システムソフトウェアと
オペレーティング・システム 61－9
（１９９３．８．１８）

超流動 OS の大域的仮想仮想記憶における ページ探索法の比較

平野 聡　一杉 裕志　田沼 均　須崎 有康

電子技術総合研究所

{hirano,ichisugi,tanuma,suzaki}@etl.go.jp

分散メモリ型超並列システムのための大域的仮想仮想記憶 (GVVM) は、PE 空間全体でのページ使用頻度に基づくページング、及び、他 PE 上の使用頻度の低いメモリをスワップ領域として用いる事により、メモリの自動負荷分散を行なう。メモリ使用頻度の少ない PE を発見する 7 種類のスワップ領域探索法の比較を行った結果、メタ情報共有メカニズムによるマップを用いて距離優先で探索する手法が有望であることが明らかになった。

Swap Space Search Methods for Global Virtual Virtual Memory

HIRANO Satoshi　ICHISUGI Yuuji　TANUMA Hitoshi　SUZAKI Kuniyasu

Electrotechnical Laboratory, Japan

{hirano,ichisugi,tanuma,suzaki}@etl.go.jp

The Global Virtual Virtual Memory (GVVM) intends load balancing of memory for massively parallel multicomputer systems by system wide LRU paging and page borrowing from other PE's. In the result of comparison among seven swap space search methods, we found that a map oriented method using "MetaShare" is the best choice among them.

1 はじめに

仮想記憶(Virtual Memory)の目的は、プログラムに実記憶よりも大きな仮想空間を提供し、実記憶の大きさを超える問題を解くことである。たとえ大量のメモリを備え広大な実記憶が使用可能な超並列システムが登場しても、より大きな問題を解きたいという利用者の願いは変わらないであろう。

並列システムにおいては、多数のPEに対して見合うだけのバンド幅を備える二次記憶を用意することは困難であるため、解くべき問題の大きさが主記憶容量を超えた場合の仮想記憶機構の性能低下は逐次マシンの場合よりも著しくなる。また、並列システムの場合、プログラムの有するメモリ使用量の偏在性と局所性が原因で、少数のPEが多量のメモリを消費し、システム全体のメモリには余裕があるにもかかわらず、それらのPEでページングが頻発し実行性能が著しく低下することがある。これらの問題のため、現在の仮想記憶機構とは異なる並列システムのための記憶管理の機構が必要である。

我々は汎用超並列システムにおけるオペレーティングシステム「超流動OS」[5, 8]の仮想記憶管理方式として大域的仮想仮想記憶(GVVM)を提案した[7]。GVVMは、多数のプログラムが混在する分散メモリアーキテクチャの超並列システム上での実行時間の短縮を目的として、PE空間全体でのページ使用頻度に基づくデマンド・ページング、及び、他PE上の使用頻度の低いメモリページをスワップ領域として用いることにより、メモリの自動的な負荷分散を行なう。

GVVMでは、あるプロセッサPE[i]でメモリが不足した際、システム中でメモリの使用頻度のより低いPE[j]を探索する。PE[j]は二次記憶にページアウトすることにより、PE[i]のためのスワップ領域となる空ページフレームを作成し、PE[i]はその空ページフレームに対し仮想的にページアウトを行なう。本論文では、比較的小規模なPE台数のシステムを対象にして、メモリ使用頻度の低いPEを探索する7つの探索法を比較検討する。大きなコストをかけてより完全な探索を行い、高いLRU (Least Recently Used)効果を得る方がよいのか、あるいは、コストをかけずに適当な探索を行なう方が全体の性能は上がるのか、といった点が課題となる。

以下、2章と3章でGVVMの基本概念と実現モデルについて述べた後、4章でスワップ領域探索の検討課題ついて述べる。そして、4章でシミュレータ上に実現したGVVM処理系を用いて7種類のスワップ領域探索法の比較を行なう。

2 大域的仮想仮想記憶の基本概念

本章ではGVVMの基本概念の概要について述べる。超

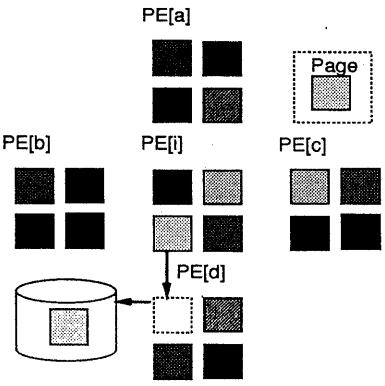

図 1: 大域的仮想仮想記憶

流動OSは複数のプログラムが混在して動作する環境であるため、メモリアクセスに関してPE内での局所性と共にPE空間内での局所性やメモリ使用量の偏りも期待できる。そこで、我々の提案するGVVMは以下の二つの基本概念を基にして高性能な仮想記憶システムの実現を図る。

大域的仮想記憶 Global Virtual Memory デマンドページングを個々のPE内でのページの使用頻度に基づいて独立して行なう方式を単一PE仮想記憶と呼ぶことにする。これに対し、大域的仮想記憶はページアウトの対象のページを各PE上のメモリ内から決定するのでなく、システム全体のPE上のメモリを対象にして決定する。例えば、置き換えアルゴリズムとしてLRUを用いるならば、全PEの全ページの使用頻度(アクセス時刻)を考慮し最も過去にアクセスしたページをページアウトする。

しかし、ページを必要とするPEと、ページアウトを行なうPEが一致することはまれであるため、前者が後者のページアウトにより空いた実メモリを使用するための仕組みが必要となる。これは次の仮想仮想記憶により実現する。

仮想仮想記憶 Virtual Virtual Memory 並列システムでは二次記憶への転送バンド幅が小さいため、全てのページングを二次記憶に対して行なうことはシステムの性能低下を引き起こす。そこで、他PE上に未使用のページがある場合、そのページをスワップ領域として用い、仮想的にページアウトを行なう[1, 4]。

大域的仮想仮想記憶(Global Virtual Virtual Memory, GVVM)は上記の二方式を融合した方式である。即ち、上記の仮想仮想記憶において、他PE上のスワップ領域として使用可能なページを選択する際、大域的仮想記憶の概念を用いる。

図1はGVVMの概念を図示したものである。四角形の箱はページを表し、色の濃さはそのページの使用頻度を表している。

PE[i]においてページフォルトが発生したとする。その際、PE[i]上で未使用の実記憶がなかった場合、最も使用頻度の低いページをページアウトし、ページフォルトが発生したページを読み込むための領域を確保しなければならない。そこで、PE[i]中で最も使用頻度の低いページ(図中では左下のページ)をPE[i]から排除するのであるが、実際に二次記憶装置へページアウトするのはシステム全体で最も使用頻度の低いページとする。例えば、図ではPE[d]の左上のページが該当する。このページを二次記憶装置へページアウトし、空いた領域をスワップ領域としてPE[i]のページを仮想的にページアウトする。

なお、本論文中では、スワップ領域を提供するPE[d]を**スワップ先PE**、PE[i]がPE[d]へページアウトすることを仮想ページアウトと呼ぶ。

アプリケーション・プログラムのプログラマの視点から見ると、メモリはGVVMによって仮想化され、実記憶以上の空間が使用可能になると共に、プログラム毎あるいはスレッド毎に異った分布を有するメモリの使用量という負荷が、OSによって自動的にシステム全体に分散される事になる。

GVVMに関連する研究として、Shared Virtual Memory (SVM)[1, 2, 3]がある。SVMは分散メモリの分散/並列システム上で仮想記憶付の共有メモリを実現する。しかし、仮想記憶管理としてはOSの仮想記憶機構をそのまま用いており、あるPEで共有メモリを保持するメモリが溢れたら、そのPEのディスクにページアウトを行う。どのページをディスクに掃き出すかを決定する優先度の算出には、共有メモリを実現する際に用いるread onlyやwrite等のページ属性を利用する。他PEのメモリをスワップ領域として用いることについても若干触れているが、実現や評価は為されていない。分散メモリシステム上で他PEのメモリをスワップ領域として利用する研究であるDistributed Virtual Memory (DVM)[4]は共有メモリ機構を有しない点でGVVMと多くの類似点がある。更に、DVMはPE間でのページの転送を利用したプロセスの移送を実現している。しかし、DVMではページのスワップは直接の隣接PEとしか行なわず、GVVMが主眼とする大域的な負荷分散を目的とはしていない。5章では比較のためDVMも合わせて評価する。

3 大域的仮想仮想記憶の実現方式

各PEには、仮想記憶空間、他PEからのページを記録する輸入記憶空間、及び実記憶空間が存在する。前二者は実記憶空間が割り当てられ、そのいずれもがページアウトの対象となる。仮想記憶空間はPEに固有であり他PEとの間で空間の共有や移動は行なわないものとする。

ページングを実現するのは、ページャプロセスである。ページャプロセスはマルチスレッド構成になっており、ページフォルト等、発生するイベント毎にスレッドが生成され、互いにメッセージを交換しながら処理を行なう。

前章ではページ単位の使用頻度(アクセス時刻)の比較によりページの移動を行なうと述べたが、実際の実現では、PE単位でのメモリの使用頻度の比較を行ない、スワップ先PE(スワップ領域を提供するPE)を決定する。そこで、あるPE[i]上の実記憶空間中のページを過去にアクセスした順に10ページ程度のアクセス時刻の平均を取った値をPE[i]のメモリ使用頻度とし、それをU(i)と表記する。

各PE上の実記憶空間のページはフリーリストによって管理され、フリーの個数が予め決められた低水位指標を下回るとページャプロセス中にページアウト・スレッドが起動され、高水位指標まで回復を図る。その際、実記憶を占める1ページをページアウトする手順として、以下のようにページアウト処理を行なう。

1. 自PE[i]上の実記憶空間中から最も使用頻度の低いページをひとつ選択する。

2. スワップ領域探索法に従い、探索対象となる複数のPE(PE[n]で代表する)にそれぞれU(i)をパラメータとする入札要求メッセージを送信する。

3. 探索対象となる各PEにページャスレッドが生成される。PE[n]のページャスレッドはフリーリストが空ではなく、渡されたパラメータのU(i)と自PEのU(n)を比較し、小さい場合はU(n)をPE[i]に応札メッセージとして送信する。フリーページがない場合、あるいは、U(i)よりU(n)が大きい場合は入札に参加しない旨を通知する。

4. PE[i]は送られてきた入札の中から最も低いメモリ使用頻度を提示したPE[d]を選択し、仮想ページアウト要求を送信する。他のPEには入札失敗メッセージを送信する。どのPEからも入札がなかった場合、PE[i]は自PEを担当する二次記憶装置にページアウトを行なう。

5. PE[i]はPE[d]に仮想ページアウトを行なう。PE[d]はページの受け入れが終了すると、受け入れたページを保持していたPEの番号iとページ番号等の情報、及びそのページの使用頻度(最後にアクセスした時間)を輸入記憶空間のページテーブルに記録する。

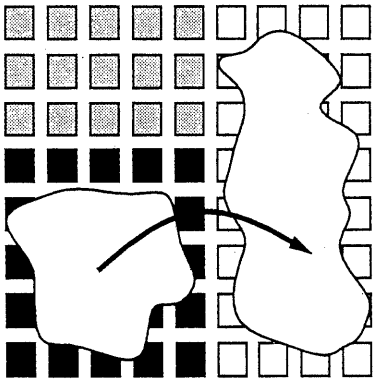

図 2: 遠隔 PE 群へのページング

4 スワップ領域探索法と評価

あるPEでメモリが不足して仮想ページアウトを行なう場合、メモリの使用頻度が低いPE上のメモリをスワップ領域として用いる。全PEを対象として最もメモリ使用頻度が低いPEを発見する事は多大なコストがかかり、現実的ではない。そこで、ほどほどに低いメモリ使用頻度のPEを低いコストで探索する「スワップ領域探索」の戦略が必要となる。例えば、探索空間に局所性を持たせ、DVMのように最近傍の数PEに限る等である。しかし、データパラレルのプログラムの存在を考慮すると、近傍のPEではメモリ使用傾向の同じプログラム(あるいはスレッド)が動作している可能性が高く、単純に近傍のPEを探索しても効果がないと考えられる。データパラレルでメモリを大量に消費するプログラムが動作するPE群のスワップ領域は、他のメモリの消費量が少ないプログラムが動作するPE群を捜し出してその上に確保することが望ましい(図2)。

探索法によってより適切なページを選択することが可能となる一方で、その探索法自身が必要とするコスト、即ち実行時間とネットワークトラフィックが不可避である。また、システム中で最も使用頻度の低いページを選ぶことより、自PEよりもほどほどにメモリ使用頻度が低いPEで、ネットワークトポロジ上でより近くにあるPEを選ぶ方がネットワークトラフィックが減り全体としての性能は高くなる可能性もある。

そこで、本論文では次の4種類の基本的な探索法を元にする7種類のスワップ領域探索法を比較する。

- コストフリー
- メタシェア (メタ情報共有メカニズム)
- ランダム
- DVM

また、各探索法に共通して付随する項目として、以下を検討する。

- 優先項目 – メモリ使用頻度優先かネットワーク上の距離優先か
- フォワードのホップ数の制限 – 輸入したページの他のPEへの転送を何ホップまで許すか。
- 入札対象PE数 – 入札をいくつのPEに対して行なうか。

まず、共通して付随する検討項目について説明し、次に探索法について説明する。

入札対象PE数 入札を行なう必要がある探索法(ランダムとメタシェア)においては、いくつかのPEに対して入札を行なう。入札の対象PE数が少ないと、適切なスワップ領域を有するPEを発見できず、対象PE数が必要以上に多いと、ネットワーク資源を浪費し、性能低下をもたらす。

フォワードのホップ数の制限 一度輸入したページを再び他のPEへ転送すること(フォワード)を任意の回数許すと、いつまでも使用されないページがPE間を「たらい回し」されてネットワーク等の資源を浪費する可能性がある(ある程度古くなればディスクへページアウトされる)。これを防ぐため、一定のホップ数フォワードされたページはディスクにページアウトした方がよい。逆に、このホップ数が少ないと、時間のかかるディスクへのページアウトが多量に発生する。両者のバランスがとれた適切なホップ数が存在するであろう。

メモリ使用頻度優先か距離優先か 入札やメタシェア等の手段によっていくつかのPEのメモリ使用頻度の情報を得た後、スワップ先PEを選択する方法は二つある。ひとつは、メモリの使用頻度が最も低いPEを選択する方法、もうひとつはメモリの使用頻度が自PEより低いPEの中からネットワーク上で最も近いPEを選択する方法である。この選択子はDVM以外のコストフリー、メタシェア、ランダムの各探索法と組み合わせることができる。

コストフリー探索法では、探索コストをかけることなくメモリ使用頻度の低いPEを発見できるものとする。即ち、本論文での評価をシミュレータ上で行なう点を生かして、入札を行なう代わりに全てのPEのメモリ使用頻度をネットワークを用いずに比較する。本探索法は実現不可能

であるが、他の探索法の比較対象としてひとつの指標を与えるために評価に加える。

優先項目がメモリ使用頻度優先であった場合、ネットワークを用いずに発見した最低メモリ使用頻度のPEをスワップ先PEとする。

優先項目がネットワーク上の距離優先であった場合、ネットワークを用いずに自PE[i]よりメモリ使用頻度が低いPE群(U(n)がU(i)×0.9より小さい[1])を発見し、そのPE群の中からネットワーク上で最短距離のPEを選ぶ。最短距離のPEが複数あった場合はそのうちのひとつを乱数で選択する。

ランダム探索法では乱数によって決定された数PEに対して入札を行なう。いくつのPEに入札を行なうかは入札対象PE数として可変にして評価を行なう。入札対象PE数が少ないと適切なスワップ先PEを発見する確率が下がり、多いとネットワーク資源を浪費する。入札に対して複数のPEが応札してきた場合、頻度優先あるいは距離優先でひとつのPEを選択し、スワップ先PEとする。

メタシェア探索法は超流動OSのメタ情報共有機構である「メタシェア[6]」の機能を使用する。メタシェアは各PE上でGVVMからメモリの使用頻度の情報を得ると、システム全体でその情報をデータパラレル的に比較し、メモリ使用頻度が低い順にいくつかのPEを選択してその情報をGVVMに提供する。メタシェアは一定の時間間隔でこの情報を更新する。

各PEはそれぞれ全PEのメモリ使用頻度を記録する「メモリ使用頻度マップ」を備える。ページアウト・スレッドはこのメモリ使用頻度マップを参考にして入札を行ない、スワップ先PEを決定する。メモリ使用頻度マップの記録はメタシェアが新たな情報を配布した場合、入札で他PEの使用頻度が判明した場合、及び、他PEから入札要求が来た場合に更新される。今回の評価で利用した簡易版メタシェアは情報収集及び配布にバイナリツリーを用い、次のような動作をする。

1. 情報収集フェーズ - 各PEは、バイナリツリーの子供のPEから送られてきたメモリ使用頻度と自PEのメモリ使用頻度を合わせ、小さい順に16PE分を選び、親に送る。これを根に当たるPE[0]まで繰り返す。
2. 情報配布フェーズ - 情報収集フェーズが終了した時点でPE[0]はシステム全体中でメモリの使用頻度が低い順に16PE分の情報を保持している。情報配布フェーズではバイナリツリーを根から葉へこの16PE分のメモリ使用頻度情報を配布する。

メタシェア探索法はメモリ使用頻度マップを利用し、自PEよりメモリの使用頻度が低いPEを頻度優先あるいは距離優先で入札対象PE個数選び、入札を行なう。入札に対して複数のPEが応札してきた場合、更に頻度優先あるいは距離優先でひとつのPEを選択し、スワップ先PEとする。

DVMでの仮想ページアウトは、ネットワークで直接隣接するPEへの転送に限定する。DVM[4]との比較を行なうため、この直接隣接に限定した転送も評価する。本方式はランダム探索法がランダムではなく隣接4PEになったものと考えればよい。

5 シミュレーションによる探索法の評価

上記のスワップ領域探索法と共通検討項目の比較を行なうため、シミュレータを作成し評価を行なった。以下に評価に用いた条件を示す。

- ネットワークは2次元トーラスでワード単位のストアアンドフォワード転送。混雑している場合はブロッキングをする。
- PEあたり実記憶50ページ、仮想記憶空間の大きさは150ページ[2]。ページサイズ100バイト。
- 実記憶のフリーリストの低水位指標は4ページ、高水位指標は6ページ。
- ページアクセスの書き込み率30%。

ネットワークの転送速度は二次記憶装置の転送速度に対し50:1の比を有するとする。これはリンク当たり100MB/秒のネットワークと2.0MB/秒のディスクを想定している。二次記憶装置はリモートI/Oの影響を避けるため各PEに存在するとする。

ページ使用の変化を把握しやすいよう、プログラムがアクセスするページのアクセス分布(ワーキングセット)は仮

[1] この値は経験的に決めた暫定値である

[2] 仮想記憶空間の大きさに対するGVVMの効果は[7]を参照のこと

想空間内で正規分布に従うものとする。また、その正規分布の分散はPE空間内で異なる分布を有するものとする。この仮定は挙動の異なる複数のプログラム/スレッドの混在を意図している。すなわち、PE空間内のある部分のPE上では仮想空間へのアクセスが集中する傾向を有し、ある部分では集中の度合が小さい。評価では、PEに0からN-1までの番号を付けたとすると、2/Nの番号を有するPE[2/N](PE空間の中央)が最も大きなワーキングセットを有し、仮想空間の100%を正規分布の99.74%がアクセスする。PE[2/N]を頂点として、PE[0]の方向とPE[N-1]の方向へ対称にワーキングセットが小さくなってゆき、PE[0]とPE[N-1](PE空間の端)では、最小のワーキングセットとなる。ここでは、仮想空間の33%を正規分布の99.74%がアクセスする。この様子を図3に示す。なお、時間の推移にかかわらず正規分布でのアクセスを行なうため、ワーキングセットが動くことはない。また、プログラムはPE間の通信を行なわない。

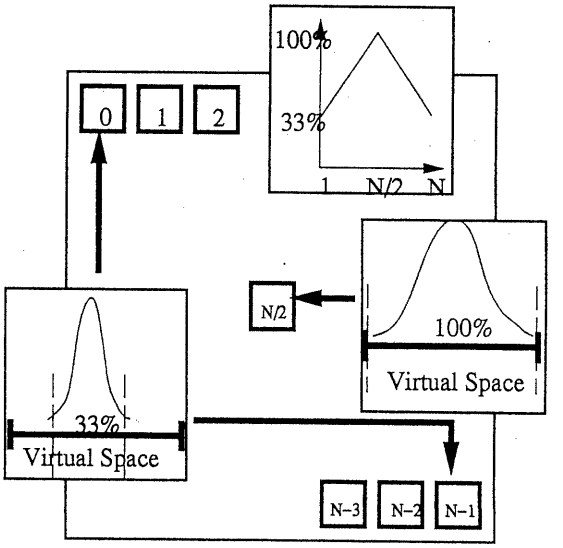

図 3: PE空間内での仮想空間アクセス分布

本評価では、各PE上のプログラムが上記の正規分布に従ってそれぞれ1500回仮想空間をアクセスする。PE空間の端に位置するPEはワーキングセットが小さいため、中央に位置するPEにスワップ領域を提供することができる。同様の理由で、PE空間の端のPEから中央に向けて順に1500ページのアクセスが終了し、メモリに空きが生じてゆく。優れたスワップ領域探索法はこの空きを素早く発見し、PE空間中央付近の高い負荷を緩和するであろう。

評価の尺度として、全てのPEが1500ページの仮想空間をアクセスし終るまでに要した実行時間を用いる。この時間はシミュレータの規定する仮想時間である。入札やページの転送等に要するネットワークの転送時間、及び、ディスクの転送時間は実行時間に含まれるが、CPUの使用時間は含まれていない。

5.1 入札対象PE数の効果

はじめに、メタシェア、ランダムにおいて、入札対象PE数の効果を調べた結果を図4に示す。64PEの場合と100PEの場合を図示してある。フォワードの制限ホップ数は2とする。横軸は入札対象PE数、縦軸は実行時間である。

図から、本評価の条件の元では、システム中のPE数にかかわらず入札対象PE数は3PEから4PEほどで十分であることがわかる。これより多くのPEに対して入札を行なっても、ネットワークが混雑するだけで意味がない。しかし、システム全体の負荷がもっと高い条件下ではより多くの入札が必要となるであろう。

5.2 探索法、優先項目、フォワード制限ホップ数の効果

64PEのシステムにおいて、各探索法、優先項目(距離優先か頻度優先か)、フォワードの制限ホップ数のいくつかを以下のように組合せ、評価を行なった。

メタシェア距離優先	(METASHARE NEAR)
メタシェア頻度優先	(METASHARE LRU)
コストフリー距離優先	(COSTFREE NEAR)
コストフリー頻度優先	(COSTFREE LRU)
ランダム距離優先	(RANDOM NEAR)
ランダム頻度優先	(RANDOM LRU)
DVM	(DVM)

結果を図5に示す。横軸はフォワードの制限ホップ数、縦軸は前項目と同様に実行時間である。仮想ページアウトされたページが再び他PEへ仮想ページアウトされる回数はフォワードのホップ数に制限される。その制限を超えた場合は、ディスクへページアウトされる。メタシェアを用いる探索法はメタシェアが消費するネットワークコストも計

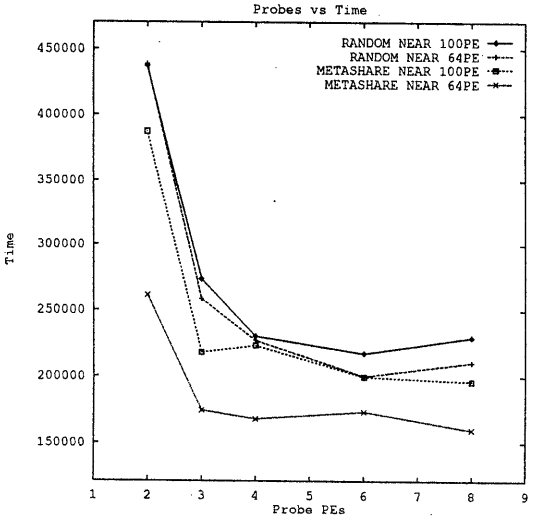

図 4: 入札対象 PE 数の効果

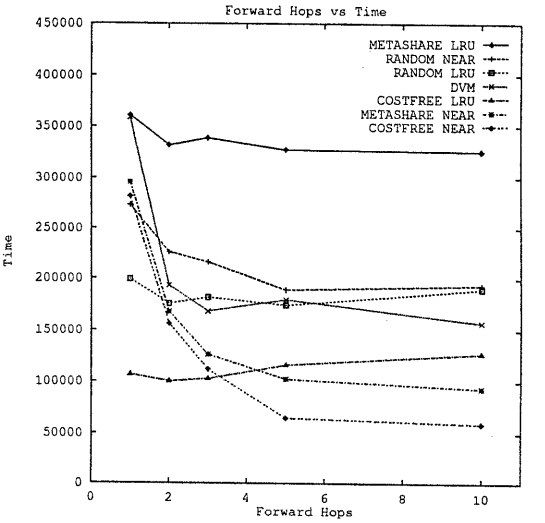

図 5: ホップ数に対する探索法の効果

測している。入札を行なう探索法では対象 PE 数は 4 である。

図中、LRU とラベルのついている頻度優先は、それ以外とは異なる傾向を示している。メモリ使用頻度を優先する場合、あるページがはじめて仮想ページアウトされて他の PE に転送される際にその時点で最もメモリの使用頻度が低い PE が直接選ばれて転送される。従って、ホップ数の影響をほとんど受けない。コストフリー頻度優先の場合は、最初からほぼ完全な使用頻度の比較がなされているため、ホップ数の制限を緩くすると必要以上のフォワードが発生して性能が低下する。メタシェア頻度優先の性能が悪いのは、メタシェアが GVVM に配布したメモリ使用頻度の低い PE へ仮想ページアウトが集中し、すぐにメモリを受け入れられなくなるためである。コストフリーも同様の状況が発生するが、コストフリーでは探索の度に新たなメモリ使用頻度の PE を探すことができるのに対し、メタシェアは一定時間間隔でしか情報を配布しないため、メモリ使用頻度の情報が古くなってしまう。

距離優先の場合、多くのホップ数が許されるほど、よりメモリ使用頻度の低い PE へページが移動するため、実行時間は短縮する。しかし、64PE のシステムであるため、そのネットワーク上の半径である約 5 ホップで実行時間は収束する。

全体の傾向として、頻度優先より距離優先の方が性能が高く、十分なホップ数が許されれば、コストフリー距離優先、メタシェア距離優先、DVM の順で良い性能を示している。

なお、GVVM を用いず、各 PE で独立にページングを行なった場合、PE 空間の中心部において多量のページング (ディスクアクセス) が発生するため、実行時間は 1260000 程になる。

図 6にネットワークと二時記憶装置の速度比が 8 倍と、二次記憶装置が比較的速い場合の結果を示す (図 5はネットワークと二時記憶装置の速度比が 50 倍)。図から、ほどんどの探索法において 2 ホップ程度で実行時間が収束していることがわかる。これは、ネットワークを用いて PE 間でページを移動するコストが、ディスクへページアウトするコストより、(二次記憶装置が遅い場合と比較して) 大きくなっていることを反映している。

5.3 システム規模に対する影響

前項で良い性能を示したコストフリー、メタシェア距離優先、DVM の各探索法とシステムを構成する PE 数の関係を評価した。入札を行なう探索法では対象 PE 数は 4 である。また、フォワードの制限ホップ数は、隣接 PE にし

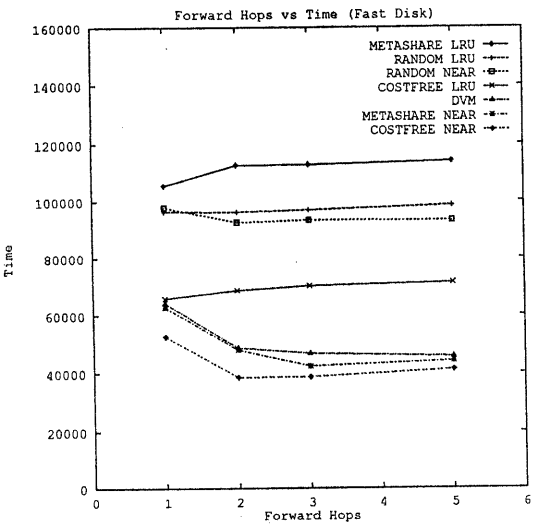

図 6: ホップ数に対する探索法の効果 (速いディスク)

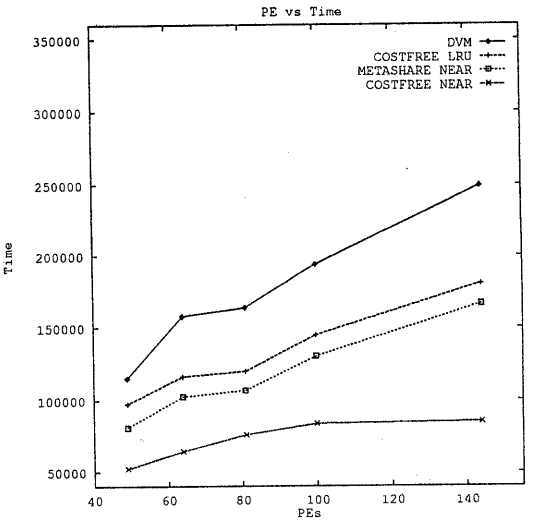

図 7: システム PE 数に対する探索法の効果

かページを移動することができない DVM は 10、それ以外は 5 とした。結果を図 7に示す。横軸はシステムの PE 数で、49PE から 144PE までをとり、縦軸は実行時間である。

コストフリー距離優先、メタシェア距離優先、コストフリー頻度優先の順で良い性能を示した。この順は、この程度の台数規模では PE 台数にあまり影響されない。理想的なコストフリーは実現不可能であるが、メタシェアを用いて距離優先でスワップ先 PE を決定すれば、コストフリーに近い性能を得ることが可能であろうと予測できる結果となった。

6 おわりに

本論文では大域的仮想仮想記憶 (GVVM) のスワップ領域探索法について、7 種類の異なった探索法を評価した。評価に用いた条件の元では、入札は 4PE に対して行なえば十分であること、超流動 OS のメタ情報共有メカニズムであるメタシェアを用いて距離優先でスワップ先 PE を決定すれば、ネットワークのコストなしでスワップ先 PE を決定できるコストフリー探索法に近い性能を得ることが可能であることが明らかになった。

今後はリモートメモリアクセスによるページ転送数の削減、及び、実機への実装と評価を行ないたい。

謝辞

本研究はリアルワールドコンピューティング計画の一環として「超並列システムアーキテクチャに関する研究」で行なわれたものである。関係各位に感謝する。

参考文献

[1] K. Li. IVY: A Shared Virtual Memory System for Parallel Computing. *Proc. of ICPP*, pp. 94–101, 1988.

[2] K. Li. Shared Virtual Memory on Loosely coupled multiprocessors. *Dissertation to Yale University*, 1988.

[3] K. Li and R. Schaefer. A hypercube shared virtual memory system. *Proc. of ICPP*, pp. I/125–I/132, 1989.

[4] M. Malkawi, M. Abaza, and D. Knox. Process Migration in Virtual Memory Multicomputer Systems. *Proc. of HICSS-26*, pp. 90–98, 1993.

[5] 須崎, 栗田, 田沼, 平野. 適応的アルゴリズム選択法による動的最適化. 第 *34* 回 プログラミングシンポジウム報告集, pp. 177–184, 1993.

[6] 田沼, 平野, 須崎. 超流動 OS のための管理情報共有機構 (MetaShare) の設計. *SWoPP'93 OS* 発表予定, 1993.

[7] 平野, 田沼, 須崎. 超並列システム用 OS 「超流動 OS」における大域的仮想仮想記憶. *JSPP'93*, pp. 237–244, 1993.

[8] 平野, 田沼, 須崎, 濱崎, 塚本. 超並列システム用オペレーティングシステム「超流動 OS」の構想. 情報処理学会研究報告 *93-OS-58*, Vol. 93, No. 27, pp. 17–24, 1993.