラスタリングをすることによって，今回の条件ではクラスタリングをしない場合より最大で 11.4 倍の速度向上が見込まれることが分かった．また，サンプリングとクラスタリングを組み合わせることで，最大 75 倍程度のシミュレーション時間の高速化が見込まれることが分かった．推定誤差については，クラスタリングをしない状態で期待する誤差（5%）に収まらなかった入力パターンでは大幅に改善している．一方で，クラスタリングをかけなくても誤差が小さい入力パターンにおいては誤差が大きくなってしまうことがあることも分かった．ただし，その場合においても期待する誤差（5%）には収められている．

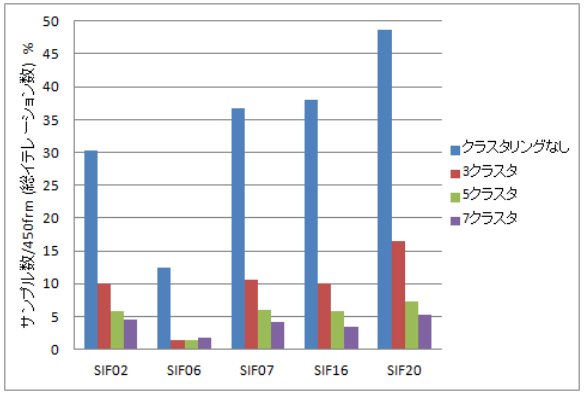

図 5 総イテレーション数に対するサンプル数の比率の比較

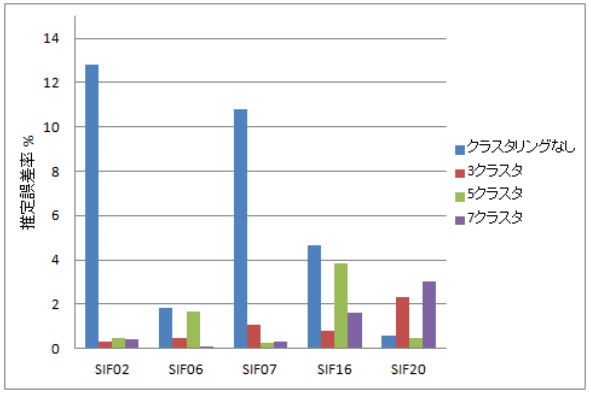

図 6 サーバー上の計測結果を元にした推定誤差の比較

謝辞

本研究の一部は科研費若手研究（B）23700064 の助成及び，経産省グリーンコンピューティングシステム研究開発により行われた．

参考文献

1) Thomas F. Wenishch, Roland E. Wunderlich, Michael Ferdman, Anastassia Ailamaki, Bavak Falsafi, and James C. Hoe, “Sim-Flex: Statistical Sampling of Computer System Simulation” Micro IEEE, Volume 26, Issue 4, pp.32-42, July-Aug, 2006
2) Erez PerelmanGreg HamerlyMichael Van Biesbrouck Timothy SherwoodBrad Calder “Using SimPoint for Accurate and Efficient Simulation” SIGMETRICS ’ 03, San Diego, California, USA. ACM 1-58113-664-1/03/0006, June 10—14, 2003
3) 石塚亮，阿部洋一，大胡亮太，木村啓二，笠原博徳，“科学技術計算プログラムの構造を利用したメニーコアアーキテクチャシミュレーション高速化手法の評価”, 情報処理学会研究報告．計算機アーキテクチャ研究会報告 2011-ARC-196(14), 1-11, 2011-07-20
4) 財団法人 NHK エンジニアリングサービス