2ZB-5 PC クラスタにおける共有メモリの実装実験(4)
―ロック実装の違いによる性能への影響―

成蹊大学大学院工学研究科情報処理専攻
大橋 祐介･緑川 博子･飯塚 肇

1. はじめに

筆者らは、クラスタ型並列コンピュータシステム上に共有メモリプログラミング環境を実現する、ソフトウェア分散共有メモリシステムの設計、実装を行ってきた。すでにバリア、ロック、条件変数などの同期機構を提供する分散共有メモリシステム SMS を構築し、その概要について報告した[1] [2] [3]。

今回、複数の方式でロックの実装を行い、数種類の基本通信パターンにおける性能の優劣、さらに実際の応用プログラムにおける性能評価を行ったので報告する。また、そのほかの実装上の工夫についても簡単に説明する。

2. SMS のメモリコンシステンシー

SMS は筆者らが新たに考案した Implicit Binding Entry Consistency (IBEC)[3]というコンシステンシーモデルを採用している。また、実装はアップデート方式となっており、ホームベースではない。そのため、ロック獲得時にはそのロック id に関連づけられた更新情報(diff)の転送が発生する。

3. ロック実装の違い

従来のロック実装方式を図 1 に示す。この方式では、そのロックに関わる diff はそのロックの最後の獲得者が持っている。そのため、ホームベースの DSM のように diff をいったんホームに送るといった処理は必要ないが、バリアインターバル間でそのロックで作成されたすべての diff を送信するため、ロック獲得パターンによっては効率の悪い場合がある。図 1 のように、2 プロセスが交互に同じロックを獲得する場合、write(3)では write(2)の diff だけあればよいにも関わらず、write(1)、write(2)の両方の diff が転送されてしまう。

これに対して、本研究で新しく考案した実装方式を図 2 に示す。この方式では、ロック解放時に diff はロックマネージャに送られ、次のロック獲得時には、ロックマネージャは必要な diff のみを獲得者に送信する。この方式では、いったん diff をロックマネージャに送るという処理が加わるが、前節で述べたような場合でもロックのたびに不要な diff が増えていくということはない。

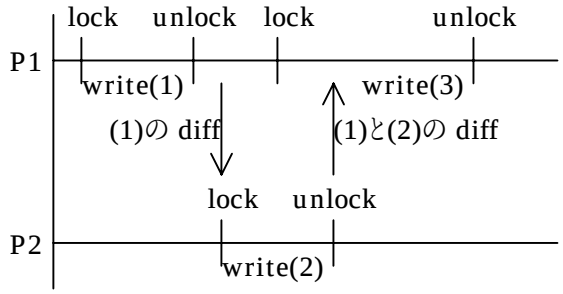

図 1:従来のロック実装方式

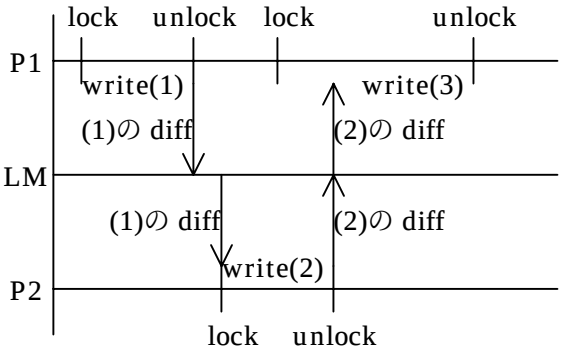

図 2:新しいロック実装方式

4. ページモードの切り替え

SMS では SEGV シグナルを減らすため、ページマネージャのみからしかアクセスされないと判断したページに関しては、ページプロテクションをかけない。この実装についての評価も以下で述べる。

5. 性能評価

5.1. 評価環境

評価は、表 1、表 2 のような 2 種類の環境で行った。

表 1:評価環境 1

CPU	Intel Celeron 400MHz
メモリ	128MB
ネットワーク	Intel PRO/1000 T 3Com SuperStack 3 Switch
OS	TurboLinux Server 6.1
台数	1～8

Implementation of new lock mechanism for Shared Memory System on PC Cluster
Yusuke Ohashi, Hiroko Midorikawa, Hajime Iizuka
Department of Information Sciences, Seikei University

表 2:評価環境 2

CPU	Intel MMX Pentium 166MHz
メモリ	64MB
ネットワーク	Intel PRO/100+ corega FSW-16L
OS	FreeBSD 2.2.8-RELEASE
台数	1～8

5.2. 実装の違いによる性能差

図 3 は、評価環境 1 において、1 つのロック変数を用いて、lock/unlock 間で 1 変数のみを変更する処理を 2 プロセスで 1000 回繰り返した場合の性能差である。図の(1)は、2 つの作業プロセス中 1 プロセスがロックマネージャを兼ねた場合で、(2)は、2 つの作業プロセス以外のプロセスがロックマネージャの場合である。(1)と(2)の差は、ロックマネージャとの通信が発生するかしないかである。従来方式にとって最も都合の悪い、このような通信パターンの場合、1.6 倍から 2.2 倍の速度向上が見られた。また、8 プロセスで各プロセスがそれぞれ 8 つのロック変数繰り返し lock/unlock した場合の性能差を図 4 に示す。この場合においても新方式のほうが優れていること、またロックマネージャをロック id ごとに分散させたほうがより効果があることがわかる。

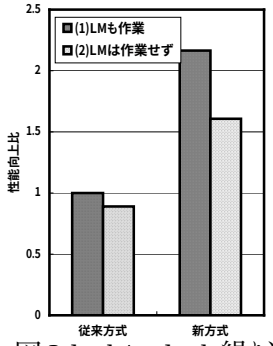

図 3:lock/unlock 繰り返し(2 プロセス)

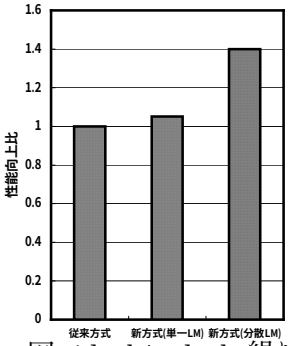

図 4:lock/unlock 繰り返し(8 プロセス)

5.3. 応用プログラムによる評価

この節では、評価環境 2 におけるいくつかのベンチマークの性能を、代表的なソフトウェア DSM である TreadMarks (Version 0.10.1.2)と比較する。使用したベンチマークは、NPB の EP、SPLASH-2 の WATER、TreadMarks の SOR、筆者らが作成した多体問題、マンデルブローなどである。8 台実行時の場合の性能向上は、EP が 6.6 倍、多体問題が 5.8 倍、マンデルブローが 5.8 倍で、いずれも TreadMarks と同程度であった。ここでは、WATER と SOR について細かく見ることにする。

図 5 は、WATER の性能向上比である。この問題は、新方式ではメモリ消費量が大幅に減り、性能も TreadMarks に並ぶ性能を見せている。8 台実行時のときの性能向上比は 6.0 倍となった。

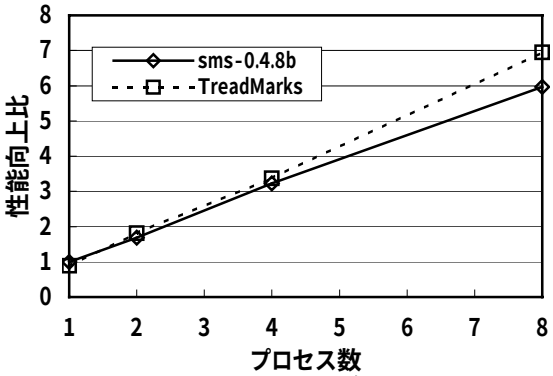

図 5:WATER による評価(1728mole)

図 6 は SOR における性能向上比である。この問題のように、大量のページを使用し、かつページアクセスのローカリティが高い問題の場合、SEGV シグナルを減らすことが非常に重要になってくる。8 台実行時の場合、SEGV シグナルの発生を抑制することで、抑制前の 1.5 倍の速度向上が見られ、TreadMarks と同等以上の性能となった。

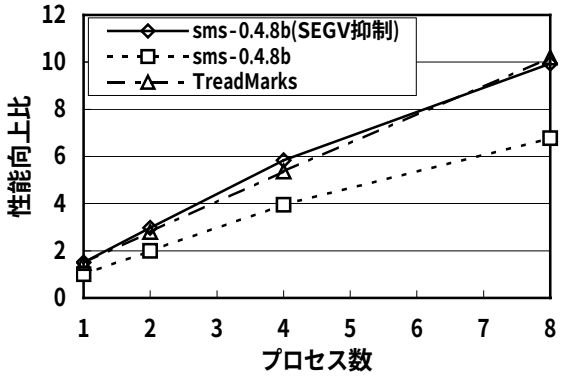

図 5:SOR による評価(1024*1536)

6. おわりに

分散共有システム SMS におけるロック実装の違いによる性能の影響ついて述べた。通信パターンによっては従来方式のほうがよい性能を上げるが、多くの実際的な問題では、diff を 1 プロセスで集中管理するほうがよいということがわかった。また、さらなる性能向上のために、細かい実装上の無駄をなくす予定である。

参考文献

[1]伊藤, 緑川, 飯塚:"PC クラスタにおける共有メモリの実装 (2) -ロックの実装-",情処全国大会 2H-2(2000,3)

[2]大橋, 緑川, 飯塚:"PC クラスタにおける共有メモリの実装 (3) -UDP による実装-",情処全国大会 8Q-08(2001,3)

[3]緑川, 飯塚:"ユーザレベル・ソフトウェア分散共有メモリ SMS の設計と実装",情報処理学会論文誌ハイパフォーマンスコンピューティングシステム Vol.42,No.SIG9(HPS 3),pp.170-190(2001,8)