超並列マシン RWC-1 における再現実行方式の検討

4H－8

市吉 伸行　関田大吾　西岡利博　吉光 宏
技術研究組合 新情報処理開発機構 超並列 MRI 研究室

概要

並列マシンにおけるバグの非再現性への対策として、再現実行（リプレイ）が有効と考えられている。これまでに幾つかの方式がメッセージ通信型並列マシン向けに提案されているが、それらは明示的な送受信プリミティブを用いたプログラム向けのものであった。一方、RWC プロジェクトで開発中の細粒度超並列マシン RWC-1 で採用している RICA アーキテクチャでは、受信側には明示的な受信タイミングを設定せず、メッセージパケットが能動的にスレッドを起動する通信モデルとなっている。本稿では、RWC-1 における再現実行実現方法についての検討内容を報告する。

1　再現実行機能

超並列 MRI 研究室では、RWC プロジェクトの一環として超並列プログラミング環境の研究開発を行なっている [5]。RWC プロジェクトで開発中の超並列マシン RWC-1 向けのデバッグ／プロファイル支援ツールの開発を当面と目標として、並列プログラム開発で直面する問題を検討している。

逐次プログラムのバグ追求においては、バグが顕在化する箇所の手前の１つないし複数の適当な箇所にブレークポイントやトレースポイントを設定して、途中状態を検証することによってバグ発生場所を狭めるという「繰り返しデバッグ (cyclic debugging)」が通常用いられる。

ところが、並列プログラムでは、要素プロセッサの非同期性に起因する動作の非決定性があるために、バグ状態が必ずしも再現せず、上記の技法の適用が難しい。再現実行（再演 (replay) とも呼ぶ）とは、プログラム実行の非決定性を記録し、後でそれを再現させることである。再現実行の研究は従来からなされてきており、ターゲットとなる並列マシンが密結合共有メモリ型であるもの、疎結合メッセージパッシング型であるもの、など幾つかの方式が提案されている [3]。

単一プロセッサ内の実行は決定的なので、外部からの（非同期の）影響 — プロセッサ内部処理のどの時点でどのような外部起源の事象が起こったか — を記録すればよいというのが、実現方式の基本的な考え方である。しかも、再現された初期状態からスタートして、並列マシンの全系で同時並行して再現実行を行なうと、各プロセッサの生成するデータ／メッセージの内容も再現されるため、タイミングのみを記録すれば十分で、データの記録は不要となる（計算機外部との相互作用は別に扱う必要がある）。再現時には、記録した各々の非同期事象の生起タイミングで、その外部事象を待ち合わせるようにする（外部事象が先に到着した場合は、その内部への影響を然るべき時点まで延期する）。

しかし、実現の難しさおよびオーバヘッドの問題などにより、再現実行機構が実際のプログラミング環境で実現された例はまだ少ない。

2　超並列マシン RWC-1

RWC プロジェクトで開発中の RWC-1 [4] は、分散メモリ、MIMD 型の超並列マシンである。特に、細粒度プログラムの効率良い実行を支援するために、命令実行と通信処理とを融合させた RICA アーキテクチャを採用している。RICA では、単純、汎用、かつ高速なハードウェア機構を実現し、並列プログラミングのためのより複雑な機能はソフトウェアで柔軟に実現しようというアプローチを取る。具体的な特徴としては、低レイテンシで大容量のネットワーク、プロセッシングノードにおける、高速同期機構などのマルチスレッドのサポート、命令実行と並列に動作するパケット送信パイプライン、パケットによる能動的なスレッド起動、などがある。

RWC-1 は、単一プロセッサからなるノードが全体で数千程度結合されたシステムとなる予定である。パケットおよびスレッドには４レベルの優先度があり（システムレベル２つ、ユーザレベル２つ）、高い優先度のパケット到着などに低い優先度のスレッドがプリエンプトされる。RWC-1 のノードには、入力パケットのキューがあり（原則として FIFO スケジューリング）、実行中のスレッドが終了した時点で、先頭のパケットが次のスレッドを起動する。パケットの生成は自ノード宛のローカルなものも、他ノード宛のリモートのものも同一の命令（mkpkt 命令）によって行なわれる。

スレッドは一旦起動すると、break 命令実行による自発的に終了するまで走り続ける。ただし、より高い優先度のスレッドによるプリエンプションやページフォルトによる間欠的な中断が起こり得る。

Execution Replay Mechanism for the Massively Parallel Machine RWC-1

Nobuyuki ICHIYOSHI, Daigo SEKITA, Toshihiro NISHIOKA, and Hiroshi YOSHIMITSU
Massively Parallel MRI Lab., Real World Computing Partnership
2-3-6 Otemachi, Chiyoda-ku, Tokyo 100, JAPAN

3 再現実行方式の検討

3.1 非同期事象タイミングの記録方法

send/receive プリミティブを持つメッセージ通信型並列マシン向けの再現実行には、「何番目の receive」という形でタイミングを記録する方式の事例がある [1]。しかしながら、RWC-1 の通信方式では受信側は明示的な受信タイミングを設定せず、メッセージパケットが能動的にスレッドを起動するようになっており（active message [2] でも同様）、パケット到着タイミングの別の記録方法を考える必要がある。（筆者らの知る限り、このような通信モデル向けの再現実行方式は発表されていない。）

検討中の方式では、外部からのパケット到着は、実行中のスレッドをプリエンプトするか／しないかで区別して、別々の方法で記録するようにする。

- スレッドをプリエンプトしないもの

  スレッド番号をメンテナンスし、「何番目のスレッドは何番プロセッサからのパケットで起動された」という情報を記録する。（通信が FIFO でない場合、「何番プロセッサからの何番目のパケットか」という情報も必要。）これは、外部起源のスレッドの先頭に記録用コードを挿入することで行なう。

- スレッドをプリエンプトするもの

  プリエンプションやページフォルトのタイミングの記録は、プリエンプトした側のスレッドやページフォルト・ハンドラが行なう。関数呼び出しやループなどによりプログラムコード中の命令は一般に複数回実行されるため、スレッド実行中のタイミングの特定にはプログラムカウンタ値だけでは不十分である。プログラム実行開始からの命令実行数を表わす命令カウンタをプロセッサに導入する方式やソフトウェア命令カウンタを導入する方式が考案されているが、ハードウェアに負担をかけないために後者を採用する方針である。

3.2 記録オーバヘッドと記録量

記録実行には、原理的な実現方法の問題の他に、オーバヘッドと記録データ量の問題がある。

試算によると、記録実行のオーバヘッドは通常実行の数割増し程度と思われる。一方、記録データ量は 1 プロセッサ当たり 1 秒間に数Mバイト程度が予想され、削減が大きな課題である。なお、記録データはメモリのログ領域に書き貯めておき、RWC-1 のプロセススイッチ機能を用いて、定期的に全系を止めてディスクにダンプする方式を考えている。これは、記録のディスク出力処理が再現しようとする実行に及ぼす擾乱を最低限に抑えるためである。

3.3 再現モード実行

再現モード実行では、通常のデバッガにあるようなブレークポイントやトレースポイント設定機能の他、イベントトレースや性能プロファイリングのためのプローブの挿入も可能であることが望ましく、そのための方法を検討中である。再現実行では、非同期事象をソフトウェア的に待ち合わせるためのオーバヘッドが大きく、記録実行時の数倍から十倍程度の速度低下が予想される。プローブの挿入によりさらに遅くなるが、それによりユーザプログラムの振舞いが変わらないのが、再現実行メカニズムの上にそれらを載せるメリットである。（ただし、性能データは一部しか再現できないであろう。）

4 おわりに

超並列マシン RWC-1 向けに検討中の再現実行メカニズムについて述べた。非同期並列プログラムのデバッギングの最大の問題点の一つは、再現実行メカニズムによって解決される。しかし、原理的には可能であるものの、オーバヘッドや記録データ量の問題があり、また、それらを解決／緩和したとしても、OS やコンパイラへの影響を局所化する工夫も必要であろう。

当研究室では、今後も継続して上記の問題点への対策を検討して行く予定である。また、それと並行して、超並列プログラムのデバッグのための違った手法も検討して行きたいと考えている。

参考文献

[1] Eric Leu, André Schiper, and Abdelwahab Zramdini. Efficient execution replay technique for distributed memory architectures. In *2nd European Distributed Memory Computing Conference (LNCS 487)*. Springer-Verlag, 1991.

[2] Thorston von Eicken, David E. Culler, Seth Copen Goldstein, and Klaus Erik Schauser. Active messages: a mechanism for integrated communication and computation. In *Proceedings of the 19th International Symposium on Computer Architecture*, 1992.

[3] 山田剛. 並列処理システムにおけるプログラムデバッギング. 情報処理, Vol. 34, No. 9, 1993.

[4] 坂井修一, 他. 超並列計算機 RWC-1 の基本構想. 並列処理シンポジウム JSPP'93, pp. 87–94, 1993.

[5] 市吉伸行, 他. 超並列プログラミング環境の検討. 第 48 回情処全大 4H-5, 1994.