

適応的再演型ロック命令を用いた並列プログラムデバッガ

6D-9

三栄 武 高橋 直久

NTT ソフトウェア研究所

1 はじめに

メモリ共有型の並列プログラムは、共有データへのアクセス動作が非決定的となるため、プログラムの動作理解やデバッグが逐次型プログラムと比べて難しい。この問題を解決するため、適応的再演型ロック命令と呼ぶロック命令と、これを用いた並列プログラムデバッガを提案し、その実現法について述べる。

2 再演法を用いた並列プログラムデバッグ

並列プログラムでは、共有オブジェクトへのアクセス順序が実行毎に非決定的であるため、実行停止位置や変数値などの状態が変化してしまう。この問題に対し、並列プログラムの実行動作を記録(実行記録と呼ぶ)し、この記録に従って実行を制御することで、繰り返し同じ動作を得る手法¹⁾²⁾(再演法と呼ぶ)が提案されている。これらの再演法を用いた並列プログラムのデバッガを実現するには、次のような課題がある。

(1) 実行記録上での停止位置の設定と最小限再演 多くのデバッガでは、静的なソースコード上での実行停止位置(ブレークポイント、BP と略す)を設定するコマンドを提供している。再演法を用いたデバッグでは、保存した実行記録を利用して、停止位置を実行動作上に対しても直接的に設定できることが望ましい。

また、BP に到達したプロセスが実行を停止しても、その他のプロセスは可能な限り実行を進めてしまう。このため、プログラマが新たに BP を設定した命令が既に実行済みである可能性が高く、この場合、再演を初めからやり直す必要がある。よって、実行記録上に停止位置を設定し、その停止位置までの再演に必要最小限な実行を抑えることが望ましい。

(2) 再演停止法 あるプロセスが BP に到達した時に、単純に他のプロセスを停止させると、停止位置が非決定的となる。全てのプロセスが実行を進めない状態(安定

状態と呼ぶ)にあることを保証した上で、できるだけ早い時点で再演を停止させる必要がある。

(3) 実現法 停止位置の設定、最小限再演や再演停止法を実現するには、常に全プロセスの実行制御が必要となる。これらの制御機能を全てデバッガに集中させると、プログラムの並列性を損なうことになる。また、種々のデバッガへの組み込みを容易にするために、プログラム言語やデバッガコマンドプロセッサに依存しない実現法が望ましい。

上記(1)~(3)について、各々3~5章で述べる。

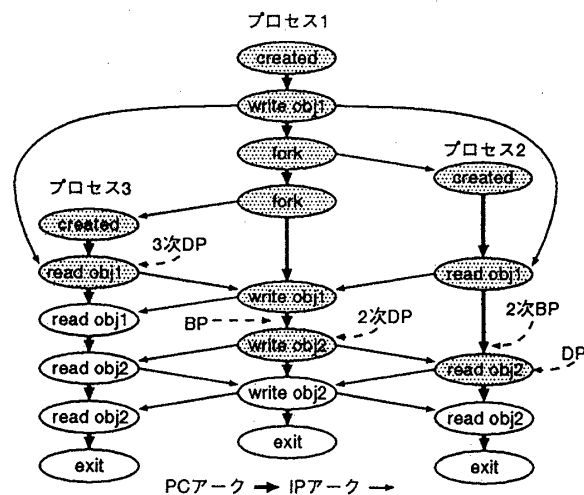


図1: 要求駆動による再演実行

3 要求駆動型再演による最小限実行

図1は、2つの共有オブジェクトにアクセスする3つのプロセスからなるプログラムの実行動作の一例を表している。各ノードは非決定的動作を生じる命令を表し、各アークはノードの実行順序を表す。PCアークは、一つのプロセス内での実行順序を表す。IPアークは、プロセス間での実行順序を表す。

プログラマが実行記録上に設定した停止位置をデマ

Parallel Debugger with Adaptively Replayable Lock.
 Takeshi MIEI (take@square.ntt.jp),
 Naohisa TAKAHASHI (naohisa@ntt-20.ntt.jp),
 NTT Software Laboratories.

ンドポイント (DP) と呼ぶ。DP だけが与えられ BP が設定されていない場合には、要求駆動再演法³⁾を用いると、以下のように DP に到達するために必要な命令だけが実行される。この手法ではまず、DP から IP アークを遡って到達できるノード (2次 DP と呼ぶ) に実行要求信号を送る。各プロセスは PC アークに沿って実行を開始し、実行要求を受けたノードを全て実行した後、停止する。各ノードでの実行は、まず、IP アークの根側のノード (先行ノードと呼ぶ) に実行要求信号を送り、その実行終了信号を待機する。その後、ノードの再演実行を行ない、IP アークに沿って実行終了信号を送る。再演中に実行要求信号を受けたノードを 3次 DP と呼ぶ。図 1 では、再演されたノードを網掛けで示している。

4 安定状態に基づく再演停止

プロセスが実行を進めないことが保証される実行位置を、そのプロセスの SP と呼び、全プロセスが各々の SP まで到達した状態を安定状態と呼ぶ。あるプロセスが DP に到達すれば安定状態であると保証される。BP を併用し DP に到達しない場合には、以下の手法により、各プロセスが各々独立に SP 到達を判定できる。

プロセスは 2次 DP、3次 DP、BP、または BP で停止したプロセスを待ち合わせる実行位置 (2次 BP と呼ぶ) で SP 到達となる。各プロセスに与えられた実行要求の数 (デマンド数と呼ぶ)、IP アーク、DP または 2次 DP を持つプロセス (DP プロセスと呼ぶ) の数 (DP プロセス数と呼ぶ) を再演前に設定しておき、SP 到達判定に用いる。プロセスは実行要求されたノードを実行する毎にデマンド数を 1 減じる。DP プロセスは、デマンド数が 0 になれば 2次 DP 到達であり、SP 到達として DP プロセス数を 1 減じる。DP プロセスでない場合、デマンド数と DP プロセス数がともに 0 になれば、SP 到達となる。また、実行終了信号を待つ際に、IP アークより求まる先行ノードのプロセスが、BP または 2次 BP に到達すれば 2次 BP 到達であり、SP 到達となる。

図 1 に、プロセス 1 の write obj2 の前に BP が表れた場合の再演を示す。この時、プロセス 2 は read obj2 を実行前に停止し (2次 BP)、プロセス 3 は read obj1 を実行後に停止する (3次 DP)。

5 適応的再演型ロック命令による実現

前章までに述べたプログラムの実行制御と停止制御の機能を、適応的再演型ロック命令と呼ぶ排他制御命令に埋め込み、図 2 のようにデバッグシステムを実現する。以下に、このロック命令のプログラム再現時の機能を示す。

(1) 要求駆動再演機能 実行記録に従った順序で、かつ実行要求を受けたアクセスのみを許可するため、実行要求信号を受信後、先に実行すべきアクセス命令に実行要求信号を送り、実行終了信号を受けるまで待機する。

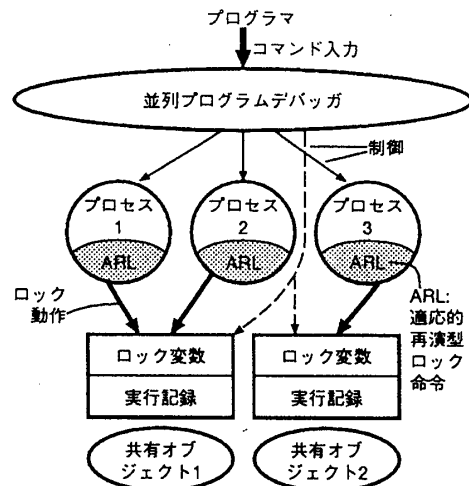


図 2: 適応的再演型ロック命令による実現

(2) 状態適応機能 2次 BP への到達を検出するために、適応型ロック命令⁴⁾の監視手法を適用し、実行終了信号を待機する際に、先行ノードのプロセスの監視を行ない、その状態に合わせて自らの状態を変化させる。

(3) SP 判定機能 各プロセスに与えられた実行要求の数と DP、2次 DP を持つプロセス数を管理し、前述の SP 到達条件の判定を行なう。

6 おわりに

メモリ共有型の並列プログラムをサイクリックにデバッグ可能な並列プログラムデバッガを提案し、その実現法について述べた。現在、プロトタイプを作成中であり、デバッガの基本機能が稼働している。

謝辞 日頃、御指導、御支援いただき後藤部長をはじめソフトウェア基礎技術研究部の皆様に感謝します。

参考文献

- 1) Carver R.H., et al., "Reproducible Testing of Concurrent Programs Based on Shared Variable," Proc. 6th Int. Conf. Dist. Comp. Sys., pp.428-433, May 1986.
- 2) LeBlanc T.J. et al., "Debugging Parallel Programs with Instant Replay," IEEE Trans. Comp., C-36, 4, pp.471-482, 1987.
- 3) 高橋直久, "データ共有型並列プログラムの要求駆動型再演システムの実現と評価", JSPP '90, pp.361-368, 1990.
- 4) 三栄武, 高橋直久, "適応型ロックを用いた並行プロセスのコスケジューリング機構の実現法", 情報 OS-53-3, 1991.