様式 C-19

科学研究費補助金研究成果報告書

平成２２年４月１日現在

研究種目：基盤研究（B）
研究期間：2006～2008
課題番号：18300006
研究課題名（和文）　次世代 PC クラスタを活用する超大規模仮想メモリ空間支援システムの研究

研究課題名（英文）　Large Virtual Memory Space Supporting System on Next Generation PC Clusters

研究代表者
石川　裕（ISHIKAWA　YUTAKA）
東京大学・大学院情報理工学系研究科・教授
研究者番号：70345122

研究成果の概要：
64bit アドレス空間を有する PC において、高性能ネットワークでつながれた PC のメモリ領域をスワップ領域として利用可能とする遠隔スワップシステム Teramem を設計し Linux に実装、評価した。従来の遠隔メモリスワップシステムで問題となっていた効率性、可搬性を解決し、数十ギガ程度の物理メモリしか搭載していない PC においても、他の PC のメモリ領域を利用することにより、テラバイト以上の仮想メモリ領域を効率よく利用できるようになった。本システムはオープンソースとして公開されている。

交付額

（金額単位：円）

	直接経費	間接経費	合　計
2006 年度	5,400,000	1,620,000	7,020,000
2007 年度	4,200,000	1,260,000	5,460,000
2008 年度	5,100,000	1,530,000	6,630,000
年度			
年度			
総　計	14,700,000	4,410,000	19,110,000

研究分野：総合領域
科研費の分科・細目：情報学、計算機システム・ネットワーク
キーワード：リモートスワップ、PC クラスタ、分散ページ、大規模メモリ

１．研究開始当初の背景

科学技術計算、大規模データベース検索、デジタルシネマなどのアプリケーションプログラムは、超大規模メモリ空間（64bit アドレス）を必要としている。創薬支援のためのタンパク質―薬剤ドッキング計算や分子軌道法計算、ビジネスデータや医療データに関わるデータマイニング処理、科学技術計算分野における計算精度をあげるために、使える限り可能なメモリ量を使用する。
また、次世代デジタル映像技術として 2005 年 7 月に規格化された 4K デジタルシネマ(885 万画素)では、デジタル映像データをフレーム単位で編集しようとすると、1 秒の映像を編集するために数ギガバイト以上のメモリを必要とする。
64 ビットアーキテクチャが PC で使用され、原理的には仮想メモリ空間は 1 テラバイト以上のメモリを使用できるようになった。しかし、1 テラバイト以上のメモリ空間を現実的時間内でアクセスできるシステムは共有メモリ型並列コンピュータなど大量の物理メモリが搭載されている計算機に限られている。大規模メモリを搭載している計算機は並列コンピュータであるという理由から、大規模メモリを使用するユーザは並列コンピュータを利用している。共有メモリ型並列コンピュータの場合、アプリケーションプログラムを変更しなくても大容量メモリが使用できる。最近利用が加速している PC クラス

タでは、クラスタ全体でテラバイトメモリを利用できる場合もあるが、そのためには MPI 通信ライブラリなどの通信機能を用いてプログラムを書き直さないといけない。
現在一般に入手できる PC を使用してテラバイトメモリ空間(64bit アドレス)が利用できれば、大規模メモリを必要とするアプリケーションが安価なコンピュータ環境で実行でき、冒頭にあげた応用分野に甚大なる貢献ができる。物理メモリ数ギガバイトに対して 1 テラバイトのメモリ空間をアクセスするには、2 次記憶（ディスク）のスワップ領域を使い、ページ単位でのメモリ退避復帰（スワッピング）を行なう必要がある。ディスクヘッド移動を伴うディスクアクセスはミリ秒単位かかる。メモリワーキングセットが物理メモリに収まらないとページ置換アルゴリズムに従ってスワッピングが頻繁に行なわれ、処理性能が著しく低下する。このため、物理メモリ数ギガバイトしかない PC では、1 テラバイトのメモリ空間をアクセスするプログラムは現実的時間内に終了しない。

２．研究の目的
本研究では、10G Ethernet、Infiniband、Myrinet-10 などの高速ネットワークでつながった複数の PC から構成される PC クラスタにおいて、1 テラバイト超仮想アドレス空間を提供する次世代並列分散システムソフトウェアの研究開発を行なう。

３．研究の方法
従来の OS は、物理メモリを超える仮想メモリ空間を実現するために、メモリの内容を一時的に格納(スワップ)する領域として磁気ディスクを想定したスワップシステムが存在する。スワップ先のデバイスとしてネットワーク上のリモートの計算機上のメモリを利用できるようにネットワークデバイスを開発する方法が従来から存在する。しかし、この手法は、2 つの点で性能の問題が生じる。
● 一つ目は、スワップ先のデバイスとして磁気ディスクを想定していることに起因する問題である。磁気ディスクは、逐次アクセスは性能が出るが、ランダムなアクセスはヘッドのシーク時間がかかることによる性能劣化がある。これを解決するために、メモリの内容をディスクに吐き出す（スワップアウトする）ときにスワップ領域の空き領域に順次格納されるように最適化している。このために、既にディスク上に格納されているメモリ領域が存在し、その領域が変更されていなくても、主記憶からディスクへの書き出しが行なわれる。本最適化は、ディスクにおいては正しいが、ネットワーク上の計算機のメモリを利用するときには、無駄なデータ転送が生じてしまう。
● 2 番目の問題点は、スワップの単位がページ単位(4Kbyte)である点である。図 1 は、Myricom 社 Myrinet-10G と呼ばれる高速ネットワークハードウェアを使ったときのネットワーク性能を計測した結果である。計測には、Myricom 社が提供している mx_pingpong プログラムを使用している。

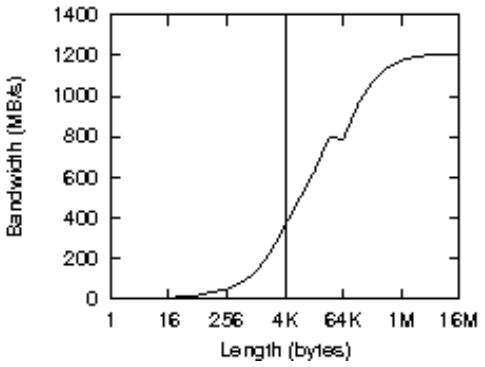

図 1　転送サイズと通信バンド幅

図 1 に示す通り、4KB(ページ単位)のデータ転送では、最大ネットワーク性能の 1/3 程度しか利用できないことが分かる。
スワップデバイスとしてリモートメモリを使用する手法と異なり、OS が提供する mmap システムコールを利用して、ユーザレベルでスワップを管理する手法も提案されている。しかし、この手法では、メモリアクセスの履歴に基づくスワップ管理が出来ない。
我々は、これら既存研究の問題を解決するために、OS が提供しているスワップ機構とは別に、リモートメモリスワップを実現する機能を Linux カーネルモジュールとして実現した。カーネルレベルで実現することにより、上記問題点が解決できた。また、様々なネットワークデバイスに対応できるようにした。

４．研究成果
(1) 64bit アドレス空間を有する PC において、高性能ネットワークでつながれた PC のメモリ領域をスワップ領域として利用可能とする遠隔スワップシステム Teramem を設計し Linux に実装、評価した。従来の遠隔メモリスワップシステムで問題となっていた効率性、可搬性を解決するために、Teramem は、以下の特徴を有する高性能高可搬なシステムとして実現した。
● OS カーネルで実現することによりカーネルでしかアクセスできないページテーブルのメモリアクセス情報を用いて、LIL0、FIFO などのメモリ置換アルゴリズムを効率

的に実現した。

● ページサイズ(Linuxは4KB)程度のデータ長で通信するとネットワーク性能を引き出すことができないため、ページをまとめて通信することにより、ネットワーク転送性能を向上させた。

● Linuxカーネルを修正せず、カーネルロードモジュールとしてTeramemを実装した。これにより、Linux利用者は、Linuxカーネルを入れ替えることなく、Teramemを利用したいときに、Teramemモジュールを追加するだけで、利用できるようにした。

● 特別なネットワークハードウェアではなく、普及しているEthernetや高性能PCクラスタで使用されているInfinibandやMyrinetなどの複数ネットワークに対応できるようにした。このために、分散並列ファイルシステムであるLustreファイルシステムが提供している通信レイヤLNetを使用した。Teramemの全体像を図 2に示す。

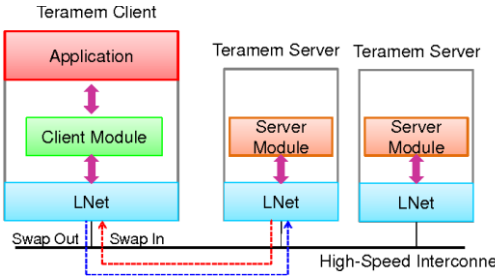

図 2 Teramemの構成

図 2において、Teramem Clientが大規模メモリ領域を使用するマシンであり、メモリスワップ領域としてTeramem Serverのメモリ領域を利用する。

(2) Teramemの有効性を示すために以下の実験を行った。実験に使用した計算機環境を表 1に示す。

表 1 実験環境

CPU	Dual Core AMD Opteron 2214 (2.2GHz) x 2
Memory	4 GB DDR2-667
Disk	80 GB SATA
Network	Myrinet 10G
OS	CentOS 5.2 (Kernel 2.6.18)
#Nodes	32

利用できる物理メモリを1GBとして、64GBの仮想メモリ領域を確保し、全ての仮想メモリ領域がリモートメモリ上に存在した時に、仮想メモリ領域を逐次アクセスしたときの性能を計測した。具体的には、Teramemが管理するメモリ領域として64GBを確保してデータを書き込み、一旦すべてのデータをスワップアウトした後、全体を最初から最後まで順に読んだときのバンド幅を計測した。さらに、スワップ領域をディスクとした場合と比較した。結果を図 3に示す。

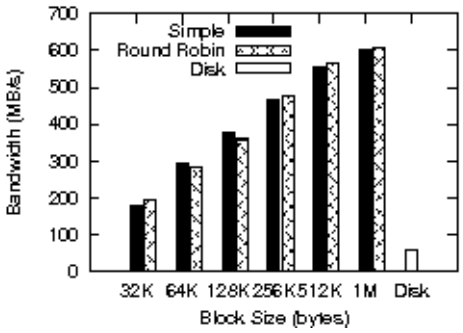

図 3 連続メモリ読み込み性能

図 3において、横軸は一回のメモリ転送サイズ(ブロック転送サイズ)を示している。横軸の右端のDiskはローカルディスクからデータを読み込んだ時の性能を示している。また、Teramem Serverの使用可能なリモートメモリを最初から順番にスワップアウト先を割り当てた場合(Simple)と、各Teramem Serverにラウンドロビン方式で割り当てた場合(Round Robin)の両方の結果を示す。割り当て方式Simpleの場合、ブロックサイズ1MBで最大の603MB/sとなった。これはローカルディスクの連続読み出しバンド幅58.9MB/sの約10.2倍である。Round Robinを用いた場合のバンド幅はSimpleの95%から107%で、スワップアウト先の割り当て方法による有意な差は見られなかった。これ以降のベンチマークでは、Simpleを採用した結果のみを示す。

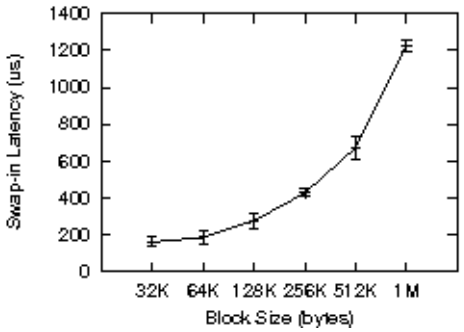

図 4 メモリスワップインにかかる遅延

(3) 前述の連続アクセスベンチマークにおいて、1回のスワップインに要した時間の平均値を図 4に示す。これはユーザプログラムにおける待ち時間を示している。ブロックサイズ1MBにおいて、1回のスワップイン

に約 1.2ms を要した。なお、どのブロックサイズにおいても所要時間の分散は少なく、標準偏差は数十μs 程度であった。

(4)　　ブロックサイズが大きいほど連続読み出しアクセスのバンド幅が向上することを示した。しかし、メモリアクセスの空間的局所性が小さいランダムアクセスの場合、スワップインしたブロックのうちごく一部だけしか使用されないことが起こり、スワップインの遅延時間のコストが性能に影響することになる。最悪のケースは、ストライドアクセスのように局所性が全くないアクセスパターンである。そこで、ブロックサイズ 32KB, 128KB, 1MB のそれぞれでメモリをストライドアクセスしたときのスワップインにかかる時間を測定した。図 5 は、ブロックサイズ 1MB、ストライド 4KB のときを基準に正規化したものである。

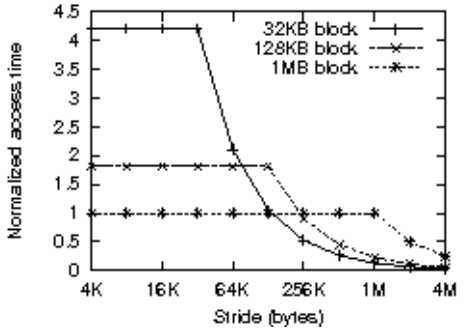

図 5　ストライドアクセス時のスワップイン性能

ストライドがブロックサイズ以下の場合には、ストライドに関係なくすべてのブロックがスワップインされるため、グラフは直線になる。ストライドがブロックサイズを超えるとスワップインすべきブロック数が減るので、ストライドに反比例して所要時間が減少する。ストライドサイズが 4KB の時、ブロックサイズ 32KB、128KB のときの所要時間はブロックサイズ 1MB と比較してそれぞれ約 4.2 倍と約 1.8 倍であった。ストライド 128KB 以下ではブロックサイズ 1MB の所要時間が最短だが、ストライド 256KB 以上ではブロックサイズ 32KB が最短になっている。また、ブロックサイズ 32KB と 128KB を比較してみても、ストライド 64KB 以下ではブロックサイズ 128KB が有利で、ストライド 128KB 以上ではブロックサイズ 32KB が有利になっている。このように、最適なブロックサイズはメモリアクセスパターンによって異なり、アクセスの空間的局所性に大きく依存していることが分かる。

(5)　　実用的アプリケーションによるベンチマークとして、GNU sort を用いてベンチマークを行った。このベンチマークでは、GNU coreutils 6.12 に含まれる GNU sort のプログラムに若干の変更を加え、大きなバッファを確保する部分で malloc/free の代わりに Teramem が提供する teramalloc/terafree を使うようにした。GNU sort が使用できる物理メモリのサイズを変化させながら、約 600MB のファイルをソートしたときの実行時間を図 6 に示す。

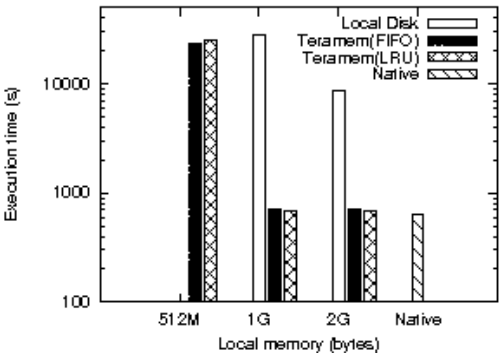

図 6　GNU Sort の実行時間 (1)

本実験では、GNU sort のメモリ使用量は約 3.3GB であった。物理メモリに収まらないデータはすべてリモートメモリまたはディスクにスワップアウトされたことになる。ここで、Native とは、GNU sort が使用するメモリは全て物理メモリに存在し、スワップされない時の性能である。ローカルメモリ 2GB, 1GB の時、Teramem を使った場合の実行時間は Native と比べ最大 10%しか増加しなかった。これに対し、同じローカルメモリサイズでローカルスワップを使った場合の実行時間は大幅に増加し、ローカルメモリ 2GB のときで Teramem(LRU)の約 12.5 倍、1GB のときで 40 倍以上だった。ブロック置換に LRU を用いたときの実行時間は、FIFO 置換よりもそれぞれ 3.3%、2.8¥%短縮された。ローカルメモリを 512MB まで制限すると、Teramem を使った場合でも実行時間が急激に増大した。これは、空間的局所性の小さいアクセスパターンを示す領域がローカルメモリに収まらなくなったためと考えられる。このときのスワップイン/アウトの量の合計は FIFO が 27.27TB、LRU が 27.15TB で、LRU のほうが若干少ないものの、LRU を実現するためのオーバーヘッドなどから、実行時間は FIFO のほうが約 9%短くなった。ローカルメモリ 512MB でローカルスワップを用いたベンチマークは、48 時間以上経過しても終了せず、実行時間を測定できなかった。

(6)　　物理メモリを 1GB に設定し、ブロックサイズを変化させた時の約 1GB のファイル

をソートしたときの実行時間を図 7 に示す。GNU sort のメモリ使用量は約 5.6GB であった。

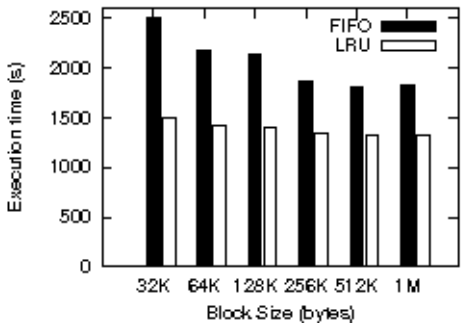

図 7　GNU Sort の実行時間 (2)

LRU を用いたときの実行時間は FIFO を用いたときの 59%から 73%で、どのブロックサイズでも FIFO より LRU のほうが短かった。このように、LRU により空間的局所性を有効利用できていることが分かる。実行時間は FIFO, LRU いずれの場合もブロックサイズ 512KB で最小となった。

(7)　　Teramem は、連続アクセスでディスクの約 10.2 倍、GNU sort を用いたベンチマークでは、40 倍以上の性能を達成した。また、カーネルレベル実装によってページテーブルの情報を利用した効率的なブロック置換ができること、ユーザプログラムから観測すると、約 1.2ms(ブロックサイズ 1MB の場合)という短い待ち時間でスワップインが行なえることなどをベンチマークプログラムで確認した。Teramem により、数十ギガ程度の物理メモリしか搭載していない PC においても、他の PC のメモリ領域を利用することにより、テラバイト以上の仮想メモリ領域を効率よく利用できる。本システムはオープンソースとして公開されている。

５．主な発表論文等
（研究代表者、研究分担者及び連携研究者には下線）
〔雑誌論文〕（計 1 件）
① 山本 和典、石川 裕、「テラスケールコンピューティングのための遠隔スワップシステム Teramem」、情報処理学会論文誌 コンピューティングシステム、ACS27、pp. 142 – 152、2009 (査読あり)

〔学会発表〕（計 4 件）
① 山本 和典、石川 裕、「テラスケールコンピューティングのための遠隔スワップシステム Teramem」、情報処理学会第 7 回先進的計算基盤システムシンポジウム SACSIS 2009 論文集、IPSJ Symposium Series Vol. 2009, No.5、pp. 255 – 264、2009 (査読あり、ACS27 と同時投稿)
② Kazunori Yamamoto and Yutaka Ishikawa, “Teramem: A Remote Swapping System for High-Performance Computing,” IEEE/ACM, SC’09, November, 2009 (査読あり、Poster).
③ 北村 裕太、松葉 浩也、石川 裕、「大規模メモリ空間の利用を支援する遠隔スワップメモリシステム」、情報処理学会研究報告、2007-HPC-111、pp. 121-126、2007(査読なし).
④ 今井照之、松葉浩也、石川裕、「分散ページングによる大規模仮想メモリ空間」、情報処理学会 研究報告、2007-HPC-109、pp. 85-90、2007(査読なし).

〔図書〕（計 0 件）
〔産業財産権〕
○出願状況（計 0 件）
○取得状況（計 0 件）

〔その他〕
以下の URL でソフトウェアを公開している。
http;//www.il.is.s.u-tokyo.ac.jp/teramem/
学会発表の①において、第 7 回先進的計算基盤システムシンポジウム優秀若手研究賞を授賞した。

６．研究組織
(1)研究代表者
石川 裕 (ISHIKAWA YUTAKA)
東京大学・大学院情報理工学系研究科・教授
研究者番号：7034 5122
(2)研究分担者
佐藤 三久 (SATO MITSUHISA)
筑波大学・システム情報工学研究科・教授
研究者番号：6033 3481
朴泰 祐 (BOKU TAISUKE)
筑波大学・システム情報工学研究科・教授
研究者番号：90209346
秋山 泰 (AKIYAMA YUTAKA)
東京工業大学・大学院情報理工学研究科・教授
研究者番号：30243091
松葉 浩也 (MATSUBA HIROYA)
東京大学・情報基盤センター・助教
研究者番号：3044 4095

(3)連携研究者
なし