3K-5

非均質環境における分散アルゴリズムの性能予測と利用支援環境の構築

千田 成樹　菅谷 至寛　阿曽 弘具
東北大学大学院工学研究科

1　はじめに

近年, クラスタ環境の普及により,MPI(Message Passing Interface) を用いた並列計算のニーズが高まってきている. 一般に並列計算のプログラムは,PU(Processing Unit) の数の増加にともなって実行性能が向上するが, ある一定数を越えると逆に低下することがある. そのため, 事前に並列プログラムの実行時間を予測し, 最適なPU 数を見積もる, あるいは実行時間予測の情報を元に最適なPU 構成を求める手法が提案されている [1]. 実行時間予測として, 岩渕らによる簡易実行による手法 [2] がある. この手法は均質環境において, 正確な予測を実現している. 本研究では非均質環境における正確な予測のために, 岩渕らの手法を改良した簡易実行による予測手法を提案する. また, この予測手法の利用支援環境を実現する支援ツールについて述べる.

2　簡易実行による実行時間予測の改良

簡易実行による予測手法は並列プログラムを計算ブロックと通信ブロックに分け, 全体のループ数を減じて実行することにより, キャッシュ効果などの影響を考慮しつつ短時間で実行時間予測を行うことを可能にしている. しかし, 岩渕らの手法では計算ブロックの実行時間と通信ブロックの通信時間の取得を別々に行っており, 通信ブロックの測定時に計算ブロックの実行はほとんど行われない. そのため, 各PU の実行性能に差がある非均質な環境で測定する場合, 本来生じるはずの通信待ちの時間が発生せず正確な実行時間予測ができない.

この問題を解決する手法ため, 計算ブロックと通信ブロックの実行時間を同時に取得することにした. これにより実行性能の差による通信の待ち時間の発生が期待できる. 従来手法ではデータ取得時のオーバヘッドを軽減するために別々に取得していたが, 実際のオーバヘッドは実測時間と比べて非常に小さく, 同時に取得することに問題なかった.

非均質環境に対する簡易実行による予測手法の概略を以下に示す.

1. MPI プログラムを計算ブロックと通信ブロックに分割
2. ループ回数を減じ, 各計算・通信ブロックの実行時間を取得し, 1 回当りの平均実行時間を求める
3. 実際の計算・通信ブロックの実行回数を測定する, この時実行回数を変化させない実行文や通信をコメントアウトする.
4. 2 で得た各ブロックの平均実行時間に3 で得た各ブロック実行回数を乗じ, その総和を予測実行時間とする.

3　自動予測ツールの構築

ユーザが予測のための計測コードを付加することは煩雑で手間がかかることである. そこでC 言語で記述された対象プログラムに対して提案手法によって予測を行うためのツールを開発した. 本ツールは対象プログラムに対して字句解析・構文解析を行い, 計算および通信ブロックの実行時間計測プログラムと実ブロック実行回数取得プログラムを自動生成する. そして取得結果から予測実行時間を算出し, 各PU の予測総計算・通信時間と計算対通信比を表示する.

以下にこのツールを使った実行時間予測の流れを示す.

1. ユーザが対象プログラムを入力
2. 本ツールが対象プログラムから計算・通信ブロック実行時間データを取得するソースを出力
3. 実実行回数取得を行うソースを出力
4. 出力したソースをコンパイルし, 実行して各パラメータを取得
5. 集計用プログラムを実行

Performance Estimation of Parallel Algorithms in Heterogenous Environment and Development Tuning Support System
CHIDA, Masaki
SUGAYA, Yoshihiro
ASO, Hirotomo
School of Engineering, Tohoku University

6. 予測実行時間等を出力

4 実験

本手法の評価のため, 姫野ベンチマーク 98[3] の実行時間を予測し, 実実行時間との比較を行う. 実験環境は以下の通りである.

- 均質環境と非均質環境両方について実験
  - ○ PentiumⅢ800MHz×8
  - ○ PentiumⅢ500MHz×4,800MHz×4
- MPI/LAM 7.1.1
- GCC 3.3.5 -O3

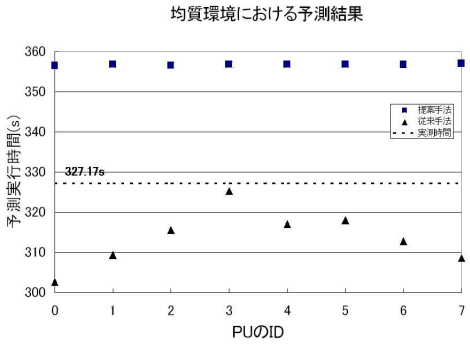

図 1: 均質環境での実験結果

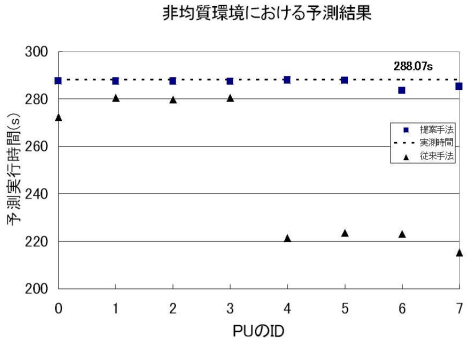

図 2: 非均質環境での実験結果

図2 に示すように非均質環境においては提案手法は誤差率 1%以下を示しており, 従来手法の 2.6 ～ 25.3%から予測精度の向上が見られる. 従来手法の PU4 ～7 の誤差は計算能力の差による通信待ち時間を考慮していない影響を表している. 図 1 の均質環境の結果では提案手法が約 9%の誤差であり, 従来手法の 0.6 ～ 7.5%と比較しても精度を大きく落としてしまっている. この原因は, ループの始めの数回において, キャッシュヒットが安定しないため, 通信待ち時間が安定しないことにあると思われる. 図 3 は PU7 のある一つの通信ブロックについて通信時間の変化を示している. 収束前の通信時間も対象として平均をとるため, 実際に有効になるはずの収束後の通信時間と大きい誤差が生じてしまう.

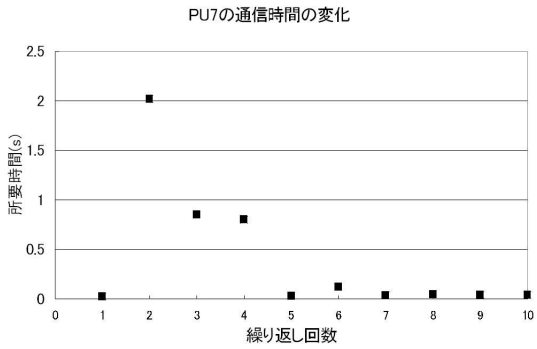

図 3: PU7 における一つの通信ブロックの通信時間

通信時間収束前のデータは平均実行時間に含めないようにする仕組みが必要と考えられる.

5 まとめ

本提案手法により, 計算ブロックと通信ブロックの実行時間を同時に取得することで非均質環境における予測誤差を小さくすることができた. また,C 言語を対象とした予測ツールを開発した.

今後の課題は不安定な初期通信時間への対応と, 本稿では姫野ベンチマーク 98 のみで実験を行ったが, その他のベンチマークプログラムに対する評価を行うこと, そしてユーザに提示する情報をもっと多様化することである. 本予測手法を用いて最適な PU 数および組合せを発見できるようにすることも今後の課題である.

参考文献

[1] Yoshinori Kishimoto,Shuichi Ichikawa: “Optimizing the configuration of a heterogeneous cluster with multiprocessing and execution-time estimation “ , Parallel Computing 31,pp.691-710(2005)

[2] 岩渕寿寛，杉田秀，山名早人: “ MPIETE2:MPI プログラム実行時間予測ツール MPIETE の通信予測誤差に関する改良 ”，情報処理学会研究報告 (HPC)，No.19，pp.175-180(2005)

[3] 姫野ベンチマーク: http://w3cic.riken.go.jp/HPC/HimenoBMT/