プログラム並列化におけるデータ分割支援システム

6D－6

三吉郁夫†, 森眞一郎†, 中島浩†, 富田眞治†

†: 京都大学工学部

1 はじめに

並列化コンパイルの重要なフェーズの1つであるデータ分割戦略の決定は、多くの場合ユーザに委ねられている。この戦略の適否が並列化によって得られる加速率を大きく左右するが、並列実行の挙動を予測するのは必ずしも容易ではないため、ユーザが最適な戦略を指定できるとは限らない。また、並列化対象のコードを書いたプログラマ以外の者が並列化を試みる場合には、最適な戦略の選択はさらに困難になる。

そこで本研究では、メッセージ交換型の分散メモリ並列計算機を対象に、ユーザが指定したデータ分割戦略の適否を判断するための支援システムを開発した。このシステムでは逐次プログラムの実行トレースを解析し、加速率を決定する主な要素であるプロセッサの稼働状況とメッセージ通信の発生タイミングを、適切な時間軸に基づいて推定する。これらの情報はビジュアルアナライザを介してユーザに提供され、並列化されたプログラムの挙動や性能の予測に用いられる。

2 方針

2.1 前提条件

1. 対象とする計算機はメッセージ交換型の分散メモリ並列計算機とする。
2. 対象とするプログラムは SPMD(Single Program Multiple Data stream) モデルに基づくものとし、データ並列性を内在するループが並列化されるものとする。
3. 分割される共有データ (以下分配データと呼ぶ) は、ユーザが指定した戦略に基づいて個々のプロセッサに割り付けられる。
4. 分配データをアクセスするループボディの実行は、Data Owner Computes Rule によって定まるプロセッサが担当する。

Supporting Tool for Data Partitioning in Parallel Programming

Ikuo Miyoshi†, Shin-ichiro Mori†, Hiroshi Nakashima†, Shinji Tomita†

†: Faculty of Engineering, Kyoto University

2.2 トレースデータの収集と解析

解析に必要な情報は、逐次プログラムを実行し、分配データへのアクセスをトレースすることによって収集する。このトレースデータからは以下の情報を求めることができる。

- プロセッサの稼働状況
  Data Owner Computes Rule に基づいているので、分配データの書き込みアドレスからループボディを実行するプロセッサが求められる。したがって特定のプロセッサが、並列化対象のループボディのどのイタレーションを実行するかを知ることができる。
- メッセージ通信
  ループボディを実行するプロセッサと、それが読み出す分配データのアドレスから、メッセージ通信の有無を知ることができる。また同一データの再送信の除去など、コンパイラによる通信の最適化を再現することもできる。

2.3 時間軸の設定

並列化されたプログラムの挙動をわかりやすく整理してユーザに示すためには、並列実行において基準となる、適切な時間軸を設定することが必要である。個々のプロセッサが非同期的に動作する並列計算機においては、同期処理に挟まれた区間を時間軸の単位として考えるとわかりやすい。そこで本システムでは、ループのイタレーション毎に同期処理を行なうことを仮定し、これを時間軸の単位として用いる。

3 実装

実装したシステムは以下に示す4つのモジュールから構成されている。

1. トレースジェネレータ
   与えられた逐次プログラムに、分配データへのアクセスをトレースするためのコードを付加し、これを実行することによってトレースデータを収集する。

2. ディストリビュータ

ユーザの指定したデータ分割戦略に基づき、トレースされた各々のアクセスがどのプロセッサで実行されるかを定める。

3. コミュニケーションアナライザ

ディストリビュータからの出力をもとに、メッセージ通信をシミュレートする。また同一データの再送信の除去など、メッセージ通信の最適化も行なう。

4. ビジュアルアナライザ

コミュニケーションアナライザからの出力を可視化し、プログラムの挙動としてユーザに示す。横軸にプロセッサ、縦軸下向きにプログラムの処理の進行をとり、ループの各イタレーションを単位として、以下の情報を色分けして表示する (図 1)。

- プロセッサの動作状態
  データ受信を伴わない実行、データ受信後実行、待機
- データ送信
  実行後データ送信、データ送信なし

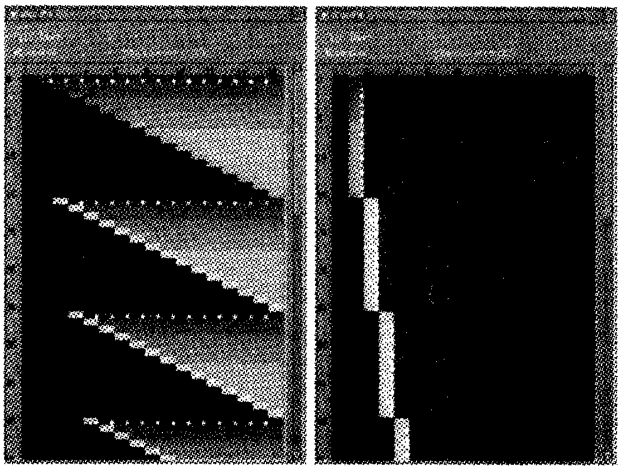
なお「データ受信後実行」に関しては、送信と受信の時間的な距離を色の階調で示しており、受信待ちによってレイテンシが生じる可能性を予測できる。

図 1の例では、黒い部分が「待機」、その他の部分が「実行」を表し、後者については階調が淡いほどデータ送受信間の時間的な距離が大きいことを示している。一方、データ送信の発生は、(a) の最上段に見られるような白いマークを用いて表している。

4 プログラムの挙動予測の具体例

コレスキー分解の並列化を例に示す。図 1は 16 × 16 行列の分解を 16 プロセッサで並列化する場合の出力結果である。これより以下のような情報が得られる。

- 行方向による分割の場合、実行状態のプロセッサ数は処理の進行にしたがって逆三角形の形で極大→極小の変化を繰り返す。一方通信については、逆三角形の底辺のイタレーションが送信を行ない、斜辺を除いたイタレーションが受信を行なう。またこの送受信間の距離は、底辺から離れたイタレーションほど大きくなり、データの受信待ちが生じにくくなる。
- 列方向による分割の場合、実行状態のプロセッサ数は処理の進行にしたがって三角形の形で極小→極大の変化を繰り返す。一方通信については、垂直な辺のイタレーションは送信のみを、その他のイタレーションは受信のみを行なう。またこの送受信間の距離は、全ての受信において最小値 0 をとるため、データの受信待ちが生じる可能性が高い。
- イタレーション数で計った処理終了までの時間、およびメッセージ通信の発生回数は両分割とも等しい。



(a) 行方向による分割　　(b) 列方向による分割

図 1: ビジュアルアナライザの出力例 (コレスキー分解)

これらより、データ受信待ちのレイテンシの影響で、行方向の分割が有利であると予測できる。この予測をもとに、並列計算機 AP1000 を用い、実際に 256 × 256 行列の分解を 16 プロセッサで行なった結果、行方向の分割で 8.7、列方向の分割で 5.6 の加速率が得られた。

5 おわりに

本稿では逐次型プログラムの並列化の際に必要となる、ユーザによるデータ分割戦略の決定を支援するためのシステムについて述べた。本システムは逐次型ソースプログラムとデータ分割戦略をもとに、並列化されたプログラムの挙動を、プロセッサの動作状態とメッセージ通信の発生の両面からユーザに示すことができる。

今後の課題は、並列化されたプログラムの挙動予測の精度を向上することである。具体的には、本システムで得られる情報に、実行時間の予測などの定量的な指標を加味し、各種の事象の発生するタイミングをより正確に求めることを考えている。

参考文献

[1] 山本 富士男: 超並列機向けデータ分割の自動評価方式, JSPP'92 論文集, pp.407-414, 1992 年 6 月.