ソフトウェア分散共有メモリシステムにおけるページ転送方式の比較

原田 浩[†] 手塚宏史[†] 堀 敦史[†]
住元真司[†] 高橋俊行[†] 石川 裕[†]

分散共有メモリシステムで採用されている2種類の一貫性維持プロトコルである更新プロトコル，無効化プロトコルの実装において，メッセージ通信，ゼロコピー通信を用いたページ転送方式を採用することによる性能の違いを検証するのが本論文の目的である．Pentium PRO 200 MHz，Myrinetネットワーク上のPM通信ライブラリによるゼロコピー通信による1ページ（4 Kbytes）の転送時間は，メッセージ通信の2.46倍高速であるが，行列のラプラシアンを求めるベンチマークを64ノードで実行した結果では，更新プロトコル，無効化プロトコルともにゼロコピー通信による性能向上効果はほとんどない．SPLASH2のLUを64ノードで実行した結果では，更新プロトコル，無効化プロトコルともにゼロコピー通信による性能向上は2.61倍，1.24倍となる．

Camparison of Page Transfer Method on Software Distributed Shared Memory System

Hiroshi Harada,[†] Hiroshi Tezuka,[†] Atsushi Hori,[†]
Shinji Sumimoto,[†] Toshiyuki Takahashi[†] and Yutaka Ishikawa[†]

Two page coherency protocols, update and invalidation protocols, are evaluated using the zero-copy and message passing communication mechanisms in this paper. The one page, 4 Kbyte message, transfer time using the PM zero-copy communication mechanism is 2.46 times faster than that of using the PM message passing mechanism in Pentium PRO 200 MHz with the Myrinet network. The result of the Laplace program running on 32 nodes shows no performance gain in both update and invalidation protocols using the zero-copy communication mechanism. However, the result of the LU application of SPLASH2 running on 32 nodes shows that update and invalidation protocols using the zero-copy communication mechanism achieve 2.61 and 1.24 times faster than that of using the message passing mechanism, respectively.

1. はじめに

ソフトウェア分散共有メモリ（以下，分散共有メモリはDSMと記述する）のページ単位の一貫性維持プロトコルとして，メモリバリア時にページデータを送信しページの更新を通知する更新プロトコルと，メモリバリア時にページ無効化メッセージを送信し，ページにアクセスが行われてからページを転送する無効化プロトコルの2つのプロトコルが主に採用されている．

また，クラスタ計算機のノード間接続のための高速ネットワークの研究では，従来のメッセージ通信のほかに，メモリコピーのオーバヘッドを削減しMyrinet等のギガビットネットワークの性能を発揮させることを目的に，送信元のユーザ空間から送信先のユーザ空間へDMA転送を行うゼロコピー通信の研究が行われている．ゼロコピー通信を用いたページ転送を実現することによって，ソフトウェアDSMシステムの性能向上を図ることが期待できる．しかし従来のメッセージ通信によるページ転送との比較で，ゼロコピー通信によるページ転送が，更新プロトコル，無効化プロトコルの2つの一貫性維持プロトコルに対して，それぞれどの程度性能に影響を与えるかは，明らかにされていない．

本論文の目的は，従来から提案されてきた2種類の一貫性維持プロトコルにおいて，メッセージ通信，ゼロコピー通信の2種類のページ転送方式を実装し，実際にアプリケーションを用いて比較，評価を行うことにより，それぞれの一貫性維持プロトコルに対して，ページ転送方式が性能に与える影響を明らかにすることにある．

評価環境として，ギガビットネットワークの1つであるMyrinet[1]上に低通信遅延かつ高通信帯域幅を提供する高速通信ライブラリPM[2]を用いて実現して

† 新情報処理開発機構つくば研究センタ
Tsukuba Research Center, Real World Computing Partnership

いるSCASH[3),4)]と呼ぶソフトウェア分散共有メモを採用する．SCASHは，メモリモデルとしてRelease Consistencyを採用しているため，バリア同期時に共有メモリの一貫性維持を行う．本論文では，バリア同期をとるだけのバリア同期機構と区別するため，メモリの一貫性維持を含むバリア同期機構をメモリバリアと記述することにする．

SCASHは，更新プロトコルと，無効化プロトコルの2つの一貫性維持プロトコルを実現している．また，SCASHの通信ライブラリである，PMには従来のメッセージの送受信による通信機能に加えて，相手ノードのユーザ空間に対してメモリを書き込む遠隔メモリ書き込みと，相手ノードのユーザ空間からメモリを読み込む遠隔メモリ読み込みの2種類のゼロコピー通信機能が実現されている．2種類の一貫性維持プロトコルに対して，2種類のゼロコピー通信によるページ転送方式が性能に与える影響を比較，評価する．

基本評価として，ページ転送時間，メモリバリア，ページフォールトハンドラのコストを測定，比較する．次に実際にアプリケーションを用いた評価として行列のラプラシアンを求めるLAPALCEとSPLASH2[5)]からLUを用いて実行時間とその詳細を測定，比較する．

その結果，従来のメッセージ通信によるページ転送との比較で，メモリバリア，ページフォールトハンドラにおけるページ転送速度でそれぞれ2.46倍，1.95倍の高速化を達成した．SPLASH2のLU（CONT）ノード数64台における実行速度で，更新プロトコル，無効化プロトコルでそれぞれ2.61倍，1.24倍の性能向上を得た．

本論文の構成は以下のとおりである．2章では，SCASHの概要とページフォールトハンドラ，メモリバリアの実行手順について述べる．3章では，メッセージ通信，ゼロコピー通信を用いたページ転送の実現と問題点について述べる．4章ではページ転送，メモリバリア，ページフォールトハンドラの基礎評価と，メモリバリアで記述された共有メモリアプリケーションを用いてSCASHの評価を行う．5章で評価結果を示し，6章では評価結果を基に，SCASHのメモリバリア，ページ転送方式，問題点について検討する．7章で関連研究を紹介し，8章でまとめを述べる．

2. SCASH

2.1 SCASHの概要

SCASHは，オペレーティングシステムのメモリ管理機能を利用し，ユーザレベルのライブラリとして実現されている分散共有メモリシステムである．SCASHは，システムの初期化，共有メモリの割当て，解放，メモリバリア等の同期機構を提供している．

共有メモリ領域の一貫性維持は，オペレーティングシステムが提供するページ単位で行われる．一貫性モデルとしてRC（Release Consistency）[6),7)]を採用し，その実装にはマルチプルライタプロトコル[8)]を用いている．さらに，ページ単位の一貫性維持プロトコルとして，ページの無効化を通知する無効化プロトコルと，ページデータを送信し，ページの更新を通知する更新プロトコルを実装している．

SCASHではオペレーティングシステムの提供するページ保護機構とページアクセスによる例外処理機構（以下，ページフォールトハンドラと呼ぶ），およびメモリバリア等SCASHの提供する同期機構を用いることにより共有メモリ領域の一貫性維持を行う．

SCASHのオーバヘッドは，ページフォールトハンドラとメモリバリア等の同期機構の2つに大別できる．以下にページフォールトハンドラとメモリバリアの実行手順を示す．

2.2 ページフォールトハンドラの実行手順

SCASHでは，各ページの一貫性維持のために，オペレーティングシステムの提供するページ保護機構を用いて共有メモリ領域内の各ページに対するアクセスを検知している．具体的には，ページに対するアクセス（読み込み，書き込み）がページ保護に違反した場合，SCASHのページフォールトハンドラが起動され，一貫性維持に必要な手続きが行われる．ページフォールトハンドラは，共有メモリ領域内のページに対して読み込み違反，書き込み違反が検知された場合，以下を行う．

- 読み込み違反
  読み込み違反が検知された場合，ページの管理ノードからページデータをコピーし，ページの共有に参加する．さらに該当ページのページ保護を読み込み可に変更する．ページの管理ノードは，読み込み違反を起こしたノードを，ページディレクトリに登録する．
- 書き込み違反
  書き込み違反が検知された場合，まだページの共有に参加していない場合は，読み込み違反を検知した場合と同様に，ページ共有に参加する．さらに，マルチプルライタプロトコル実現のためページデータのコピーを作成する．ページ保護を読み書き可に変更する．

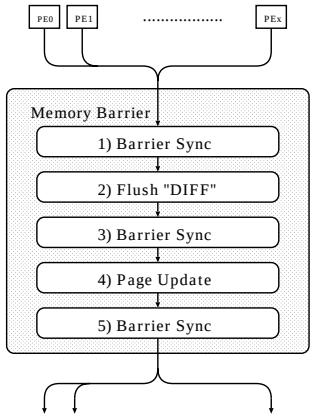

図 1 更新プロトコルを用いたメモリバリア
Fig. 1 Memory barrier using update protocol.

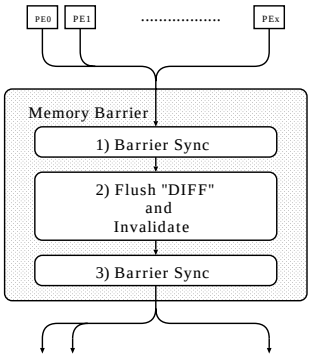

図 2 無効化プロトコルを用いたメモリバリア
Fig. 2 Memory barrier using invalidate protocol.

2.3 メモリバリア

SCASH のメモリバリアは，一貫性維持プロトコルとして，ページを共有しているノードに無効化メッセージを通知する無効化プロトコルと，ページデータを送信し，ページの更新を通知する更新プロトコルを選択できる．以下に 2 つの一貫性維持プロトコルを用いた場合のメモリバリアの実行手順を述べる．

2.3.1 更新プロトコルを用いたメモリバリア

更新プロトコルを用いたメモリバリアの実行の概要を**図 1** に示す．メモリバリアの実行手順は以下のとおりである．

(1) Barrier Sync
最初に全ノード間でバリア同期をとり，全ノードが，共有メモリへのアクセスを中断し，メモリバリアの実行開始を確認する．バリア同期は，PM のメッセージ通信を用いて実現している．

(2) Flush DIFF
全ノードで，書き込みが行われた全ページについて，書き込みが行われる以前のページと書き込みが行われたページを比較して，差分（DIFF）を作成する．作成した差分はそれぞれの該当ページのホームノードへ送信される．差分を送信した後，ページ保護を読み書き可から読み込み可に変更する．ページ保護を読み込み可に変更するのは，ページに対する書き込みアクセスによってページ例外処理ルーチンを起動させるためである．起動されたページ例外処理ルーチンは，ページの複製を作成し，ページプロテクションを読み書き可に戻す．

(3) Barrier Sync
すべての差分の送信が終了した時点でバリア同期をとる．

(4) Page Update
ホームノードは，差分を受信したすべてのページについて，ページディレクトリからページを共有しているノードを検索し，ページを共有しているノードのページを更新する．ページの更新には，メッセージ通信によるページ転送とゼロコピーによるページ転送を選択することができる．

(5) Barrier Sync
すべてのページを更新した後に，バリア同期をとって処理を終了する．

2.3.2 無効化プロトコルを用いたメモリバリア

無効化プロトコルを用いたメモリバリアの実行の概要を**図 2** に示す．メモリバリアの実行手順は以下のとおりである．

(1) Barrier Sync
最初に全ノード間でバリア同期をとり，全ノードが，共有メモリへのアクセスを中断し，メモリバリアの実行開始を確認する．

(2) Flush DIFF and Page Invalidate
全ノードで，書き込みが行われた全ページについて，書き込みが行われる以前のページと書き込みが行われたページを比較して，差分を作成する．作成した差分はそれぞれの該当ページのホームノードへ送信される．差分を送信した後，ページ保護を読み書き可から読み込み可に変更する．差分を受信したホームノードは該当ページを共有している全ノードに対して，無効化メッセージを送信し共有ノードのページを無効化する．

(3) Barrier Sync
すべての差分，無効化メッセージの送受信が終了した時点でバリア同期をとる．

3. ページ転送方法

一貫性維持プロトコルとして更新プロトコル，無効

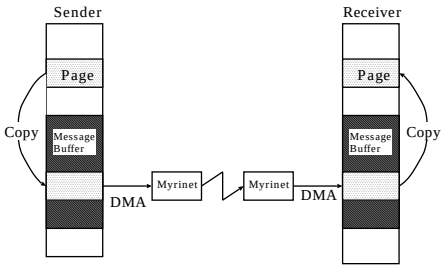

図 3 メッセージ通信によるページ転送
Fig. 3 Page transfer using message passing.

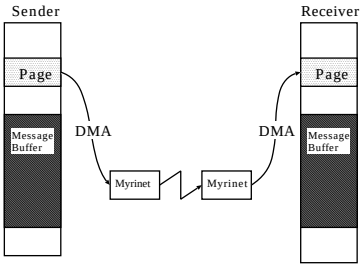

図 4 ゼロコピー通信によるページ転送
Fig. 4 Page transfer using zero copy communication.

化プロトコルを採用した場合，プログラムの実行中，およびメモリバリア実行中にページ転送が発生する．本論文では，以下の 2 つの通信方法によるページ転送を実現，比較する．

3.1 メッセージ通信

メッセージ通信によるページ転送は，ページデータを格納したメッセージを送受信することによってページを転送する方法である．メッセージ通信によるページ転送方法の概要を**図 3** に示す．

メッセージ通信によるページ転送では，送信側はページを通信バッファに，受信側は通信バッファからページにデータをコピーするオーバヘッドが生じる．

3.2 ゼロコピー通信

メッセージ通信によるページ転送では，送信側，受信側双方において，ページとバッファ間でメモリコピーを行う必要がある．メモリコピーのオーバヘッドを削減し，効率的なページ転送を実現する方法として，ゼロコピー通信によるページ転送方法がある．**図 4** に示すように，ゼロコピー通信によるページ転送では，送信側から受信側へページを直接 DMA 転送することによって，バッファを介さずに直接ページを転送することができる．PM では，遠隔メモリ書き込み，遠隔メモリ読み込みの 2 つのゼロコピー通信を実現している．

3.2.1 遠隔メモリ書き込み

遠隔メモリ書き込みは，要求元のユーザ空間から，相手先のユーザ空間へ直接メモリ転送を行う．遠隔メモリ書き込みを用いたページ転送は，一貫性維持プロトコルとして更新プロトコルを用いた場合のメモリバリア中のページ更新に用いることができる．

一貫性維持プロトコルに更新プロトコルを用いた場合，遠隔メモリ書き込みを用いたページ転送を実現することによって，ページデータをコピーするオーバヘッドを解消し，Myrinet の高帯域幅を活かした効率的なメモリバリアが実装できると考えられる．

3.2.2 遠隔メモリ読み込み

遠隔メモリ読み込みは，相手先のユーザ空間から，要求元のユーザ空間へ直接メモリ転送を行う．遠隔メモリ読み込みを用いたページ転送は，一貫性維持プロトコルとして更新プロトコル，無効化プロトコルの双方で，新たにページの共有に参加するノードが，ホームノードからページをコピーするのに用いることができる．遠隔メモリ読み込みを用いたページ転送によって，メモリフォールトハンドラの高速化が期待できる．

3.2.3 メモリのピンダウン

ゼロコピー通信を行うには，メモリ空間の転送元，転送先の双方が，ピンダウンされていなくてはならない．さらに要求元のノードは，メモリ転送時に相手先のピンダウンアドレスを指定しなくてはならない．ページ転送を行うにあたり，逐一，要求元，相手先のページをピンダウンし，ピンダウンアドレスをメッセージ通信によって交換していては，効率的なページ転送を期待できない．我々の評価環境として採用した PCC2（4.5 節に後述）では，1 ページのピンダウン，ピンダウンアドレスの交換にはそれぞれ 47 μsec，18 μsec を要する．

SCASH では，ページのピンダウン，ピンダウンアドレスの交換に要するコストを削除するため，共有メモリの初期化時にすべての共有メモリ領域をピンダウンし，ピンダウンアドレスを他の全ノードへブロードキャストすることにした．SCASH では共有メモリ領域は全ノードで共通のアドレスに配置されているので，ページの送信ノードは，ページのアドレスと初期化時にブロードキャストされたピンダウンアドレスから，送信先のページのピンダウンアドレスを求めることができる．そのため，SCASH ではページ転送のために，逐一ページのピンダウンと，ピンダウンアドレスの交換をせずに，即座にゼロコピー通信によるページ転送が可能である．

ゼロコピー通信を行う場合，一般にネットワークインタフェースカード（以下，NIC と記述する）上にピンダウン領域のページテーブルを展開するため，ピンダウン領域の大きさは NIC 上のメモリの大きさに制

限されてしまうことが多い．しかし我々が通信ライブラリとして採用したPMでは，ピンダウン領域のページテーブルを，NIC上ではなくホストメモリ上に展開し，ホストメモリからNICへページテーブルの内容をDMA転送することにした．そのためピンダウン領域の大きさはNICのメモリの大きさに制限されることはない．

4. 評価方法

4.1 ページ転送速度

メッセージ通信，遠隔メモリ書き込み，遠隔メモリ読み込みの3つの通信方法による1ページの転送コスト，すなわち4Kバイトの転送に要するコストを測定する．メッセージ通信によるページ転送コストは，ページから送信バッファへのコピー，受信バッファからページへのコピーに要するコスト含むものとする．遠隔メモリ書き込み，遠隔メモリ読み込みによるページ転送コストは，1ページの転送に要するコストとする．

4.2 メモリバリアの基本評価

実行ノード数ごとのメモリバリアの最少コストを2種類の一貫性維持プロトコルについて測定する．共有メモリ領域にアクセスを行わずに，単純にメモリバリアを複数回ループ実行した場合のメモリバリアの実行時間を測定し，メモリバリアの最少コストとする．すなわち，差分の作成，送信，およびページの更新をまったく行わないメモリバリアの実行時間をメモリバリアの最少コストとして測定する．測定はノード数2台から64台まで，メモリバリアを1万回ループさせて行う．

4.3 ページフォールトハンドラの基本評価

ページフォールトハンドラの実行において顕在化する最少限のコストとページ共有のためのページ転送のコストを以下のとおり測定する．

4.3.1 ページフォールトハンドラのコスト

ページフォールトハンドラの実行において顕在化する最少限のコストを測定する．ページフォールトハンドラのコストとして，ページに対して，それぞれ読み込み例外，書き込み例外が起きた場合の，ページフォールトハンドラの実行時間を測定する．

測定は，初期化直後のページに対して1ワードの読み込み，書き込みを行うことによって行う．共有記憶領域に対する読み込み，書き込みによってページフォールトハンドラが起動されてから，読み込み，書き込みに応じた処理を終了するまでの時間を測定する．メッセージ通信によるページ転送と遠隔メモリ読み込みを用いたページ転送の双方について測定する．

4.3.2 ページ共有のコスト

ページフォールトハンドラの実行中に，新たに共有するページを管理ノードから受信するまでのコストを測定する．従来のメッセージ通信によるページ転送と遠隔メモリ読み込みによるゼロコピー通信を用いた2通りについて測定する．メッセージ通信によるページ共有のコストは，管理ノードへページ共有を要求するメッセージを送信し，管理ノードからページを受信し，バッファからコピーするまでの時間を測定する．一方，遠隔メモリ読み込みによるページ共有は，ページ共有を要求するメッセージを送信し，遠隔メモリ読み込みによりページを管理ノードから転送するまでの時間を測定する．この測定により，メッセージ通信によるページ共有と遠隔メモリ読み込みによるページ共有のコストを比較する．

4.4 アプリケーションによる評価

2種類の一貫性維持プロトコルに対して，ページ転送方式が実際のアプリケーションの性能に与える影響を明らかにことがアプリケーションによる評価の目的である．

そのため評価に用いるアプリケーションは，バリア同期のみで記述されているものが適していると考えられる．またアプリケーションの種類としては，共有メモリに対する局所性が高く，一貫性維持プロトコル，ページ転送方式の双方が性能に与える影響が小さいと考えられるLAPLACEと，バリア同期のみで記述されたアプリケーションのベンチマークプログラムとして広く採用されているものとしてSPLASH2[5]からLU（CONT）の2つを採用する．

上記の2つのアプリケーションを用いて実行時間，ページフォールトハンドラ，メモリバリアのコスト，およびページの更新，共有に要するコストと実行回数を測定する．LAPLACE，LUともゼロコピー通信，メッセージ通信の2種類のページ転送方式を2種類の一貫性維持プロトコルに適用して測定を行う．実行ノード数は1台から64台までとする．ノードが1台の実行時間は，LUはSPLASH2のnullmacroを用いた実行時間，LAPLACEはシングルスレッドでの実行時間を測定する．LAPLACE，LUともにノード数1台の実行では，SCASHのライブラリは呼び出されないので，SCASHのオーバヘッドは存在しない．測定に用いる問題サイズは**表1**のとおりである．

測定はすべて10回行い，平均値を採用する．LAPLACE，LUともに共有メモリ領域のページの管理ノードについては，ページに対して最もアクセス頻度の高いノードを管理ノードに割り当てている．

表 1 問題サイズ
Table 1 Problem size.

LAPLACE	1022 × 1022 Matrix, Iteration 50
LU (CONT)	2048 × 2048 Matrix, 32 × 32 Block

表 2 PC クラスタの主な仕様
Table 2 PC cluster specification.

# of Node	128
CPU	Intel PentiumPro 200 Mhz
Cache	512 KBytes
Chipset	440FX
Memory	EDO 256 MBytes/Node
Network	Myrinet 1.28 GBits/sec
Node OS	Linux Ver 2.2.9

4.5 評価環境

測定はすべて我々が開発した PC クラスタ 2 号機上で行う．PC クラスタ 2 号機の主な仕様を**表 2** に示す．Myrinet 用通信ライブラリ PM の最小遅延時間と最大帯域幅はそれぞれ，7.2 μsec，117.6 MBytes/sec である．

5. 評価結果

5.1 ページ転送速度

メッセージ通信，遠隔メモリ書き込み，遠隔メモリ読み込みの 3 つの通信方法によるページの転送コストと，通信コストから得られたバンド幅を**表 3** に示す．メッセージ通信との比較で，遠隔メモリ書き込み，遠隔メモリ読み込みによるページ転送速度はそれぞれ，2.46 倍，1.24 倍高速である．

5.2 メモリバリアの基本評価

メモリバリアの最小コストの測定結果を**図 5** に示す．2 つの一貫性維持プロトコルについてメモリバリアの実行時間を比較すると，すべての実行ノード数において更新プロトコルと無効化プロトコルとの実行時間の比がほぼ 3 対 2 になっているのが分かる．これは，更新プロトコルと無効化プロトコルそれぞれのバリア同期の実行回数に等しい．

ページ一貫性維持プロトコルに更新プロトコルを採用した場合，メモリバリア中にページ更新が発生するが，これは単純にページをメッセージ通信または遠隔メモリ書き込みによってページを転送するので，ページ更新時間は，ページ転送時間に等しい．

5.3 ページフォールトハンドラの基本評価

ページ転送方式ごとの，ページフォールトハンドラの実行コストを**表 4** に示す．読み込み違反によるページフォールトハンドラの実行コストを比較すると，遠隔メモリ読み込みによるページ転送を用いた場合，メッセージ通信によるページ転送と比較して 1.88 倍高速であることが分かる．

表 3 ページ転送コスト
Table 3 Page transfer cost.

転送方法	コスト (μsec)	バンド幅 (MBytes/sec)
メッセージ通信	122	32.0
遠隔メモリ書き込み	49.6	78.8
遠隔メモリ読み込み	98.4	39.7

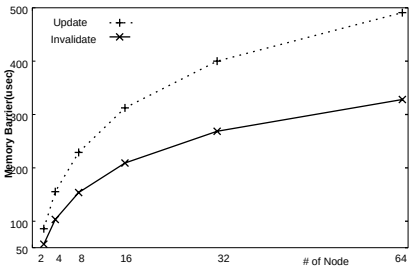

図 5 メモリバリアの最少コスト
Fig. 5 Minimum cost of memory barrier.

表 4 ページフォールトハンドラ
Table 4 Page fault handler cost.

転送方法	Read Fault (μsec)	Write Fault (μsec)
メッセージ通信	235	82
遠隔メモリ読み込み	125	82

表 5 ページ共有コスト
Table 5 Page share cost.

転送方法	コスト（μsec）
メッセージ通信	219
遠隔メモリ読み込み	112

次に読み込みによるページフォルトハンドラのコストに占めるページ共有のコストを，ページ転送方式別に**表 5** に示す．遠隔メモリ読み込みによるページ共有が，メッセージ通信と比較して，1.96 倍高速であり，読み込みによるページフォールトハンドラの実行時間の差は，ページ転送コストによるページ共有のコストの差であることが分かる．

5.4 アプリケーションによる評価

LAPLACE，LU を 1 台のノードで逐次実行した実行結果を**表 6** に示す．次に，LAPLACE，LU を並列実行した実行時間と逐次実行した実行時間を比較して得られた台数効果をそれぞれ**図 6**，**図 7** に示す．

LAPLACE，LU を並列実行した場合のメモリバリアの実行時間と，メモリバリア中のページ更新に要した時間をそれぞれ**図 8**，**図 9** に示す．4 本ずつ並んでいる縦棒は，左から順に更新プロトコル・メッセージ

表 6 LAPLACE，LU の逐次実行時間
Table 6 Sequential execution time of LAPLACE and LU.

Application	Time (sec)
LAPLACE	13.57
LU	131.8

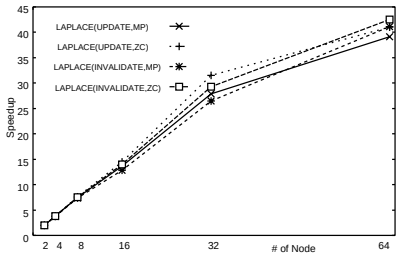

図 6 LAPLACE の台数効果
Fig. 6 Speedup of LAPLACE.

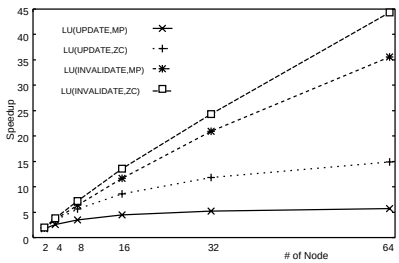

図 7 LU の台数効果
Fig. 7 Speedup of LU.

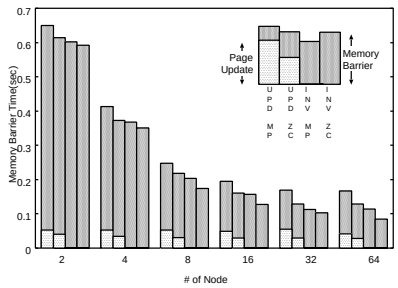

図 8 LAPLACE のメモリバリア実行時間
Fig. 8 Memory barrier execution time of LAPLACE.

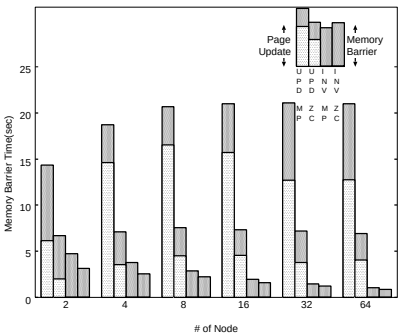

図 9 LU のメモリバリア実行時間
Fig. 9 Memory barrier execution time of LU.

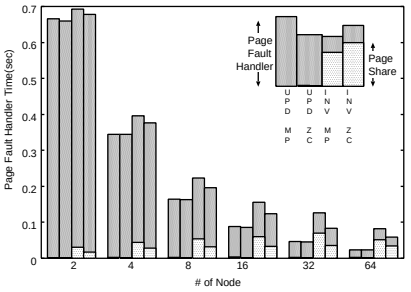

図 10 LAPLACE のページフォールトハンドラ実行時間
Fig. 10 Page fault handler execution time of LAPLACE.

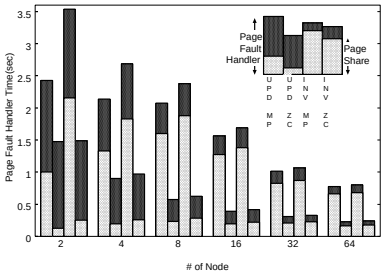

図 11 LU のページフォールトハンドラ実行時間
Fig. 11 Page fault handler execution time of LU.

通信，更新プロトコル・ゼロコピー通信，無効化プロトコル・メッセージ通信，無効化プロトコル・ゼロコピー通信の組合せである．棒全体の高さがメモリバリアの実行時間であり，そのうちのページ更新の実行時間を下の棒で示す．

LAPLACE，LU を並列実行した場合のページフォールトハンドラの実行時間と，ページ共有に要する割合をそれぞれ**図 10**，**図 11** に示す．4 本ずつ並んでいる縦棒は，バリア同様に左から順に更新プロトコル・メッセージ通信，更新プロトコル・ゼロコピー通信，無効化プロトコル・メッセージ通信，無効化プロトコル・ゼロコピー通信の組合せである．棒全体の高さがページフォールトハンドラの実行時間であり，そのうちのページ共有の実行時間を下の棒で示す．

最後に LAPLACE，LU を並列実行した場合のページ更新，ページ共有の実行回数をそれぞれ**図 12**，**図 13** に示す．

6. 検　　討

6.1 LAPLACE

台数効果としては，64 ノードの実行でゼロコピー通信を採用した場合，更新プロトコル，無効化プロトコルでそれぞれ，41.1 倍，42.5 倍のスピードアップを得ている．

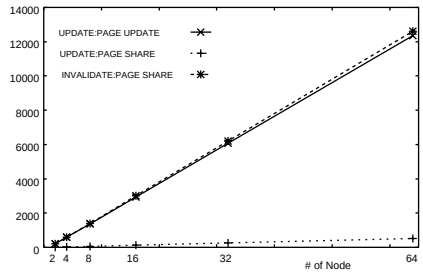

図 12 LAPLACE のページ更新，共有回数
Fig. 12 Number of page update and share of LAPLACE.

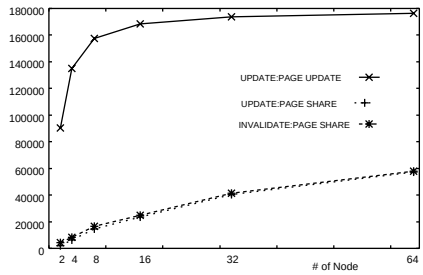

図 13 LU のページ更新，共有回数
Fig. 13 Number of page update and share of LU.

ただし，図 6 から分かるように，LAPLACE では，更新プロトコル，無効化プロトコルの双方において，ページ転送方式による性能差はほとんどなく，ゼロコピー通信を用いた方が若干優れている程度である．メモリバリア，ページフォールトハンドラの実行時間を比較しても，ゼロコピー通信を用いた方が実行時間が若干小さい程度で大きな差は見られない．

LAPLACE は図 12 から分かるように更新プロトコルを用いた場合のページ共有の実行回数が少なく，ページの局所性が高いアプリケーションである．さらに，更新プロトコルのページ更新回数と，無効化プロトコルのページ共有回数がほぼ等しいことから，無駄なページ更新がほとんど行われていないことが分かる．

以上から LAPLACE のようにページの局所性が高く，ページ転送回数が少ないアプリケーションでは，ページ転送方式の違いによる性能差はほとんど見られないことが分かる．

6.2 LU

ページ転送方式別に，LU の実行時間を比較すると，64 ノードで実行した場合，ゼロコピー通信によるページ転送の方が，メッセージ通信によるページ転送を行った場合との比較で，更新プロトコル，無効化プロトコルでそれぞれ 2.61 倍，1.25 倍高速である．台数効果としては，64 ノードの実行でゼロコピー通信を採用した場合，更新プロトコル，無効化プロトコルでそれぞれ，14.9 倍，44.3 倍のスピードアップを得ている．

更新プロトコル，無効化プロトコルの双方において，メモリバリアの実行時間は測定した全実行ノードにおいて，ゼロコピー通信の方が実行時間が小さいことが分かる．更新プロトコルを採用した場合，ページ更新の実行時間の差以上にメモリバリアの実行時間の差が大きい．無効化プロトコルを採用した場合でもゼロコピー通信を用いた方が，バリアの実行時間は小さいことが分かる．

ページフォールトハンドラの実行時間を比較すると，すべての実行ノード数で，無効化プロトコルを採用した方が，更新プロトコルを採用した場合よりも大きい．無効化プロトコルと，更新プロトコルのページフォールトハンドラの実行時間の差は，ページ共有の実行時間に大きく依存していることが分かる．更新プロトコル，無効化プロトコルともに，すべての実行ノード数において，ゼロコピー通信を用いた方が高速であり，更新プロトコル，無効化プロトコルにおけるページフォールトハンドラの実行時間の差は，ページ共有の実行時間におおきく依存しているといえる．

無効化プロトコルの 64 ノード実行についてページ転送プロトコルごとにページ共有の実行時間を比較すると，ゼロコピー通信は，メッセージ通信に対して，3.87 倍も高速である．これは基本評価で求めたページ共有のスピードアップ 1.96 倍よりも大きい．遠隔メモリ読み込みを用いたページ共有の場合，相手先ノードのメッセージ受信を待たずに，ページ読み出しが可能であるため，実際のアプリケーションでは，ページ転送速度以上の性能差が生じたためであると考えられる．

ページの更新・転送回数については，更新プロトコルを用いた場合のページ転送数，すなわちページ更新とページ共有の実行回数の和が，全実行ノード数において無効化プロトコルより多い．ページ転送回数の面から，LU にはページ転送回数が小さい無効化プロトコルが適していると考えられる．

以上から，LU のようにページ転送の多いアプリケーションでは，ゼロコピー通信を用いたページ転送によって，大きく性能を改善できることが分かる．

7. 関 連 研 究

ソフトウェアによる DSM システムの性能評価はこれまでにも多数行われている．リモートメモリアクセスによるソフトウェア DSM の実現としては，論文 9) がある．この論文では MBCF [10] と呼ばれるメモリベース通信機能を用いたソフトウェア分散共有メモリ

について論じている．

Myrinetを用いた分散共有メモリの性能評価としては，論文11)がある．この論文では，複数のメモリ一貫性モデルと，複数のページサイズの粒度についてSPLASH2を用いた性能評価を行っているが，16ノードまでの性能評価にとどまっている．また，ページへのアクセスチェックには専用のハードウェア[12]を用いており，SCASHのようにソフトウェアですべてを実現しているわけではない．

マルチプルライタプロトコルを用いたソフトウェア分散共有メモリの性能に関しては，Keleher[13]が，シングルライタとSCによるDSMの実装とマルチプルライタとLRCによる性能の比較評価を行っている．この論文は一貫性モデル，ページ一貫性プロトコルとしてそれぞれ，LRCと無効化プロトコルを採用しており，更新プロトコルを用いた比較は行われていない．

上記のすべての論文は，ノード数が，8台または16台までの性能評価しか行われていないが，本論文では，64ノードまでの性能評価を行っている．

8. まとめ

本論文では，分散共有メモリシステムで採用されている2種類の一貫性維持プロトコルである更新プロトコル，無効化プロトコルの実装において，メッセージ通信，ゼロコピー通信を用いたページ転送方式を採用することによる性能の違いについて検証を行った．

Pentium PRO 200 MHz，Myrinetネットワーク上のPM通信ライブラリによるゼロコピー通信による1ページ（4 Kbytes）の転送時間は，メッセージ通信の2.46倍高速であった．更新プロトコルにおけるページ更新時間は，ページ転送時間に等しい．一方，無効化プロトコルにおいて，ページフォールトハンドラによりページが共有されるときに転送されるページをゼロコピー通信で実現した場合には，メッセージ通信で実現したときに比べて1.96倍高速になった．

行列のラプラシアンを求めるベンチマークを64ノードで実行した結果では，更新プロトコル，無効化プロトコルともにゼロコピー通信による性能向上は1.05，1.04倍にとどまり，効果が認められなかった．これはページの局所性が高くページ転送の割合が少ないためである．

SPLASH2のLUを64ノードで実行した結果では，更新プロトコル，無効化プロトコルともにゼロコピー通信による性能向上は2.61，1.24倍となった．更新プロトコルにおけるゼロコピー通信の効果が大きいことが分かる．

本論文の貢献は，同一プラットフォームにより，2種類の一貫性維持プロトコルに対して，ゼロコピー通信の効果をメッセージ通信と比較している点であり，このような論文は今まで公表されていない．

参考文献

1) http://www.myri.com.
2) Tezuka, H., Hori, A., Ishikawa, Y. and Sato, M.: PM: An Operating System Coordinated High Performance Communication Library, *High-Performance Computing and Networking '97* (1997).
3) 原田 浩，石川 裕，堀 敦史，手塚宏史，住元真司，高橋俊行：ソフトウェア分散共有メモリSCASHにおけるページ管理ノードの動的再配置機構の実装と評価，情報処理学会研究報告 99-HPC-77（SWoPP '99），pp.89–94 (1999).
4) 原田，手塚，堀，住元，高橋，石川：Myrinetを用いた分散共有メモリシステムの評価，ハイパフォーマンスコンピューティング研究会資料，98-HPC-73, pp.73–78, 情報処理学会 (1998).
5) Woo, S.C., Ohara, M., Torrie, E., Singh, J.P. and Gupta, A.: The SPLASH-2 Programs: Characterization and Methodological Considerations, *Proc. 22nd International Symposium on Computer Architecture*, pp.24–36, ACM (1995).
6) Gharachorloo, K., Lenoski, D., Laudon, J., Gibbons, P., Gupta A. and Hennessy, J.: Memory Consistency and Event Ordering in Scalable Shared-Memory Multiprocessors, *Proc. 17th Annual Symposium on Computer Archtecture*, pp.15–26 (1990).
7) Carter, J.B, Bennett, J.K. and Zwaenepoel, W.: Implementation and Performance of Munin, *Proc. 13th Symposium on Operating Systems Principles*, pp.152–164 (1991).
8) Amza, C., Cox, A.L., Dwarkadas, S. and Zwaenepoel, W.: Software DSM Protocols that Adapt between Single Writer and Multiple Writer, *Proc. 3rd IEEE Symp. on High-Performance Computer Architecture* (*HPCA-3*), pp.261–271 (1997).
9) Matsumoto, T., Komaarashi, T., Uzuhara, S. and Hiraki, K.: The Asymmetric Distributed Shared Memory Using Memory-Based Communication Facilities (in Japanese), 情報処理学会コンピュータシステムシンポジウム，pp.37–44 (1996).
10) Matsumoto, T. and Hiraki, K.: MBCF: A Protected and virtualized High-Speed User-Level Memory-Based Communication Facility, *1998 International Conference on Supercomputing*,

Melbourne, pp.259–266, ACM (1998).
11) Zhou, Y., Iftode, L., Singh, J.P., Li, K., Toonen, B.R., Schoinas, I., Hill, M.D. and Wood, D.A.: Relaxed Consistency and Coherence Granularity in DSM Systems: A Performance Evaluation, *Proc. 6th ACM Symposium on Principles and Practice of Parallel Programming* (1997).
12) Pfile, R.W.: Typhoon-Zero Implementation: The Vortex Module Technical Report, Technical report, Wisconsin University, CS department (1995).
13) Keleher, P.J.: The Relative Importance of Concurrent Writers and Weak Concsistency Models, *Proc. IEEE COMPCON '96 Conference* (1996).

(平成 11 年 9 月 9 日受付)
(平成 12 年 2 月 4 日採録)

原田　浩（正会員）
1988 年東京理科大学理学部物理学科卒業．同年（株）ソフトウェア・リサーチ・アソシエイツ入社．1997 年より技術研究組合新情報処理開発機構研究員．現在に至る．オペレーティングシステム，並列・分散システム等に興味を持つ．ACM 会員．

手塚　宏史（正会員）
1980 年東京大学教養課程中退．1981 年（株）生活構造研究所入社．1985 年ソニー（株）入社．1988 年（株）ソニーコンピュータサイエンス研究所入社．1990 年ソニー（株）入社．1993 年北陸先端科学技術大学院大学研究生．1995 年より技術研究組合新情報処理開発機構研究員．現在に至る．オペレーティングシステム，リアルタイム処理，マルチメディア処理等に興味を持つ．日本ソフトウェア科学会会員．

堀　敦史（正会員）
1979 年早稲田大学電気工学科卒業．1981 年同大学大学院理工学研究科計測制御工学専攻修士課程修了．同年（株）三菱総合研究所入社．1992 年より技術研究組合新情報処理開発機構に出向．JSPP '98 最優秀論文賞授賞．並列オペレーティングシステムの研究に従事．並列プログラミング言語，並列アーキテクチャ等に興味を持つ．工学博士（東京大学工学部）．

住元　真司（正会員）
1986 年同志社大学工学部電子工学科卒業．同年（株）富士通入社．（株）富士通研究所にて並列オペレーティングシステム，並列分散システムソフトウェアの研究開発に従事．1997 年より新情報処理開発機構に出向．コモディティネットワークを用いた高速通信機構の研究開発に従事．並列分散システムのアーキテクチャ，システムソフトウェア等に興味を持つ．

高橋　俊行（正会員）
1993 年東京理科大学理工学部情報科学科卒業．1995 年同大学大学院修士課程修了，1995〜1998 年東京大学理学系研究科情報科学科博士課程，1998 年より新情報処理開発機構研究員．現在に至る．プログラミング言語におけるメタレベルアーキテクチャと並列計算ソフトウェア技術に興味を持つ．理学修士．

石川　裕（正会員）
1987 年慶應義塾大学大学院電気工学科博士課程修了．同年電子技術総合研究所入所．1988〜1989 年カーネギー・メロン大学客員研究員．1990 年日本ソフトウエア科学会高橋奨励賞を授賞．1993 年から新情報処理開発機構に出向．並列・分散システム，適応可能並列プログラミング言語/環境/処理系，リアルタイム処理等に興味を持つ．日本ソフトウエア科学会，ACM，IEEE 各会員．工学博士．