目次

図目次

表目次

1. はじめに

1.1 研究背景

近年、サーバの仮想化が注目を集めている。サーバを仮想化することでこれまで多くのサーバを運用する必要があった企業において、複数のサーバの OS 環境をそれぞれ仮想マシンとして単一のサーバ上に移行することでサーバの統合が可能になり、企業にとっては場所代、人件費、電気代などのコスト削減が可能になった。また、アプリケーションを実行した状態のまま物理マシンをまたがった仮想マシンのマイグレーションが可能なため、サーバにトラブルが発生した場合やサーバメンテナンスの場合にサービスを継続するための技術として多くの企業に利用されてきている。

仮想化技術をサーバに組み込むクラウドコンピューティングサービスは Amazon、NEC、富士通、NTT データ、IBM 等、多くの企業ですでに提供されている。これらの企業で提供されているサービスでは、サーバに VMware[1]や Xen[2]等のソフトウェアをサーバに組み込み、仮想化したサーバ群を顧客の需要に応じてリソースを切り分け顧客に提供している。このサービスを利用することで各企業は自社で所有するサーバ数を大幅に削減できるため、運用コストの低減につながる。

また、近年では仮想化技術を用いることにより低コストでサーバの運用が可能になるため、仮想化技術は資金力が先進国と比べて劣る新興国企業のニーズも高まってきており、各企業は世界各地にクラウドサービス用のデータセンターの建設を相次ぐ現状にある。

2010 年、NTT データでは中国企業へクラウドサービスを提供するために中国国内にデータセンターを設置することを発表し[3]、同年富士通は 1 千億円を投資し、米国、英国など 5 カ国にデータセンターを設置する方針を掲げるなど、各企業は積極的に海外でクラウドサービスを提供する環境を整えている[4]。

先進国ではサーバ等の IT 資産の投資の減少やコストカットのニーズが高く、新興国では安価なサーバ運用に関するニーズが高い現在の経済情勢下を考慮すると、仮想化の導入は今後さらに多くのサービスに広がっていくと考えられる。

サーバにかかる場所代、人件費、電気代などのコスト削減やメンテナンスのしやすさからサーバの仮想化が進む中、近年仮想化の High Performance Computing(HPC)分野での利用が注目されてきている[5][6][7][8]。Amazon が個人向けのクラウドサービスを開始し、仮想化したサーバを提供することで利用者は自分のニーズに合わせてカスタマイズした仮想マシンを利用でき、容易に大

規模な科学技術計算を行うアプリケーションを実行することが可能になったことが原因であったと考えられる。Amazon が提供しているようなサービスにおいて重要な点は計算資源をどのようにユーザに割り当てるかが挙げられる。

これまでグリッド環境やクラスタ環境おいて計算資源を割り当てるための手法は多く研究されてきた[9][10][11][12][13][14]。しかし仮想マシン環境においてはアプリケーションの実行に使用する物理マシンを選択しても、アプリケーションの実行に使用する仮想マシン数や仮想マシンに割り当てる物理コア数をさらに選択する必要があるため、既存の計算資源選択手法だけでは実行環境を選択できない。

仮想マシン環境でアプリケーションを高速に実行する研究として[15] [16]等が挙げられる。

[15]では、仮想マシン PC クラスタのストレージアクセス時に接続インターフェースとして IP ネットワークを利用する IP-SAN を導入し、サーバとストレージ間の広域環境における通信を低コストで行うことを可能にすることで、遠隔地にある計算資源同士を PC クラスタとして構成することを目的としている。

[16]では仮想マシン PC クラスタ環境で、通信の発生しないジョブを複数生成した際に、各物理マシンにジョブを割り当てる割り当て手法を提案している。

しかしいずれの研究においても、仮想マシンを物理マシンにどのように割り当てると最も実行時間が短くなるかに関しては言及されていない。そこで本研究では物理マシン数、仮想マシン数、物理コア数の異なるいくつかの仮想マシン実行環境でを実行し、MPI プログラムごとに最短実行時間を達成する仮想マシン実行環境を見つけ出し、その原因を考慮し、MPI プログラムごとに最短実行時間を達成する仮想マシン実行環境を選択する手法を提案する。

1.2　研究の目的

前節で述べた背景を踏まえ、本研究ではMPIプログラムに応じて最短実行時間を達成する仮想マシン実行環境を選択する手法を提案し、実験、検証することを目的とする。特に本研究では以下の点について重点を置き、MPIプログラムの仮想マシン実行環境の効率的な運用手法を見つけることを目指す。

●物理マシン数、仮想マシン数、仮想マシンに割り当てる物理コア数の組み合わせが異なる仮想マシン実行環境で各MPIプログラムを実行する。
●MPIプログラムごとに仮想マシン数、物理コア数、物理マシン数を組み合わせた仮想マシン実行環境と実行時間の間になんらかの規則性が見られるかどうか調査する。
●各MPIプログラムに応じて最短実行時間を達成する仮想マシン実行環境を選択する手法を提案する。
●ベンチマークプログラムを用いて提案手法の有効性を検証する。

1.3　本論文の構成

本章以後の本論文の構成は次の通りである。まず2章では仮想マシン数、物理コア数、物理マシン数の異なる仮想マシン実行環境及び非仮想マシン実行環境でMPIプログラムを実行した際の実行時間の測定結果及び考察を記述する。3章ではこれらの実験の結果から、各MPIプログラムを実行する際の実行時間と仮想マシン実行環境の間に見られる規則性を調べるための解析実験を行い、4章でMPIプログラムの仮想マシンへの割り当て手法を提案していく。

2. 各 MPI プログラムの規則性を調査するための予備実験

本章では種々の仮想マシン数、物理コア数、物理マシン数を組み合わせた仮想マシン実行環境及び非仮想マシン実行環境と実行時間の間の規則性を調査するための予備実験、及び考察を行った結果を記述する。

2.1. 予備実験環境

本研究の予備実験を行った際のハードウェア環境、及びソフトウェア環境を表 1 に示す。既存のクラウドコンピューティングサービスの代表として挙げられる Amazon EC2 サービスでは仮想化ソフトウェアとして Xen を用いているため、本研究においても Xen を用いるものとする。また、実験に使用するベンチマークプログラムとしては、NASA Advanced Supercomputing (NAS) Division で開発された NAS Parallel Benchmark(NPB)[17]を使用する。NPB を使用する理由として、種々のプロセス数や問題サイズにおいて容易に MPI プログラムを用いて実験できることが挙げられる。NPB では表 2 に示す 8 種類のベンチマークプログラムで構成されている。

本実験では NPB のうち BT、CG、LU、MG、SP、EP を用いて行い、この実験から得られた知見を元に仮想マシン実行環境を選択する手法を提案する。その後 NPB の FT、IS を用いて本研究の提案手法を適応し、適切な仮想マシン実行環境が選択されているか評価する。

実験環境の通信速度をネットワークベンチマークプログラム netperf[18]を用いて測定した結果を図 1 に示す。以後用いる各ベンチマークプログラムの実行環境を示す簡易図の汎例を図 2 に示す。

各ベンチマークプログラムにはそれぞれ問題サイズが class S、class W、class A、class B、class C、class D、class E の 7 種類を選択することが可能となる。各問題サイズは、ベンチマークプログラムの計算サイズや繰り返し回数が異なる特徴を持ち、class S が最もサイズが小さく、class E が最も大きくなる。本研究では実験環境のメモリ容量に合うサイズである問題サイズの中で最も大きな問題サイズである class B 及び classC について実験を行う。

表 1：本研究の実験環境

memory	DDR3 1333MHz 4GB×3
CPU	GenuineIntel Core(TM)i7 3.07GHz
Kernel	2.6.18-194.11.3.el5xen
物理マシン数	4

表 2：NPB の 8 種類のベンチマークプログラム

kernel		プロセス数
EP	乗算合同法による一様乱数、正規乱数の生成	2の乗数
MG	簡略化されたマルチグリッド法のカーネル	2の乗数
CG	正値対称な大規模疎行列の最小固有地を求めるための共役勾配法	2の乗数
FT	FFTを用いた3次元編微分方程式の解法	2の乗数
IS	大規模整数ソート	2の乗数
Simulated CFD Application Benchmarks		
LU	Synmetric SOR iterationによるCFDアプリケーション	2の乗数
SP	Scalar ADI iterationによるCFDアプリケーション	nの2乗
BT	5*5 block size ADI iterationによるCFDアプリケーション	nの2乗

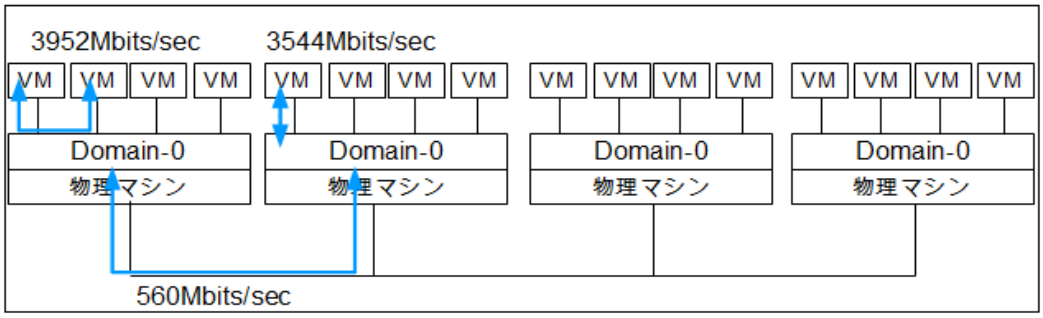


図 1：実験環境のネットワーク速度

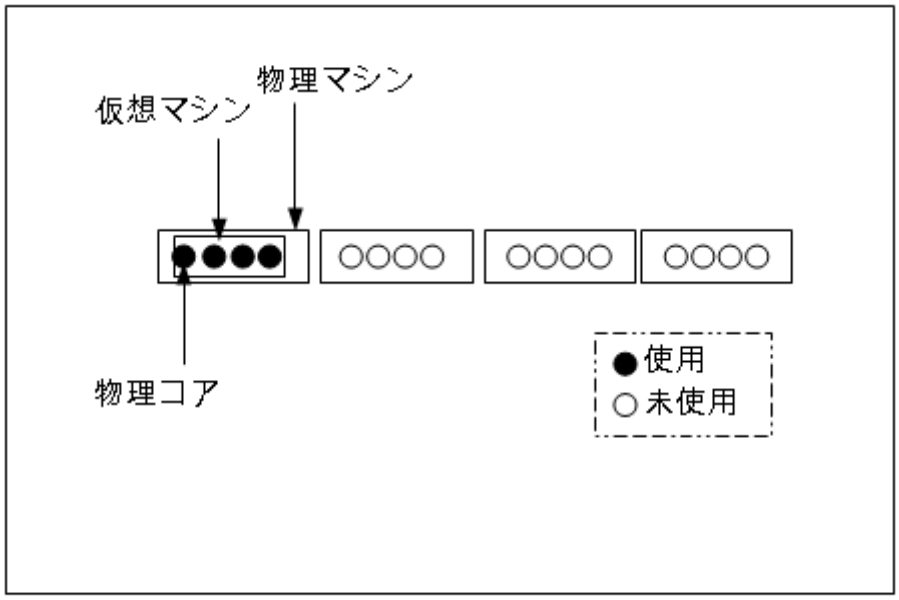


図 2 : 実 行 環 境 簡 易 図 の 汎 例

2.2　予備実験結果：プロセス数と実行時間との関係

仮想マシン環境及び非仮想マシン環境の両者において class B 及び class C の各ベンチマークプログラムについてプロセス数を変化させて予備実験を行った。class B と class C は同様の傾向を示しているため、class B の結果を本文中図 3～9 に示し、class C については付録に示した。また、図 3～9 では横軸は実行環境番号を表し、縦軸は各ベンチマークプログラムの最長実行時間を 1 とした場合の各実行環境における実行時間の相対値として表している。

表 3：実行環境対応表(class B,プロセス数＝2)

実行環境番号	実行に使用した2つの物理コアを配置
1	●●○○ ○○○○ ○○○○ ○○○○
2	●○○○ ●○○○ ○○○○ ○○○○
3	●●○○ ○○○○ ○○○○ ○○○○
4	●●○○ ○○○○ ○○○○ ○○○○
5	●○○○ ●○○○ ○○○○ ○○○○

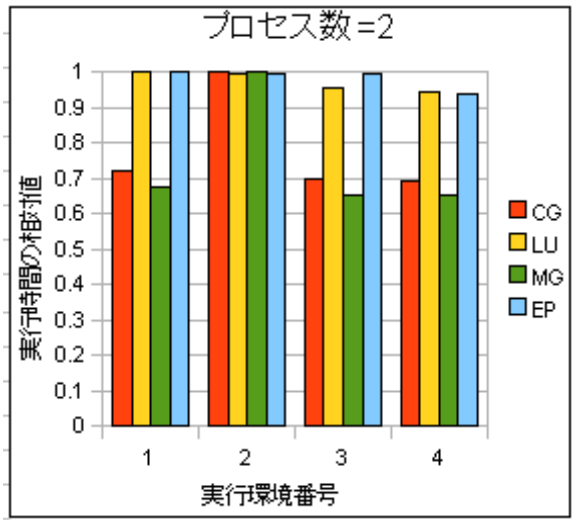


図 3：実行時間の相対値(class B プロセス数＝2)

表 4：実行環境対応表(class B,プロセス数＝4)

実行環境番号	実行に使用する4つの物理コア
1	●●●● ○○○○ ○○○○ ○○○○
2	●●○○ ●●○○ ○○○○ ○○○○
3	● ○○○ ● ○○○ ● ○○○ ● ○○○
4	●● ●● ○○○○ ○○○○ ○○○○
5	●●○○ ●●○○ ○○○○ ○○○○
6	●●●● ○○○○ ○○○○ ○○○○
7	●●●● ○○○○ ○○○○ ○○○○
8	●●○○ ●●○○ ○○○○ ○○○○
9	● ○○○ ● ○○○ ● ○○○ ● ○○○

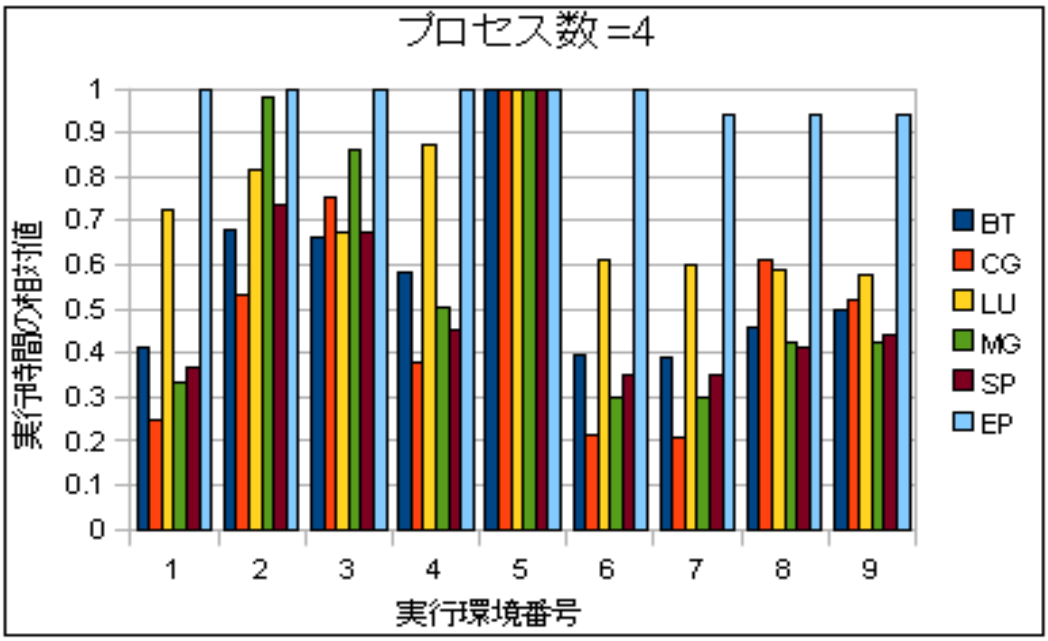


図 4：実行時間の相対値(class B,プロセス数＝4)

表 5：実行環境対応表(class B,プロセス数＝8)

実行環境番号	実行に使用する4～8つの物理コア
1	●●●● ○○○○ ○○○○ ○○○○
2	●● ○○ ●● ○○ ○○○○ ○○○○
3	● ○○○ ● ○○○ ● ○○○ ● ○○○
4	●●●● ●●●● ○○○○ ○○○○
5	●● ○○ ●● ○○ ●● ○○ ●● ○○
6	●● ●● ○○○○ ○○○○ ○○○○
7	●● ○○ ●● ○○ ○○○○ ○○○○
8	●● ●● ●● ●● ○○○○ ○○○○
9	●● ○○ ●● ○○ ●● ○○ ●● ○○
10	●●●● ○○○○ ○○○○ ○○○○
11	●●●● ●●●● ○○○○ ○○○○
12	●●●● ○○○○ ○○○○ ○○○○
13	●●●● ●●●● ○○○○ ○○○○
14	●● ○○ ●● ○○ ●● ○○ ●● ○○

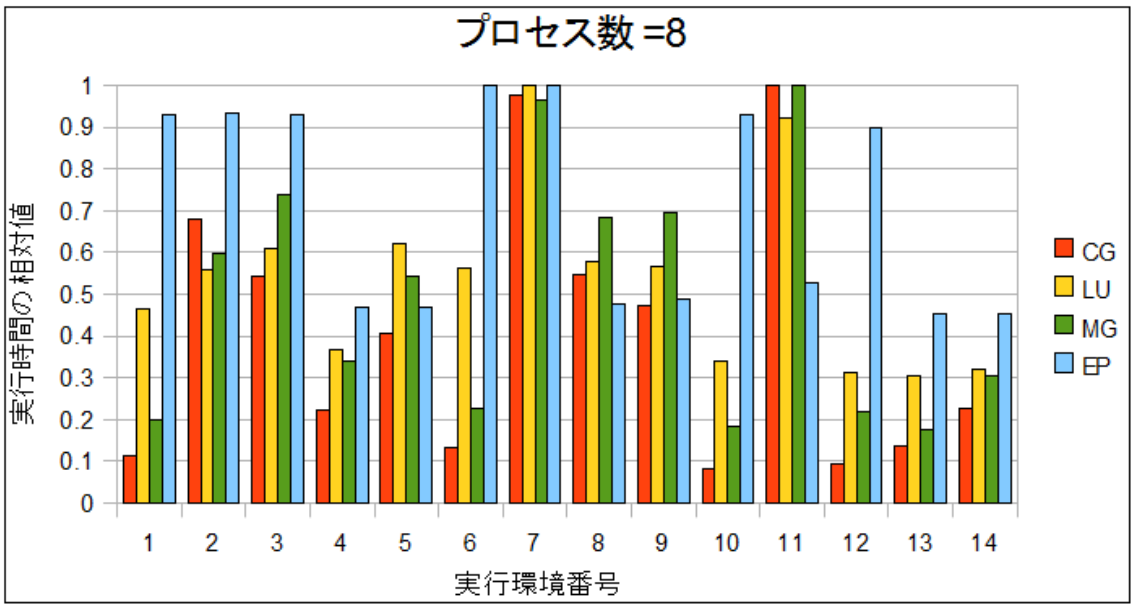
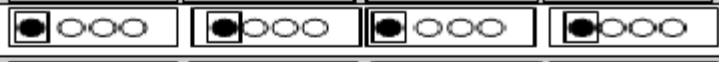


図 5：実行時間の相対値(class B,プロセス数＝8)

表 6：実行環境対応表(class B,プロセス数＝16~128)

実行環境番号	実行に使用した4～16個の物理コア
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	

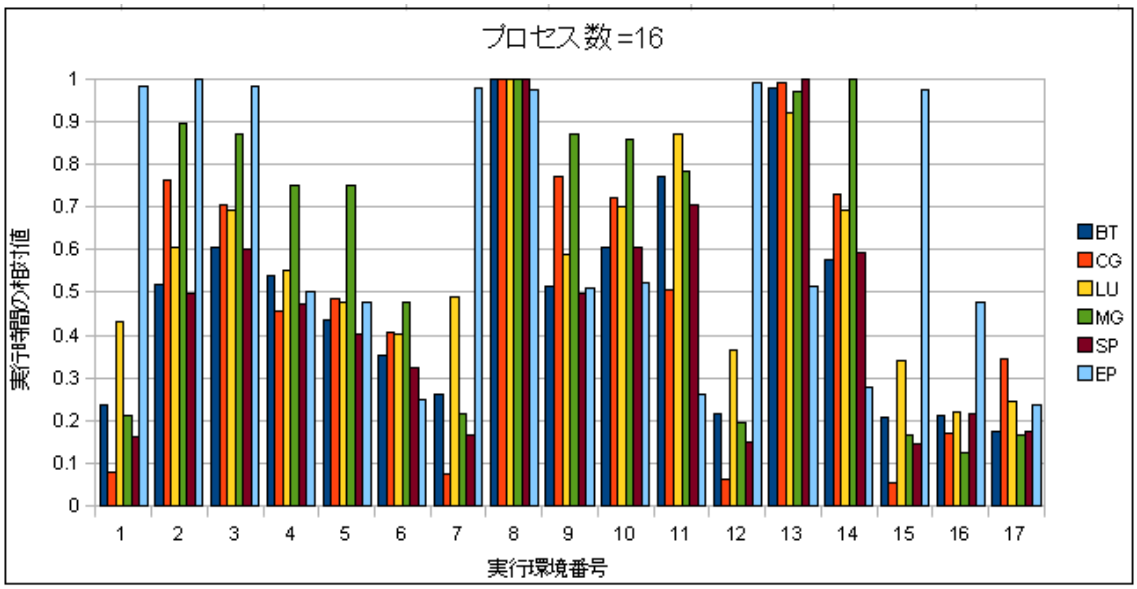


図 6：実行時間の相対値(class B,プロセス数＝16)

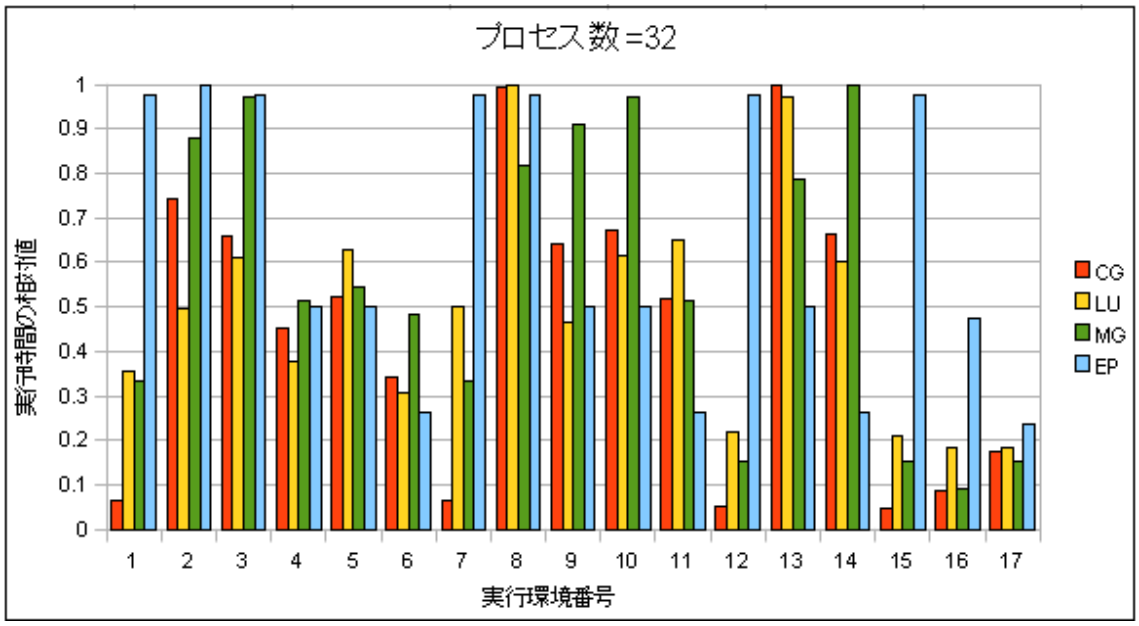


図 7：実行時間の相対値(class B,プロセス数＝32)

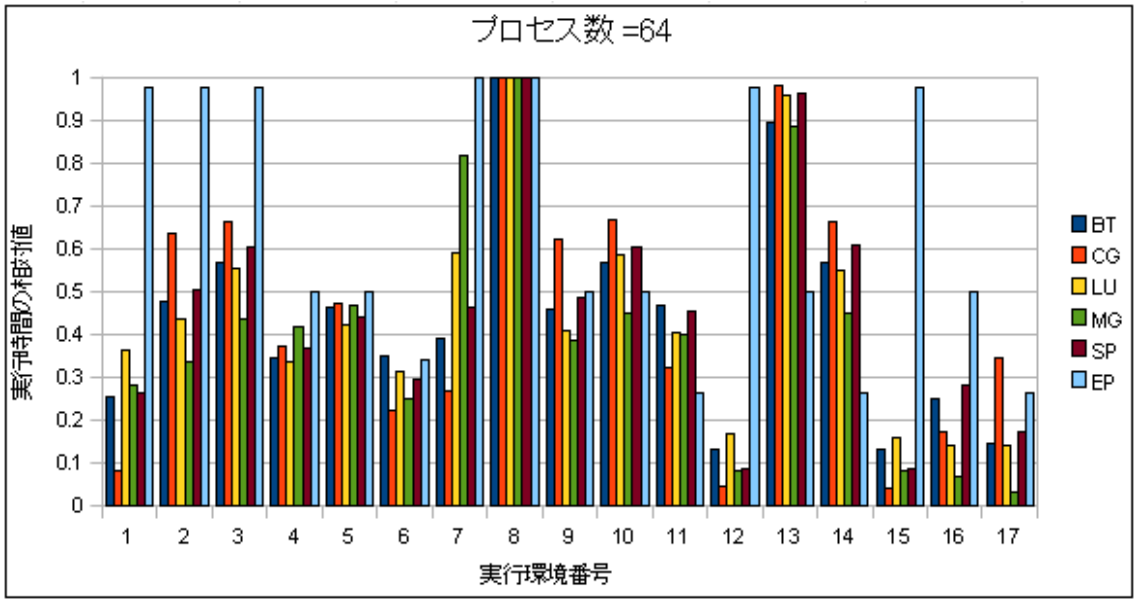


図 8：実行時間の相対値(class B,プロセス数＝64)

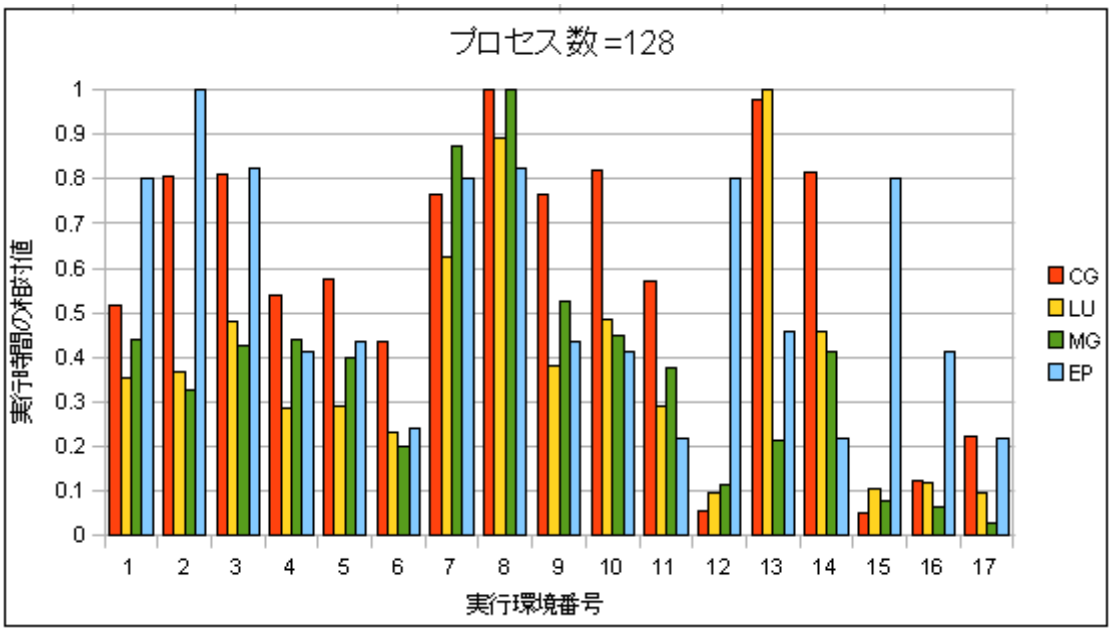


図 9：実行時間の相対値(class B,プロセス数＝128)

図 3〜9 から次のことがわかる。
(1)BT、CG、LU、MG、SP のいずれのプロセス数の場合においても表 4 の実行環境番号 6、表 5 の実行環境番号 10、表 6 の実行環境番号 12（1 つの物理マシン、1 つの仮想マシン、仮想マシンにはその物理マシンの全物理コア（4 コア）を割り当てる）で実行時間が最も短くなる。
(2)EP、LU 以外のすべてのベンチマークプログラムにおいて複数の物理マシンを実行に使用する場合には表 6 の実行環境番号 6（4 つの物理マシン、各物理マシンの全物理コア数と同数の仮想マシン、各仮想マシンには 1 つの物理コアを割り当てる）で実行時間が最も短くなる。
(3)非仮想マシン環境においてはいずれのプロセス数においても
BT、LU、SP、EP を実行する際には表 6 の実行環境番号 17（4 つの物理マシン）で実行時間が最も短くなり、CG を実行する場合には表 6 の実行環境番号 15（1 つの物理マシン）で実行時間が最も短くなる。
(4)非仮想マシン環境の MG においてはプロセス数が 32 以下の場合には表 6 の実行環境番号 16（2 つの物理マシン）で実行時間が最も短くなり、プロセス数が 64 以上になると表 6 の実行環境番号 17（4 つの物理マシン）で実行時間が最も短くなる。

以上の結果から仮想マシン環境と非仮想マシン環境では実行時間に異なる傾向があることが明確に示されている。また、これらの各傾向に対しては、以下の理由が考えられる。
(1)BT、CG、MG、SP のベンチマークプログラムはプロセス間通信量が多く、複数の物理マシンを使用すると物理マシン間通信がボトルネックとなるため、1 つの物理マシンで各ベンチマークプログラムを実行した場合に実行時間が最も短くなったものと考えられる。
(2)3 章において理由を探るための実験を行っている。
(3)Xen では同一物理マシン内の複数の仮想マシンが同時に物理マシンの外部と通信ができず、各仮想マシンは順番に通信する構造となっている。そのため複数物理マシン上の仮想マシン間通信が頻発すると通信時間が極端に増加するが、非仮想マシン環境で MPI プログラムを実行した場合にはそういった問題が発生しない。そのために、物理マシンを複数使用しても物理コアを多く使用できている分だけ実行時間が短くなっていると考えられる。しかし通信量が最も多い CG だけは通信時間が増加し、1 つの物理マシンで実行した方が複数の物理マシンで実行するよりも実行時間が短くなってると考えられる。

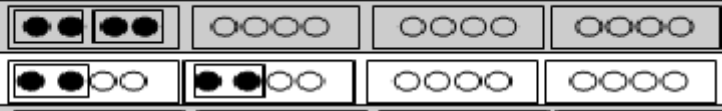
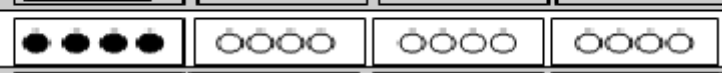
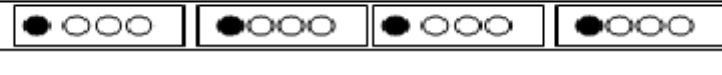
(4)プロセス数を増やし、プロセス数が 32 を超えると各物理コアでのプロセス切り替えオーバヘッドが大きくなるために MG の実行に使用する物理マシン数を増やした方が実行時間が短くなると考えられる。

これらの考察から、各 MPI プログラムごとに最適な仮想マシン実行環境を選択する際の指標として用いるべきは主として MPI プログラムのプロセス間通信量であることがわかる。

2.3 予備実験結果：物理コア数と実行時間との関係

2.2 の結果を同じ物理コア数の実行環境ごとに図にまとめたものを図 10～12 に示す。
横軸は「実行環境番号_プロセス数」、縦軸は各ベンチマークプログラムの最長実行時間を 1 とした場合の相対値で表すものとする。

表 7：実行環境対応表(class B,物理コア数＝4)

実行環境番号	実行に使用した4つの物理コア
1	
2	
3	
4	
5	
6	
7	
8	
9	

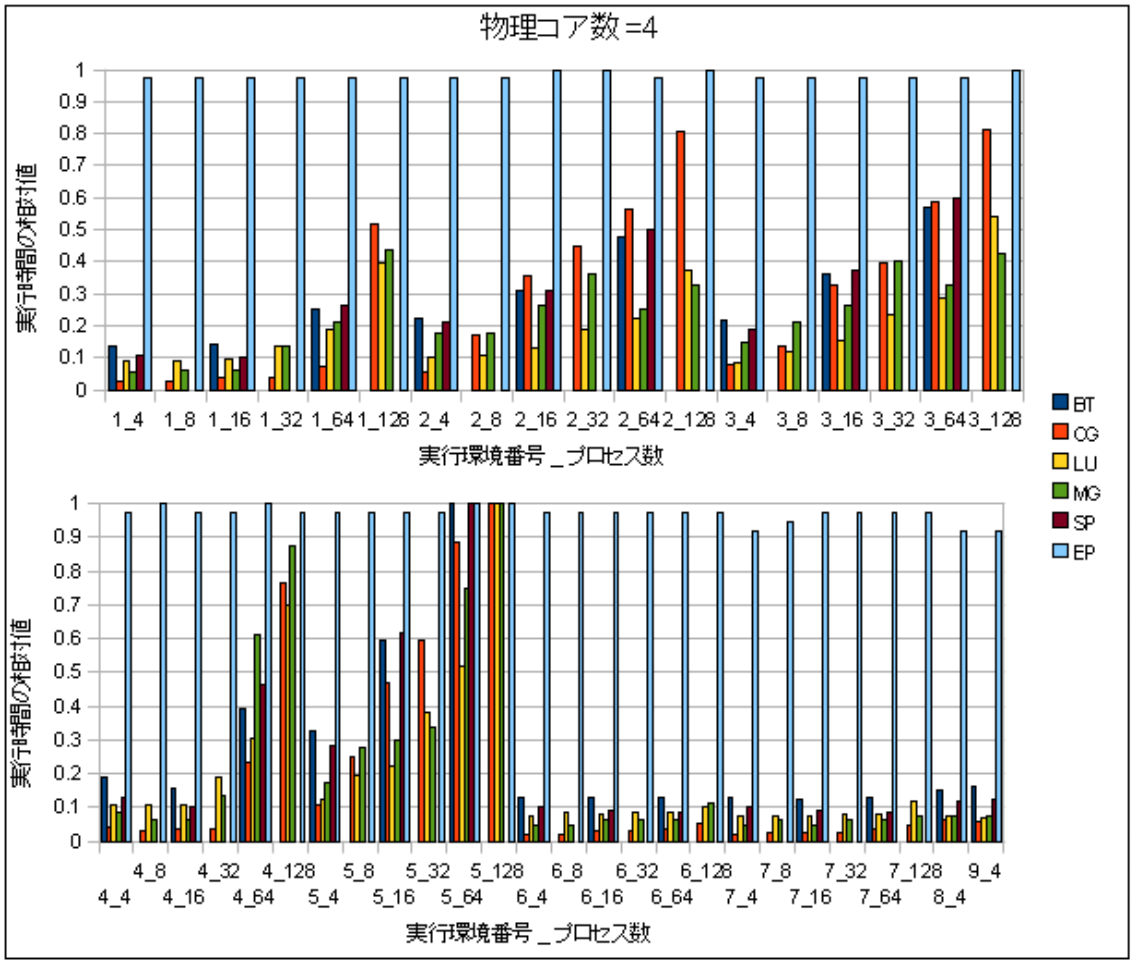


図 10：実行時間の相対値(class B,物理コア数＝4)

表 8：実行環境対応表(class B,物理コア数＝8)

実行環境番号	実行に使用した8つの物理コア
1	●●●● ●●●● ○○○○ ○○○○
2	●●○○ ●●○○ ●●○○ ●●○○
3	●●●● ●●●● ○○○○ ○○○○
4	●●○○ ●●○○ ●●○○ ●●○○
5	●●●● ●●●● ○○○○ ○○○○
6	●●●● ●●●● ○○○○ ○○○○
7	●●○○ ●●○○ ●●○○ ●●○○

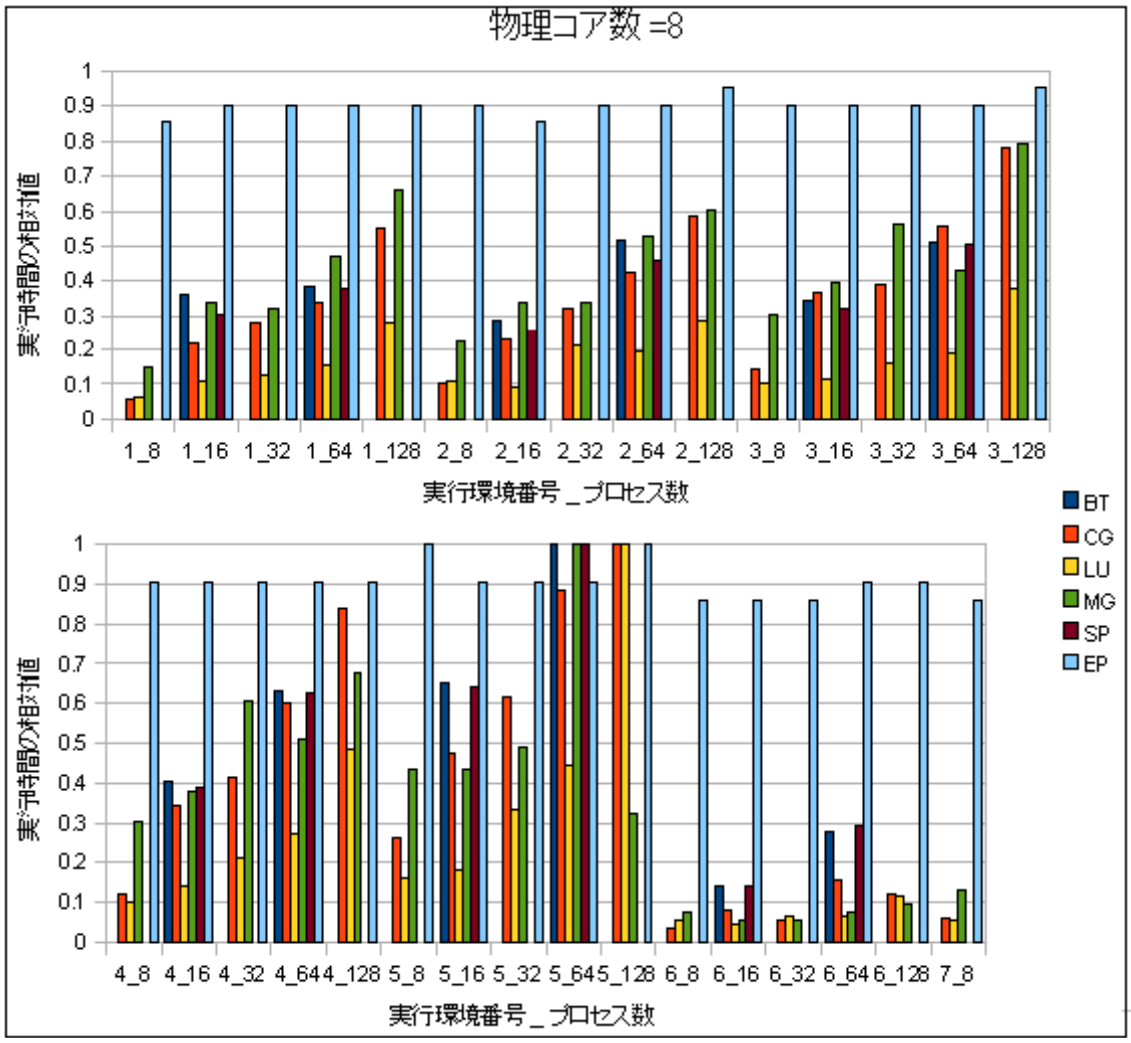


図 11：実行時間の相対値(class B,物理コア数＝8)

表 9：実行環境対応表(class B,物理コア数＝16)

実行環境番号	実行に使用した 16個の物理コア
1	
2	
3	
4	

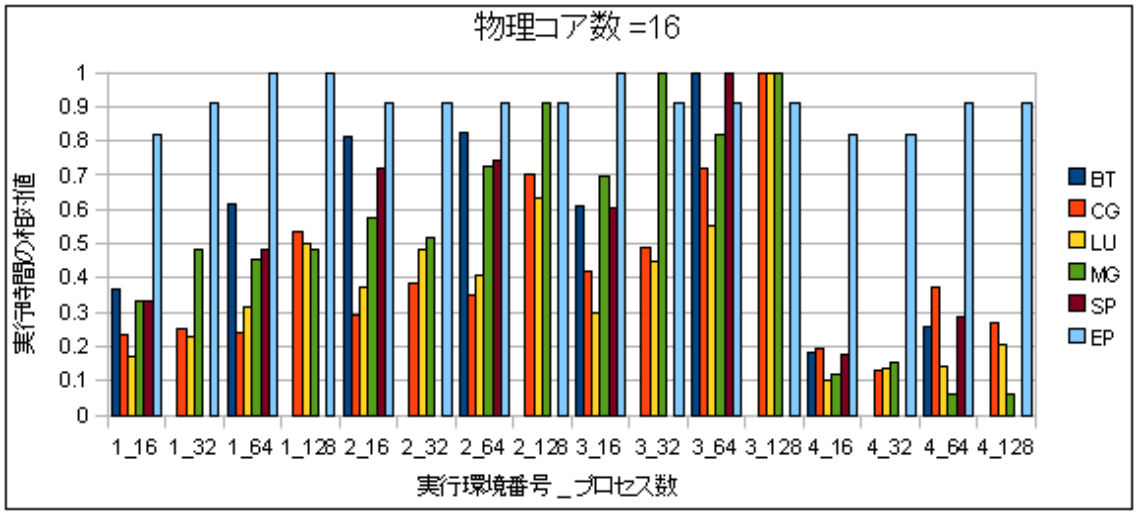


図 12：実行時間の相対値(class B,物理コア数＝16)

図 10~12 の結果から、次のことがわかる。
(1)全体的な傾向としてはベンチマークプログラムの実行に使用する物理コア数が等しい仮想マシン実行環境どうしで比較した場合にはプロセス数と物理コア数が等しい場合に最も実行時間が短くなり、プロセス数を物理コア数よりも増やすと実行時間は長くなる傾向にあることがわかる。
(2)表 7 の実行環境番号 1（1 つの物理マシン、物理コア数と同数の仮想マシン、各仮想マシンには 1 つの物理コアを割り当てる）と実行環境番号 4（1 つの物理マシン、2 つの仮想マシン、各仮想マシンには 2 つの物理コアを割り当てる）では各ベンチマークプログラムのプロセス数が 64 以上になると 32 以下の場合と比べて実行時間が長くなる結果となる。
(3)表 7 の実行環境番号 1（1 つの物理マシン、物理コア数と同数の仮想マシン、各仮想マシンには 1 つの物理コアを割り当てる）と実行環境番号 4（1 つの物理マシン、2 つの仮想マシン、各仮想マシンには 2 つの物理コアを割り当てる）では各ベンチマークプログラムのプロセス数が 64 以上になると実行環境番号 1 の方が実行環境番号 4 と比べて実行時間が短くなる傾向にあることがわかる。
(4)複数の物理マシンを使用する場合には全体的に仮想マシンに物理コアを 1 つ割り当てた場合が最も短くなり、物理コアを 4 つ割り当てた仮想マシンで実行すると最も長くなる傾向にあることがわかる。
(5)表 7 の実行環境番号 6（1 つの物理マシン、1 つの仮想マシン、仮想マシンにはその物理マシンの全物理コア（4 コア）を割り当てる）場合、仮想マシン環境は非仮想マシン環境の表 7 の実行環境番号 7（1 つの物理マシン）と同等の実行時間で実行できる。
(6)物理マシンを複数使用する場合には仮想マシン環境は非仮想マシン環境と比べて大幅に実行時間が長くなる結果となる。

これらの傾向の理由についての考察を以下に示す。
(1)プロセス数を増やすとプロセス間通信量が増加し、プロセス間通信時間の増大によって MPI プログラム実行時間が長くなると考えられる。
(2)プロセス数を増やすと、プロセス間通信量が増加しプロセス数が 64 以上になると単一物理マシン上の仮想マシン間通信がボトルネックとなっていることが原因であると考えられる。

(3)(4)このような傾向になる原因はわかっていないが、仮想マシン間通信では、1 つの仮想マシンに割り当てる物理コア数はできるだけ少なくした方が通信速度が速くなると考えられる。
(5)単一仮想マシンでは仮想マシン間通信がないため、異なる物理マシンに配置された仮想マシン間の通信時間が増加しやすいという Xen の欠点が現れず、非仮想マシン環境とかわらない実行時間で実行できたと考えられる。
(6)異なる物理マシン上に配置された仮想マシン間の通信時間が増加しやすい構造となっている Xen の欠点が顕著に現れ、非仮想マシン環境と比べて実行時間が長くなる傾向にあることがわかる。

また、以下では問題サイズ別に各ベンチマークプログラムで最短実行時間を達成した仮想マシン実行環境を表 10、表 11 に示す。

表 10：各ベンチマークプログラムの最短実行時間を達成した実行環境(class B)

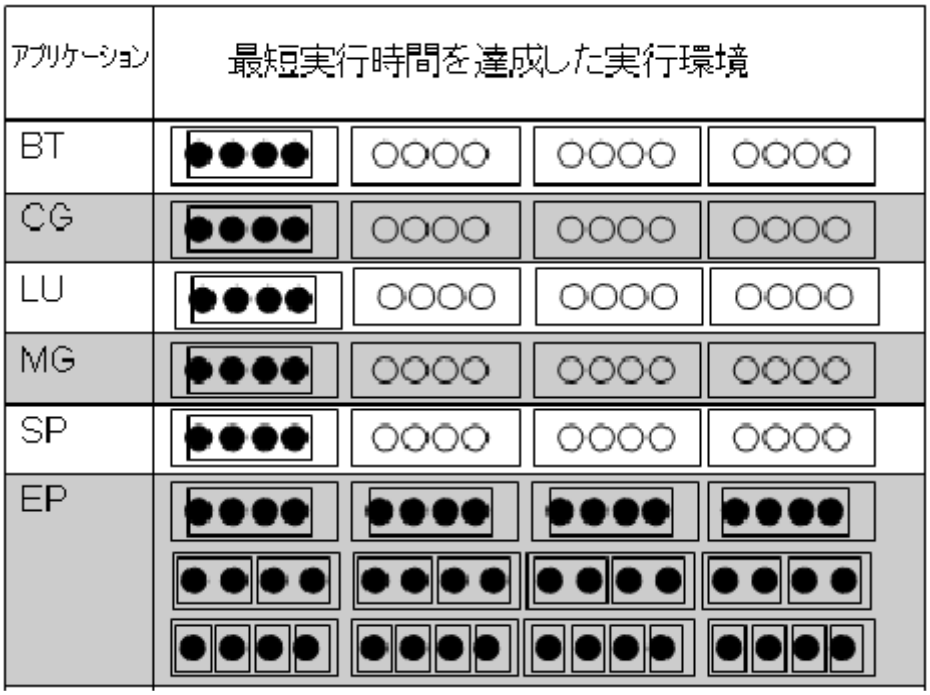
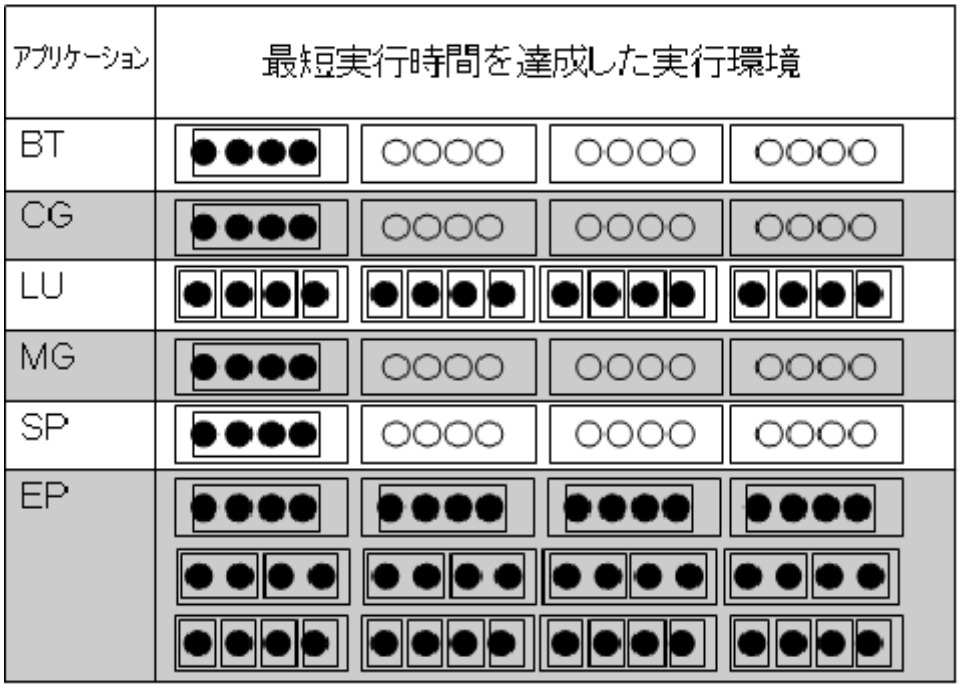


表 11：各ベンチマークプログラムの最短実行時間を達成した実行環境(class C)



結果的に LU(class C)、EP ではできるだけ多くの物理コアを仮想マシンに割り当てると、最短実行時間を達成し、それ以外のベンチマークプログラムにおいては１つの物理マシンで実行すると最短実行時間を達成した。

LU においては実行に使用する物理マシン数が問題サイズによって異なる結果となった。この原因として、LU のプロセス間通信量が少ないため、通信時間があまり増加せず、計算サイズが大きくなったことで物理コアの負荷が大きくなり、外部の物理マシンの物理コアも使用した方が実行時間が短くなった結果と考えられる。または問題サイズが大きくなったことでメモリアクセス量が増え、メモリアクセス時間が増大したため複数の物理マシンを実行に使用した方が実行時間が短くなったことも原因として考えられる。

この結果から、以降の章で最短実行時間を達成する仮想マシン実行環境を選択する指標について議論する。

3. 予備実験結果の解析実験

本章では前章の結果に関してさらに詳細な解析を行う。また各 MPI プログラムの最短実行時間を達成する仮想マシン実行環境を選択する際の指標を見つけ、実験、検証していく。

3.1 MPI プログラムログの解析

前章の予備実験において複数の物理マシンを実行に使用する場合には各 MPI プログラムの実行時間は全体的な傾向として仮想マシンに物理コアを 1 つ割り当てた場合最も短くなり、仮想マシンに物理コアを 4 つ割り当てた場合最も長くなる傾向にあることがわかったが、その原因は不明であった。また、その考察として前章で 1 つの仮想マシンに割り当てる物理コア数は少ない方が仮想マシン間の通信速度が速いと結論付けた。

この原因の調査と考察の正当性を確かめるために MPI プログラムのログ分析ツール jumpshot を使用して 2 つの実験環境での実行した BT のログを可視化して解析した。以下の図 13、図 14 では各実験環境ごとにプロセス数 64 の BT を実行した際のログを可視化したチャートを図に示している。

２つの物理マシンに物理コア数と同数の仮想マシンを配置し、各仮想マシンに 1 つの物理コアを割り当てた仮想マシン実行環境で実行した BT の実行ログを可視化したチャートを図 13 に示す。

2 つの物理マシンに物理マシン数と同数の仮想マシンを配置し、各仮想マシンに 4 つの物理コアを割り当てた仮想マシン実行環境で実行した BT の実行ログを可視化したチャートを図 14 に示す。

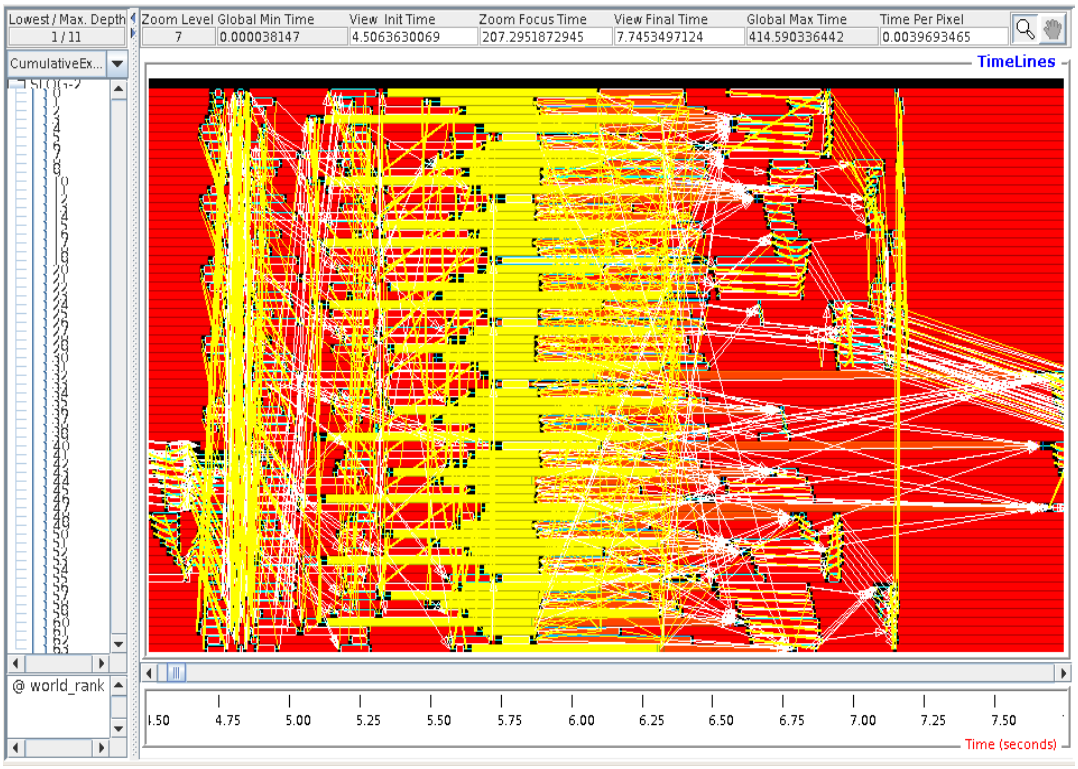


図 13：仮想マシンに物理コアを 1 つ割り当てた際の BT 実行ログ

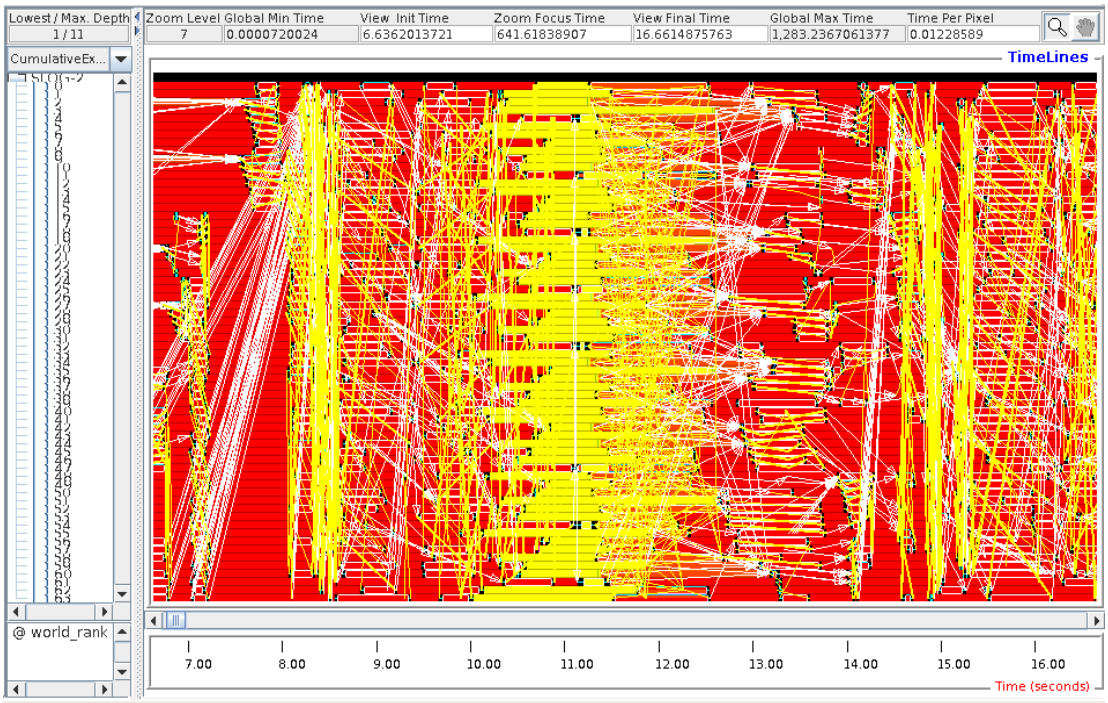


図 14：仮想マシンに物理コアを 4 つ割り当てた際の BT 実行ログ

図 13、図 14 中で矢印がプロセス間通信を表し、図の中央に位置する長方形の集合はバリア同期を表している。縦軸がプロセス番号、横軸は経過時間を表している。

図 13 では実行開始後 4.75 秒の時点で、図 14 では 8 秒の時点で通信が 1 度集約され、その後プロセス間通信を経てバリア同期に至っていることが見てとれる。この場合のプロセス間通信を開始してから一番最初にいずれかのプロセスがバリア同期を開始するまでの経過時間を調べると図 13 では 0.5 秒未満、図 14 では 2 秒かかっている。この結果から、２つの物理マシンに物理コア数と同数の仮想マシンを配置し、各仮想マシンに１つの物理コアを割り当てた仮想マシン実行環境の方が 2 つの物理マシンに物理マシン数と同数の仮想マシンを配置し、各仮想マシンに物理コアを 4 つ割り当てた仮想マシン実行環境よりもプロセス間通信時間が短いためにプログラム全体の実行時間が短くなることがわかり、考察の正当性は示すことができた。

しかし、この結果からは仮想マシンに割り当てる物理コア数が少ない方が多い方と比べて仮想マシン間通信の通信速度が速くなる原因はわからない。

3.2 各 MPI プログラムのメモリアクセス速度の測定

前章で各仮想マシン実行環境において、LU を実行する場合には物理マシン数が複数の方がメモリアクセス時間が短縮され、実行時間が短くなる可能性を示した。そこで各 MPI プログラムごとにどれだけメモリアクセス時間が変化するか実験を行った。実験にはメモリアクセス速度を測定するツール stream[19] を使用した。実験環境は１つの物理マシンに物理コア数と同数の仮想マシンを配置し、各仮想マシンには物理コアを１つづつ割り当てる。この仮想マシン実行環境において、次の 3 つの場合において仮想マシンでそれぞれ独立にベンチマークプログラムを実行しながら 1 つの仮想マシンでメモリの各速度を測定する。

●各ベンチマークプログラムを実行する仮想マシン数:1
●各ベンチマークプログラムを実行する仮想マシン数:2
●各ベンチマークプログラムを実行する仮想マシン数:3

結果を図 15～20 に示す。図 15～20 の横軸は各ベンチマークプログラムを実行した仮想マシン数、縦軸はメモリの copy、scale、add、triad に対する速度を表す。

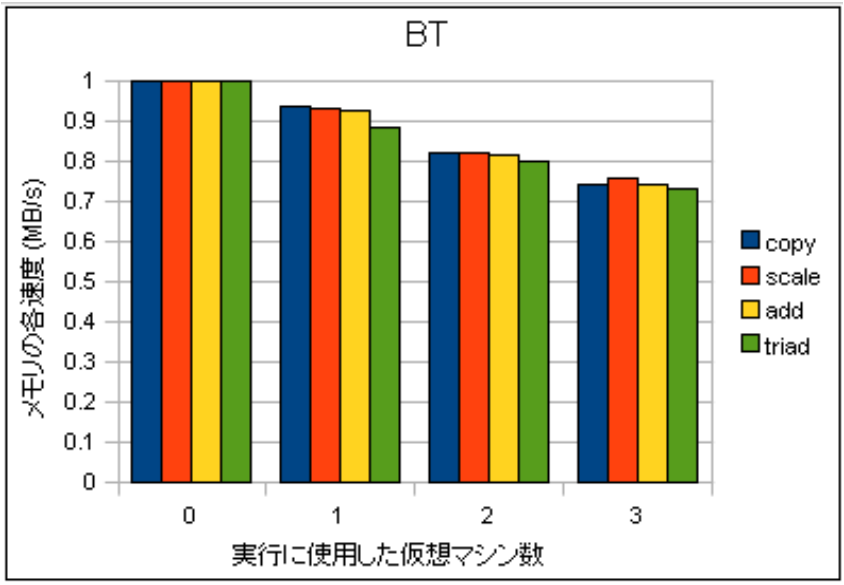


図 15：BT のメモリ速度の測定結果

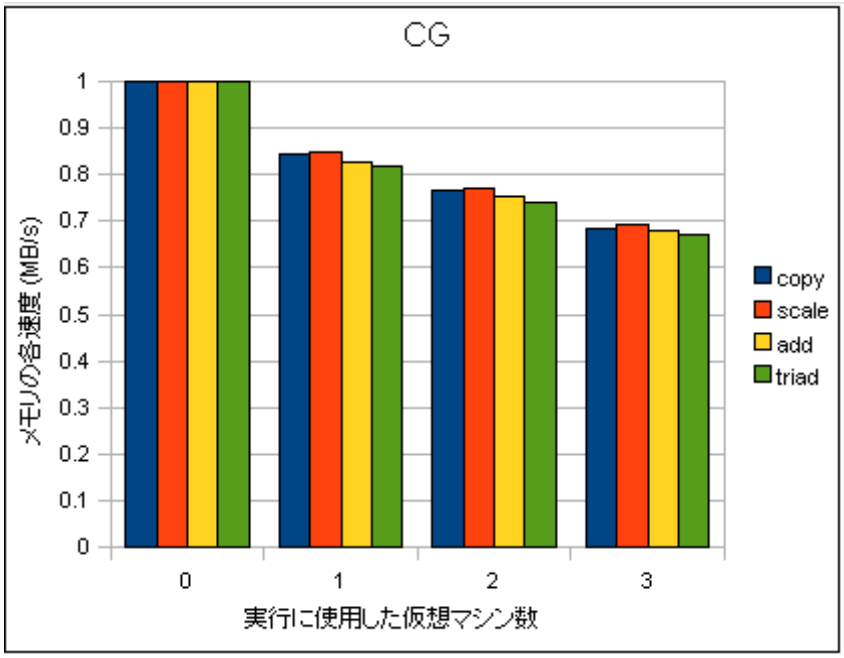


図 16：CG のメモリ速度の測定結果

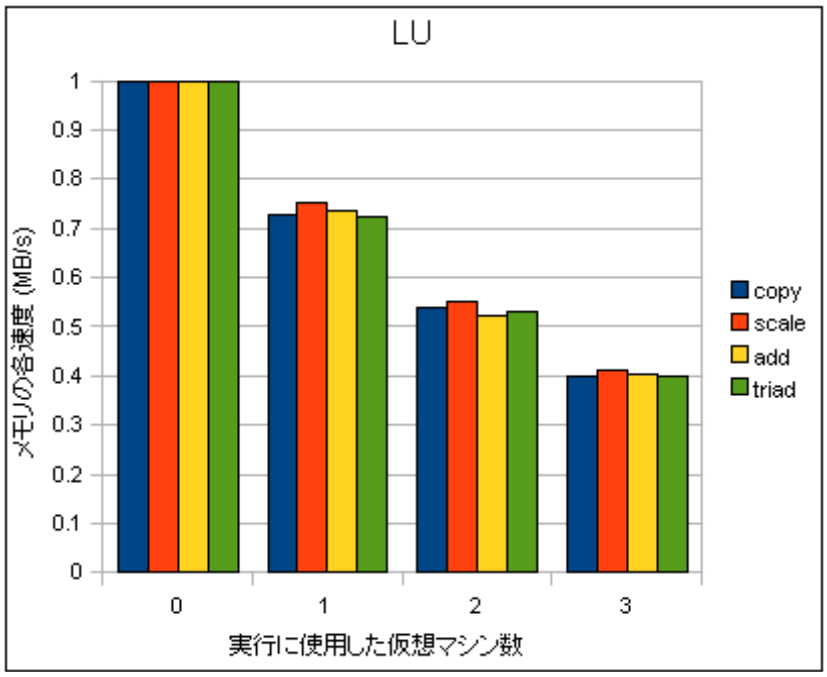


図 17：LU のメモリ速度の測定結果

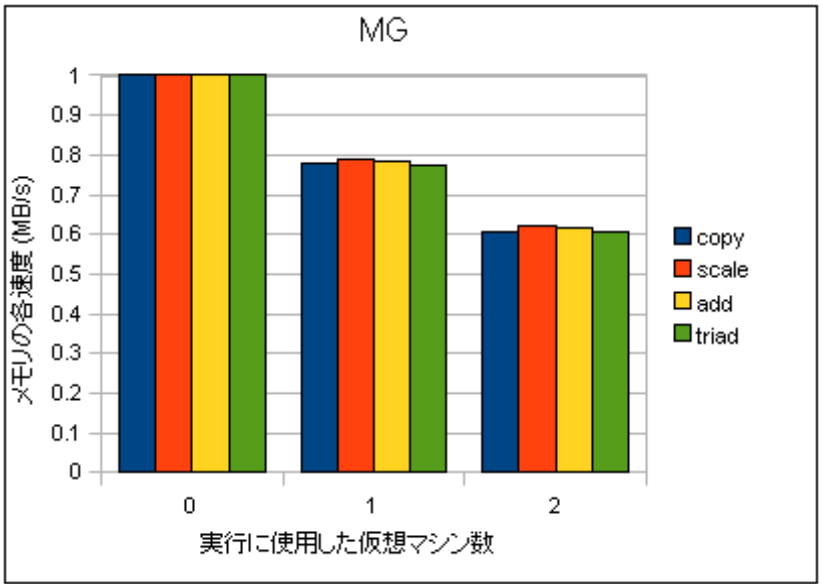


図 18：MG のメモリ速度の測定結果

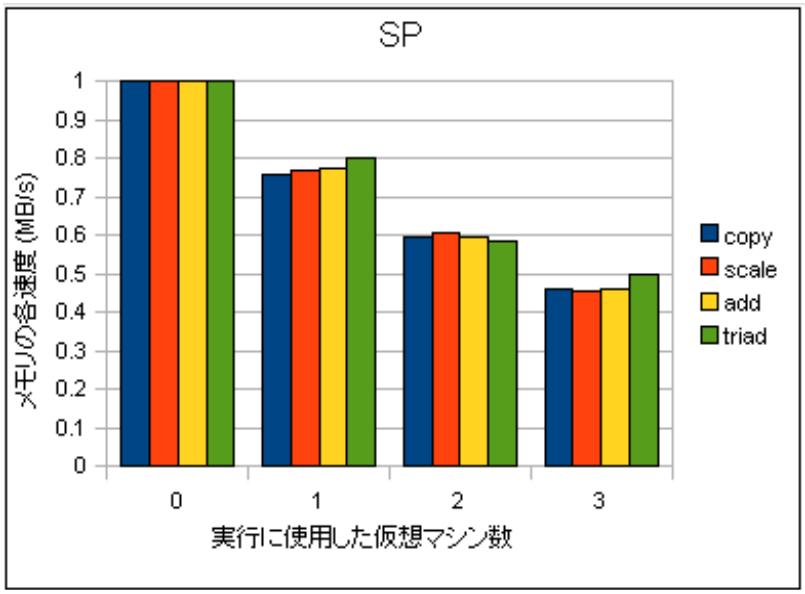


図 19：SP のメモリ速度の測定結果

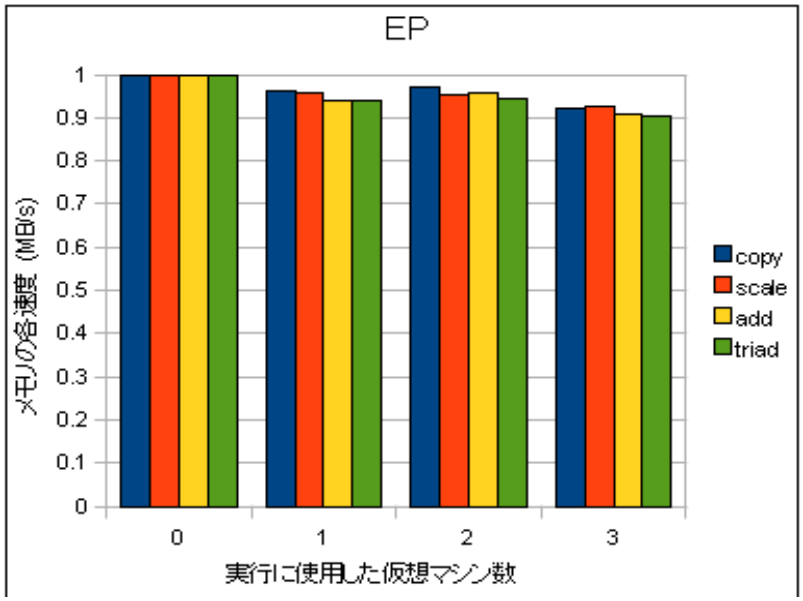


図 20：EP のメモリ速度の測定結果

MG についてはメモリ不足で仮想マシン数が 3 の場合には測定できなかった。結果として、LU や SP は実行する仮想マシン数が 3 の場合には仮想マシン数が 1 の場合と比べてメモリアクセス速度が 4 割に低下し、BT、CG は 7 割に低下していることがわかる。

この実験結果から、MPI プログラムによってメモリアクセス速度の低下度合いが異なり、特に MG、LU、SP の場合では大きく低下していることがわかる。これらの MPI プログラムにおいてはメモリアクセス速度の低下がボトルネックとなり、2 つの物理マシンの場合よりも 1 つの物理マシンの場合の方がプログラム全体の実行時間が長くなるとも考えられる。

しかし、表 11 と図 15～20 を比較した場合、メモリアクセスの低下が起こる可能性のある MG、LU、SP のうち、LU 以外は物理マシン数が 1 の場合に最も実行時間が短くなる結果となっているため、メモリの速度はあまり実行時間に影響を与えていないものと考えられる。

3.3 class B における CPU 使用率及びパケット送受信量の測定

各 MPI プログラムを実行する際に仮想マシンに割り当てられた物理コアをどの程度使用しているか把握するために各 MPI プログラムの物理コアの使用率を測定する。ここで使用率とは単位時間あたりの物理コア使用時間の割合であるものとする。以下では 2 つの物理マシンに物理マシン数と同数の仮想マシンを配置し、各仮想マシンに物理コアを 1 つ割り当てた仮想マシン実行環境において、各ベンチマークプログラムをプロセス数 4 で実行する際の各物理コアの使用率を測定した。各物理コアの使用率は動的に監視し、ベンチマークプログラム実行の開始から終了までの間の平均値を使用するものとする。また、物理コアを 2 つ使用するので 2 つの物理コア使用率が挙げられるが、どちらの物理コア使用率を選択しても値はほとんど変化しないため、特定の 1 つの物理コアの使用率を用いる。以後、特定の 1 つの物理コア使用率の平均値を CPU 使用率と呼ぶものとする。CPU 使用率を測定した結果を以下の図 21 に示す。

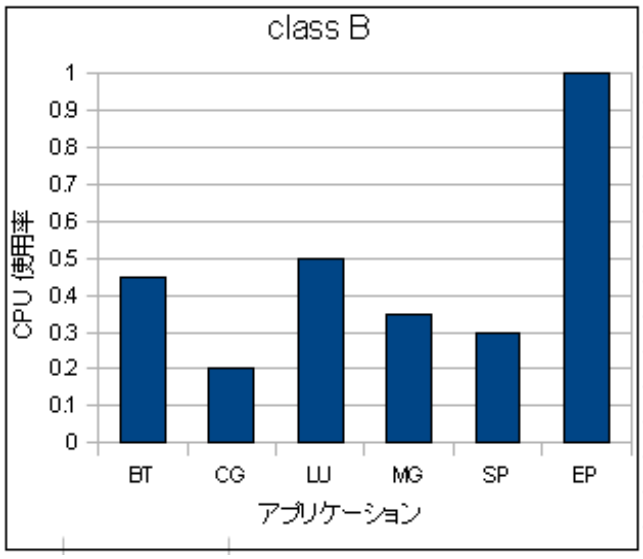


図 21：各ベンチマークプログラムを実行した場合の CPU 使用率(class B)

図 21 から、CG、BT、MG、SP の各ベンチマークプログラムの CPU 使用率は低くなり、EP は特に高く、LU はその中間くらいの値を示す結果になった。これは CG や SP の CPU 使用率が低いのは仮想マシン間通信がボトルネックとなり、与えられた物理コアを十分に使用できていないことが原因と考えられる。つまり CPU 使用率が高い MPI プログラムの場合は複数の物理マシンを実行に使用すると実行時間が短くなり、CPU 使用率が低い MPI プログラムの場合は 1 つの物理マシンを実行に使用すると実行時間が短くなると考えられる。この実験結果から、CPU 使用率の値は MPI プログラムごとに最短実行時間を達成する物理マシン数を選択するための判断材料になると考えられる。

図 21 をより詳細に分析すると、LU の CPU 使用率が 0.51、EP の CPU 使用率が 0.99 となった。表 10、表 11 の結果から、LU を実行する際の最短実行時間を達成する仮想マシン実行環境は 1 つの物理マシンの場合であり、EP を実行する際の最短実行時間を達成する仮想マシン実行環境は 4 つの物理マシンの場合であった。このことから物理マシン数を 1 にすべきか 4 にすべきかを決定する際の CPU 使用率による判断基準の値は 0.51～0.99 の間に存在すると考えられる。

次に各 MPI プログラムごとのネットワーク速度の重要度を把握するために各ベンチマークプログラムを実行した際の 1 秒あたりのパケット送受信量を測定

した。この実験では２つの物理マシンに物理マシン数と同数の仮想マシンを配置し、各仮想マシンに物理コアを１つ割り当てた仮想マシン実行環境において、プロセス数 4 の各ベンチマークプログラムを実行した場合の 1 秒あたりのパケット送受信量を測定した。実験結果を以下の図 22 に示す。

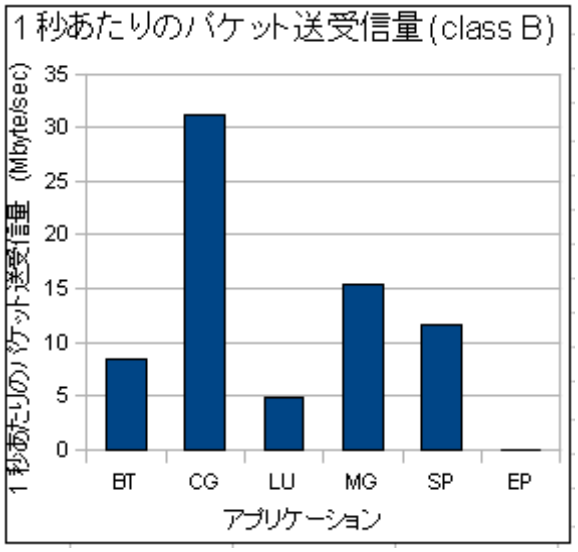


図 22：各ベンチマークプログラムの 1 秒あたりのパケット送受信量(class B)

図 22 から 1 秒当たりのパケット送受信量は CG の値が最も高く、EP の値が最も低いことがわかる。この結果から、1 秒あたりのパケット送受信量が多い MPI プログラムは１つの物理マシンを使用する場合の方が複数の物理マシンを使用する場合と比べて実行時間が短くなり、逆に 1 秒あたりのパケット送受信量が少ない MPI プログラムは複数の物理マシンを実行に使用する場合の方が 1 つの物理マシンを実行に使用する場合よりも実行時間が短くなると考えられる。

図 10、図 11 から、LU の最短実行時間を達成する仮想マシン実行環境は物理マシン数が 1 の場合であり、EP の場合の最短実行時間を達成する仮想マシン実行環境は物理マシン数が 4 の場合である。よって実行に使用する物理マシン数を 4 にすべきかどうかの判断するための 1 秒あたりのパケット通信量による判断基準の値は 1~5MB/sec になると考えられる。

問題サイズが class B の場合のみで判断した場合、CPU 使用率と 1 秒あたりのパケット送受信量の値は MPI プログラムを実行する際、最短実行時間を達成する仮想マシン実行環境を選択する際に役立つ材料になると考えられる。

3.4 class C における CPU 使用率及びパケット送受信量の測定

class B 同様、class C についても CPU 使用率と 1 秒あたりのパケット送受信量を測定した。仮想マシン実行環境は 2 つの物理マシンに物理マシン数と同数の仮想マシンを配置し、各仮想マシンには 1 つの物理コアを割り当てた仮想マシン実行環境においてプロセス数 4 及び 64 の各ベンチマークプログラムを実行するものとする。結果を図 23、図 24 に示す。

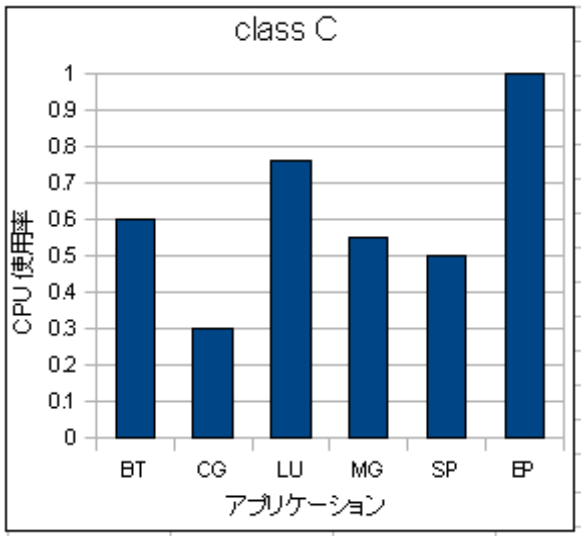


図 23：各ベンチマークプログラムを実行した場合の CPU 使用率(class C)

図 21、図 23 から class B の場合と同様、EP の CPU 使用率が高くなった。また、問題サイズが class C の LU における CPU 使用率は class B の場合よりもかなり大きな値となった。表 11 から、class C の最短実行時間を達成する実行環境の物理マシン数は 4 のため、CPU 使用率の高さから物理マシン数を複数にすべきかどうかの判断材料になる結果となった。図 30 から、問題サイズが class C の MG、BT においては表 19 の 6（4 つの物理マシン、各物理マシンの全物理コア数と同数の仮想マシン、各仮想マシンには 1 つの物理コアを割り当てる）の場合と表 19 の 12（1 つの物理マシン、1 つの仮想マシン、仮想マシンにはその物理マシンの全物理コア（4 コア）を割り当てる）の場合で実行時間の差が 6%以内となる結果となった。そのためこれらの CPU 使用率である 0.55 から 0.6 までの間が実行に使用する物理マシン数を 4 つにすべきかどうかが決めるための基準となる値になると考えられる。

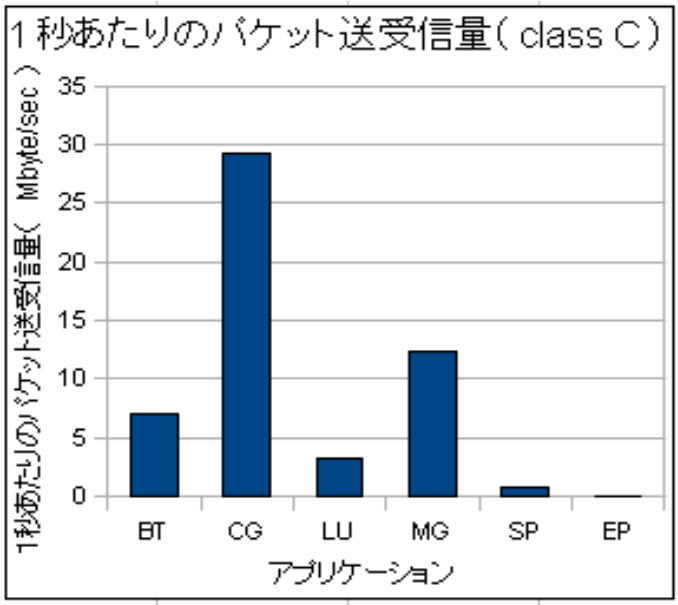


図 24：各ベンチマークプログラムの 1 秒あたりのパケット送受信量(class C)

図 22、図 24 から、各ベンチマークプログラムの 1 秒あたりのパケット送受信量は class B と class C では異なる結果となった。問題サイズが class C の多くのベンチマークプログラムにおいてパケット送受信量は class B の場合と比べて増加した一方で SP だけは極端に減少する結果となった。

図 30 から、問題サイズが class C の MG、BT においては表 19 の 6（4 つの物理マシン、各物理マシンの全物理コア数と同数の仮想マシン、各仮想マシンには 1 つの物理コアを割り当てる）の場合と表 19 の 12（1 つの物理マシン、1 つの仮想マシン、仮想マシンにはその物理マシンの全物理コア（4 コア）を割り当てる）の場合で実行時間の差が 6%以内となる結果となった。そのため MPI プログラムの実行に使用する物理マシン数を使用できる最大物理マシン数にすべきかどうかをきめるための 1 秒当たりのパケット送受信量の基準となる値は MG、BT の値である 7～13MB/sec となると考えられる。

class C の SP を除いて考えると、1 秒当たりのパケット送受信量が少ない MPI プログラムであれば複数の物理マシンを割り当て、1 秒当たりのパケット送受信量が多い MPI プログラムは 1 つの物理マシンを割り当てればいいように見える。しかし図 22、図 24 を合わせて考慮すると、class B では実行に使用する物理マシン数を使用できる最大物理マシン数にすべきかどうかの基準となる

値が 1～5MB/sec だったにも関わらず、class C においては 7～13MB/sec に変化しているため、問題サイズが異なると同一の基準値を用いて物理マシン数を選択できない結果となったため、1 秒当たりのパケット送受信量は MPI プログラムの最短実行時間を達成する実行環境の物理マシン数を決定するためにはあまり役に立たない結果となった。

3.5 予備実験結果の解析実験のまとめ

class B と class C での実験結果から、MPI プログラムの最短実行時間を達成する実行環境の物理マシン数を決定する際には CPU 使用率の値が有用であり、実行に使用する物理マシン数を 4 つにすべきかどうかの基準となる値が 0.55～0.6 の間に存在し、この値を超える場合には実行に使用する物理マシン数を 4、超えない場合には実行に使用する物理マシン数を 1 を選択すると最短実行時間を達成する実行環境の物理マシン数を選択できることがわかった。

4. MPI プログラムの仮想マシンへの割り当て手法の提案

本章ではこれまで行った予備実験から得た知見から、各 MPI プログラムごとに最短実行時間を達成する仮想マシン実行環境を選択する手法を提案し、実験と考察を行っていく。

4.1 MPI プログラムの仮想マシンへの割り当て手法

予備実験から得た知見を用いて MPI プログラムの仮想マシンへの割り当て手法を検討する。本研究の前提条件として次の 3 点を挙げる。

●本研究の実験環境として 4 つの物理マシンを使用するものとする。
●ユーザは実行したい MPI プログラムと問題サイズを選択する。
●実行に使用する仮想マシン数、仮想マシンに割り当てる物理コア数、仮想マシンの物理マシンへの配置はシステム運用者によって決められるものとする。

これらの前提条件で、MPI プログラムに応じて最短実行時間を達成する仮想マシン実行環境を選択する手法の前提条件を以下に示す。

●MPI プログラムの実行に使用するプロセス数は実行に使用する物理コア数と同数とする。
● 2 つの物理マシンに、物理マシン数と同数の仮想マシンを配置し、各仮想マシンには 1 つの物理コアを割り当てる仮想マシン実行環境において、プロセス数 64 の MPI プログラムを実行した際の CPU 使用率があらかじめ測定されているものとする。
●実行に使用する物理マシン数が 1 の場合、その物理マシンに 1 つの仮想マシンを配置し、仮想マシンにはその物理マシンのすべての物理コア（4 つ）を割り当てるものとする。
●実行に使用する物理マシン数が 4 つの場合は物理マシンに物理コア数と同数の仮想マシンを配置し、各仮想マシンに 1 つの物理コアを割り当て、実行するものとする。
●使用する物理マシン中の物理コアはすべて使用するものとする。

これらの前提条件の下、本提案手法ではMPIプログラムの実行に使用する物理マシン数を決定する際にCPU使用率の値を利用する。この値の大きさに応じて実行に使用する物理マシン数が決定する。

CPU使用率の値に応じた物理マシン数を表現するのに最も単純に対応表を用いるものとする。

実行対象のMPIプログラムのCPU使用率から実行に使用する物理マシン数を示す表を作成し、これを物理マシン数対応表と呼ぶものとする。これを以下の表12に示す。

表12：物理マシン数対応表(n=MPIプログラムのCPU使用率)

CPU使用率	0.0<=n<=0.6	0.6<n<=1.0
物理マシン数	1	4

物理マシン数対応表から物理マシン数を選択する。

実行に使用する物理マシン数を決定した後、物理マシン数に応じて仮想マシンの構成を決定する。

4.2 実験と考察

本手法の有効性を確認するために、これまで予備実験で用いていないベンチマークプログラムのFTとISに本手法を適用し、仮想マシン実行環境を選択する。実験結果を
13に示す。

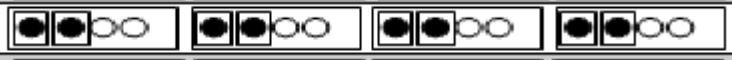
表13：FT及びISにおける実験結果

MPIプログラム	問題サイズ	CPU使用率	物理マシン数
FT	class B	0.4	1
	class C	0.5	1
IS	class B	0.4	1
	class C	0.4	1

表13に示す通り、4通りのベンチマークプログラムに対してすべて1つの物理マシンを仮想マシン実行環境として選択した。そのため、1つの物理マシンに1つの仮想マシンを配置し、その仮想マシンに4つの物理コアを割り当てた仮想マシン実行環境が選択される。

これらの仮想マシン実行環境が最短実行時間を達成する仮想マシン実行環境となるかどうかを確認するために種々の仮想マシン数、物理コア数、物理マシン数を組み合わせた仮想マシン環境において実行時間を測定した結果を図 25、図 26 に示す。図 25、図 26 の横軸は「実行環境番号__プロセス数」を示し、縦軸は各ベンチマークプログラムの最長実行時間を 1 とした場合の相対値を示している。

表 14：仮想マシン実行環境対応表

実行環境番号	実行に使用した4～16個の物コア
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	

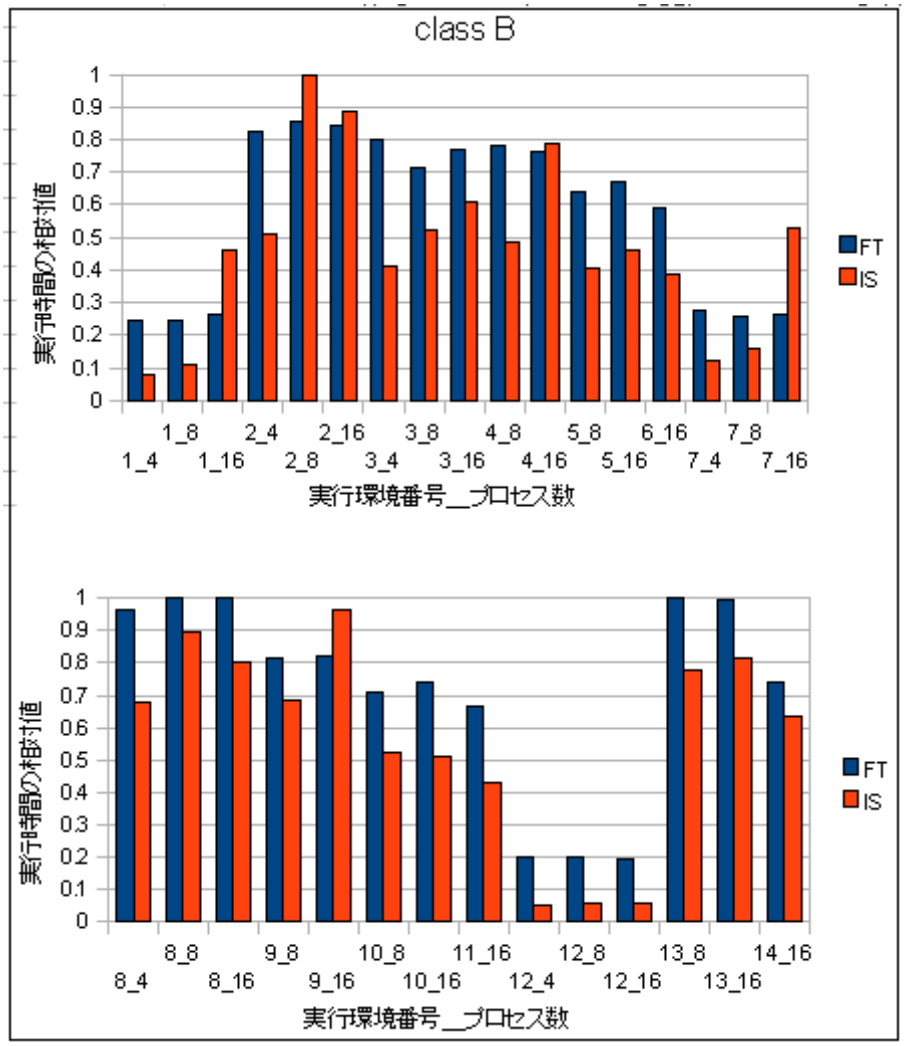


図 25：FT、IS の種々の仮想マシン実行環境における実行時間の相対値(class B)

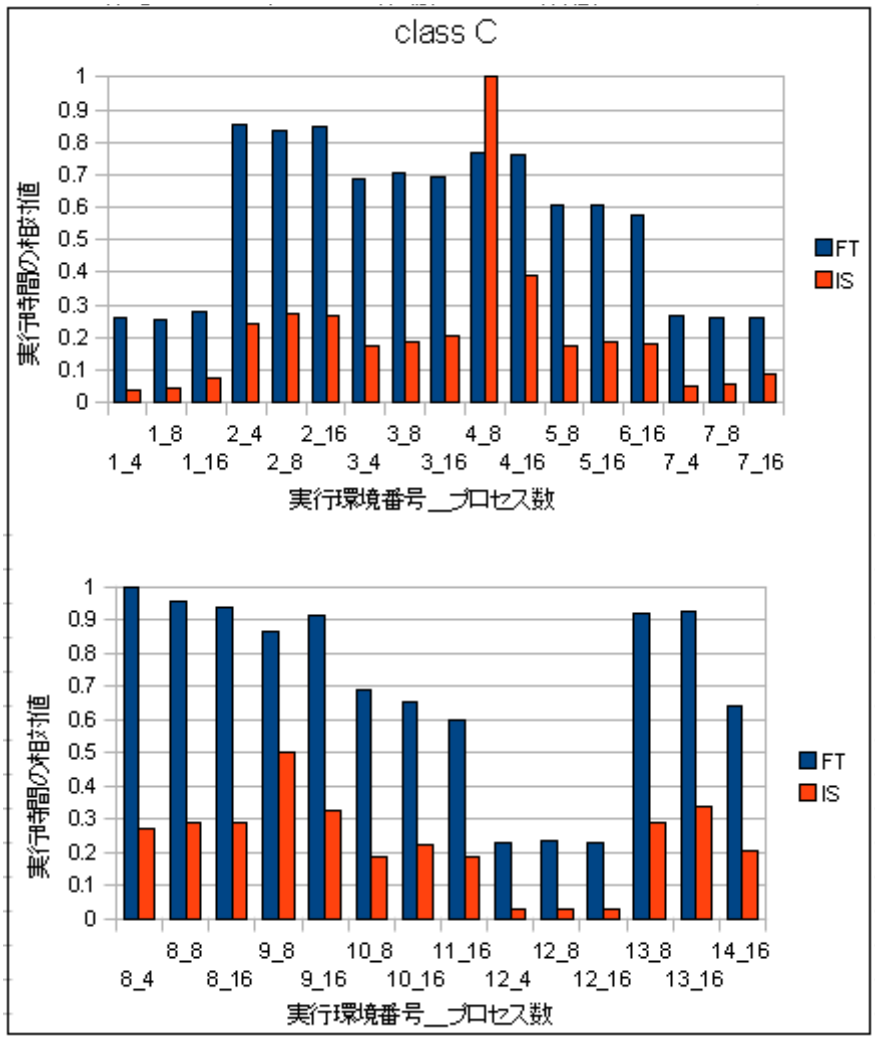


図 26：FT、IS の種々の仮想マシン実行環境における実行時間の相対値(class C)

図 25、図 26 から FT と IS において class B、class C 共に最短実行時間を達成する仮想マシン実行環境を選択することができた。また、参考までに予備実験対象とした 6 つのベンチマークプログラムについても本手法を適応させた結果を表 15 に示す。

表 15：各ベンチマークプログラムの仮想マシン実行環境

MPIプログラム	問題サイズ	CPU使用率	物理マシン数
BT	class B	0.45	1
	class C	0.62	1
CG	class B	0.20	1
	class C	0.32	1
LU	class B	0.51	2
	class C	0.76	4
MG	class B	0.35	1
	class C	0.43	1
SP	class B	0.35	1
	class C	0.55	1
EP	class B	1.00	4
	class C	1.00	4

以上の結果から、NPB では本研究の提案手法の有効性を確認することができたと判断した。

4.3 今後の課題

本研究では CPU 使用率を把握することで、最短実行時間を達成する仮想マシン実行環境が選択できることを明らかにした。

しかし物理マシンを 1 つにすべきか使用できる最大物理マシン数にすべきか選択する際に判断基準となる CPU 使用率の値は実験環境ごとに異なるため、実験環境ごとに判断基準となる CPU 使用率の値を調査する必要がある。そのためには本研究のようにいくつかの種類の MPI プログラムを実行させ、実行時間を比較することから実行に使用する物理マシン数を決定する際に判断基準となる CPU 使用率の値を求める方法も考えられるが、以下では 1 つの MPI プログラムを用いて実行に使用する物理マシン数を決定する際に判断基準となる CPU 使用率の値を求める方法を示す。

プロセス間通信量を変化させることで CPU 使用率を自由に調節できる MPI プログラムを作成し以下の手順で物理マシンを 1 つにすべきか使用できる最大物理マシン数にすべきか選択する際に判断基準となる CPU 使用率の値を求められると考えている。

(1)1 つ物理マシンの場合と実験に使用できる物理マシン数すべての場合で実行し、実行時間を比較する。
(2)(1)を繰り返しながら MPI プログラムの CPU 使用率を段階的に上げていく。
(3)(1)の結果、各仮想マシン実行環境において実行時間が誤差の範囲と判断できる程度に実行時間が近い値となった場合 MPI プログラムのプロセス間通信量を固定する。
(4)2 つの物理マシンに物理マシン数と同数の仮想マシンを配置し、各仮想マシンには物理コアを 1 つ割り当てた実行環境において MPI プログラムを実行し、その場合に測定された CPU 使用率が物理マシンを 1 つにすべきか使用できる最大物理マシン数にすべきか選択する際に判断基準となる CPU 使用率の値となる。

本手法の対象とする物理マシン数は 4 の場合と限定したが、実験に使用する物理マシン数を増やした際に MPI プログラムの実行時間と仮想マシン実行環境の間の規則性に変化が生じるかどうか確認し、必要に応じて物理マシン数選択手法に変更を加えることが今後の課題となる。

5. おわりに

本研究では行ったこととして以下の 3 つが挙げられる。

●NPB のベンチマークプログラムを仮想マシン数、物理コア数、物理マシン数を組み合わせた異なる仮想マシン環境において実行し、仮想マシン実行環境と実行時間に規則性があるかどうか調査。
●各 MPI プログラムに応じて最短実行時間を達成する物理マシン数、仮想マシン数、仮想マシンに割り当てる物理コア数の組み合わせを選択する手法を提案する。
●提案手法の有効性を検証する。

これらの研究を通して、仮想マシン環境で MPI プログラムを実行する際に各 MPI プログラムごとに最短実行時間を達成する物理マシン数、仮想マシン数、仮想マシンに割り当てる物理コア数の組み合わせを選択する手法を提案し、実験、検証することができた。

今後の課題として以下の 3 点が挙げられる。

●通信量を自由に調節できる MPI プログラムを作成する。
●本研究の提案手法が実験環境の異なる場合においても最短実行時間を達成する仮想マシン実行環境を提供できることを確認する。
●物理マシン数を増やしていく場合に、最短実行時間を達成する実行環境にどのような変化が生じるかを調査する。

参考文献

[1]vmware :http://www.vmware.com/

[2]Xen :http://www.xen.org/

[3]NTT データ HP:http://www.nttdata.co.jp/cloud/index.html

[4]富士通 HP :http://pr.fujitsu.com/jp/news/2010/07/13.html

[5] IBM HP:http://www-06.ibm.com/jp/press/2010/11/0201.html

[6] Amazon HP:http://aws.amazon.com/jp/hpc-applications/

[7]John Rehr, Fernando Vila, Jeffrey Gardner, Lukas Svec, Micah Prange, "Scientific Computing in the Cloud," Computing in Science and Engineering, 13 Jan. 2010. IEEE computer Society Digital Library. IEEE Computer Society

[8]Watson P; Lord P; Gibson F; Periorellis P; Pitsilis G
Silva, F., Barreira, G. and Ribeiro, L., “Cloud Computing for e-Science with CARMEN,”Proceedings of the 2nd Iberian Grid Infrastructure (IBERGRID) Pagination: 3-14 , Portugal , 12-14 May 2008

[9]Amar, L.; Barak, A. Levy, E.;An On-line Algorithm for Fair-Share Node Allocations in a Cluster,;Okun, M.;Dept. of Comput. Sci., Hebrew Univ. of Jerusalem, Jerusalem : Cluster Computing and the Grid, 2007. CCGRID 2007. Seventh IEEE International Symposium on 14-17 May 2007 page83 - 91

[10]Hayashi, Miyamoto, Otani, Tanaka, Takefusa,Nakada, Kudoh, Nagatsu, Sameshima, and Okamoto: Managing and Controlling GMPLS Network Resources for Grid Application, Optical Fiber Communications Conference (OFC) 2006 (2006).

[11]広渕　推宏、横井　威、江原　忠志、谷村　勇輔、小川　宏高、中田　秀基、田中　良夫、関口　智 :“複数サイトにまたがる仮想クラスタの構築方法” 先進的計算基盤システムシンポジウム SACSIS2008 pp333-340, 2008

[12]長沼　翔、高橋 慧，斎藤秀雄、柴田 剛志，田浦 健次朗，近山 隆:“ネットワークトポロジを考慮した効率的なバンド幅推定方法”先進的計算基盤システムシンポジウム SACSIS2008 pp359-366,2008

[13]上田　清詩、本多　弘樹，弓場　敏嗣:“グローバルコンピューティングシステムにおけるネットワーク Gantt 図を用いたジョブスケジューリング手法の提案”情報処理学会研究報告. [ハイパフォーマンスコンピューティング] IPSJ SIG Notes 2000(93) pp.49-54 20001006

[14]Fortaleza, Ceara, Brazil:“Supporting self-organization for hybrid grid resource

scheduling”Symposium on Applied Computing Proceedings of the 2008 ACM symposium on Applied computing Pages 1981-1986 Year of Publication: 2008

[15]豊島 詩織, 原 明日香, 小口 正人：「仮想マシン PC クラスタにおける並列データ処理アプリケーション実行時のストレージアクセスに関する一検討」並列/分散/協調処理に関するサマー・ワークショップ (SwoPP2009)，電子情報通信学会技術研究報告 Vol.109，No.168，CPSY2009-11，pp.7-11，仙台，2009 年 8 月

[16] Resource Allocation using Virtual Clusters,Mark Stillwell1 David Schanzenbach Frederic Vivien Henri Casanova,Proceedings of the 2009 9th CCGrid,Pages 260-267,2009.

[17]Nas Parallel Benchmark :http://www.nas.nasa.gov/Resources/Software/npb.html

[18]netperf :http://www.netperf.org/netperf/

[19]stream :http://www.cs.virginia.edu/stream/

謝辞

本研究を進めるに当たり、ご指導及びご助言をいただいた高性能コンピューティング学講座の本多弘樹教授、近藤正章准教授、平澤将一助教に深く感謝致します。

また、日ごろから研究の助言をくださった穂園 智哉さんをはじめ、高性能コンピューティング学講座の学生のみなさんに心から感謝致します。本当にありがとうございました。

付録

表 16：実行環境対応表(class C,プロセス数＝2)

実行環境番号	実行に使用した2つの物理コアを配置
1	
2	
3	
4	
5	

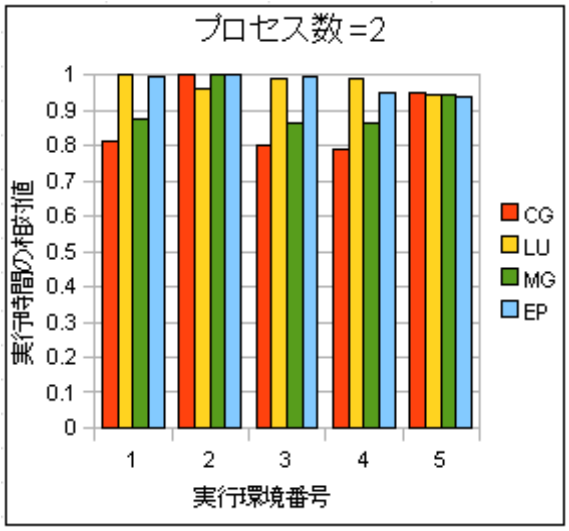


図 27：実行時間の相対値(class C プロセス数＝2)

表 17：実行環境対応表(class C,プロセス数＝4)

実行環境番号	実行に使用する4つの物理コア
1	●●●● ○○○○ ○○○○ ○○○○
2	●●○○ ●●○○ ○○○○ ○○○○
3	●○○○ ●○○○ ●○○○ ●○○○
4	●●●● ○○○○ ○○○○ ○○○○
5	●●○○ ●●○○ ○○○○ ○○○○
6	●●●● ○○○○ ○○○○ ○○○○
7	●●●● ○○○○ ○○○○ ○○○○
8	●●○○ ●●○○ ○○○○ ○○○○
9	●○○○ ●○○○ ●○○○ ●○○○

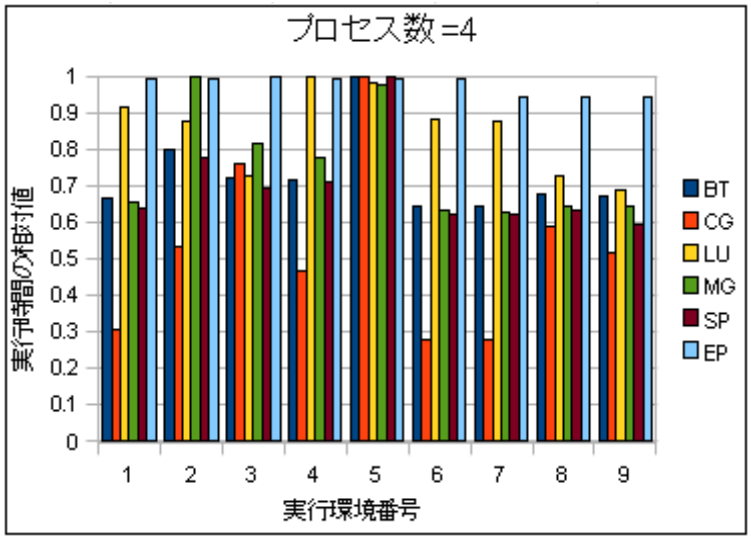


図 28：実行時間の相対値(class C,プロセス数＝4)

表 18：実行環境対応表(class C,プロセス数＝8)

実行環境番号	実行に使用する4～8つの物理コア
1	●●●● ○○○○ ○○○○ ○○○○
2	●●○○ ●●○○ ○○○○ ○○○○
3	●○○○ ●○○○ ●○○○ ●○○○
4	●●●● ●●●● ○○○○ ○○○○
5	●●○○ ●●○○ ●●○○ ●●○○
6	●●●● ○○○○ ○○○○ ○○○○
7	●●○○ ●●○○ ○○○○ ○○○○
8	●●●● ●●●● ○○○○ ○○○○
9	●●○○ ●●○○ ●●○○ ●●○○
10	●●●● ○○○○ ○○○○ ○○○○
11	●●●● ●●●● ○○○○ ○○○○
12	●●●● ○○○○ ○○○○ ○○○○
13	●●●● ●●●● ○○○○ ○○○○
14	●●○○ ●●○○ ●●○○ ●●○○

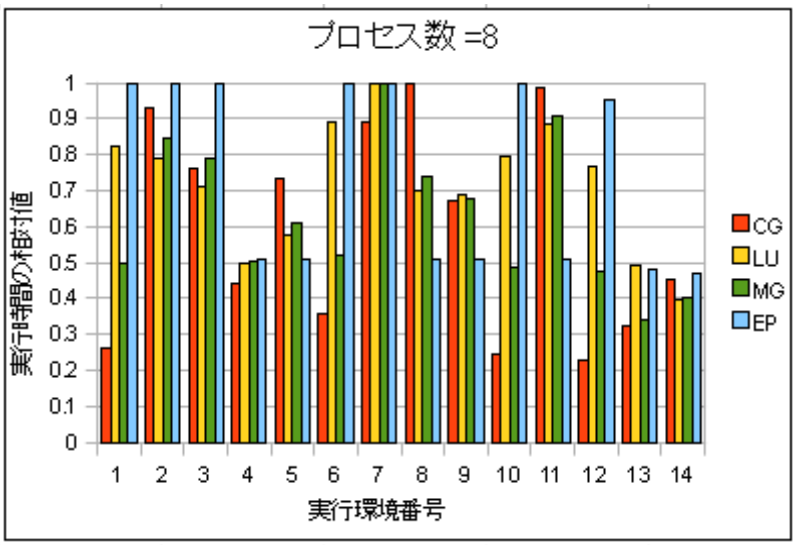
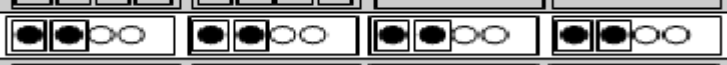
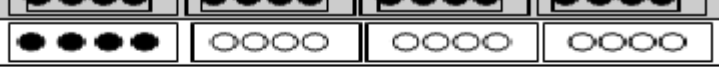


図 29：実行時間の相対値(class C,プロセス数＝8)

表 19：実行環境対応表(class C,プロセス数＝16~128)

実行環境番号	実行に使用した4～16個の物理コア
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	

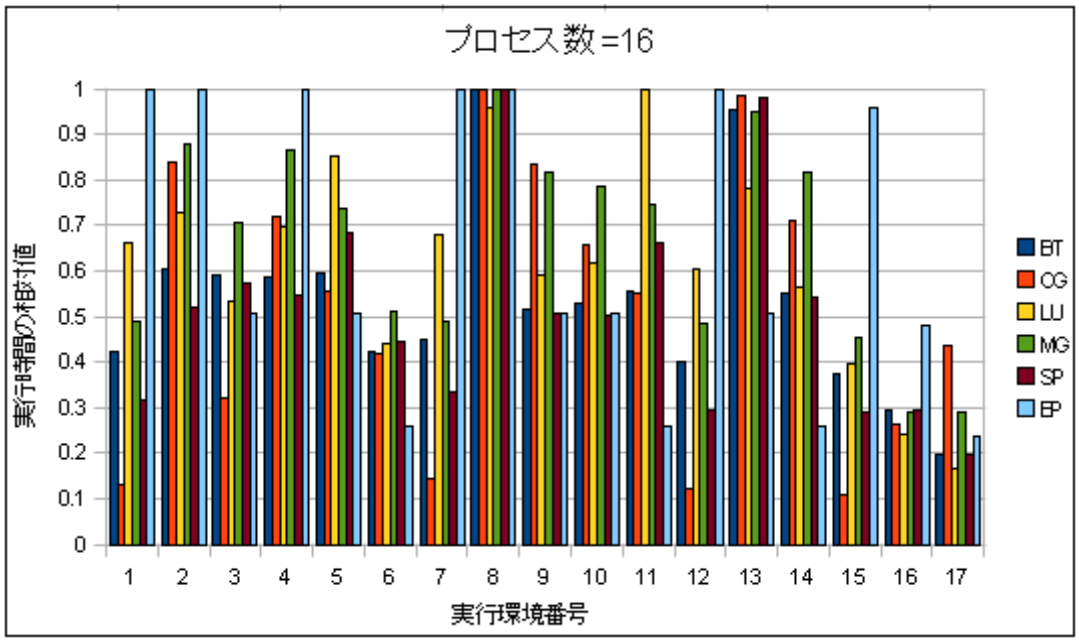


図 30：実行時間の相対値(class C,プロセス数＝16)

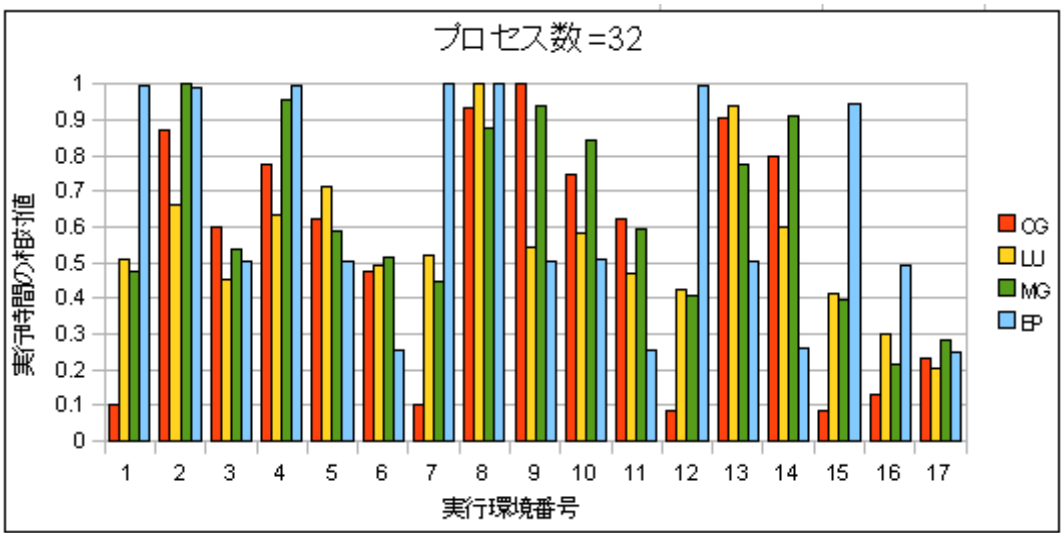


図 31：実行時間の相対値(class C,プロセス数＝32)

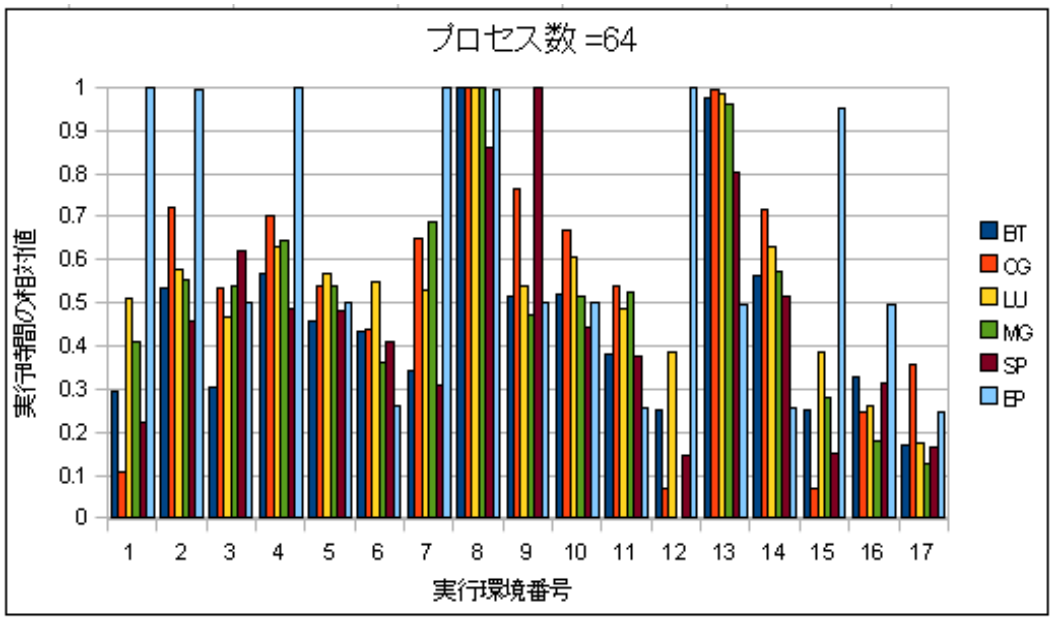


図 32：実行時間の相対値(class C,プロセス数＝64)

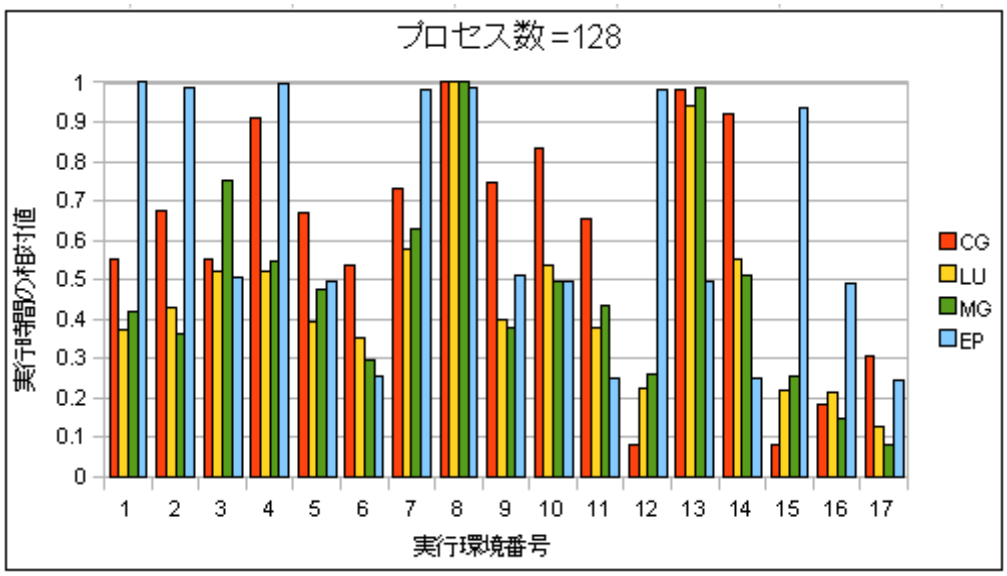


図 33：実行時間の相対値(class C,プロセス数＝128)

表 20：実行環境対応表(class C,物理コア数＝4)

実行環境番号	実行に使用した4つの物理コア
1	●●●● ○○○○ ○○○○ ○○○○
2	●●○○ ●●○○ ○○○○ ○○○○
3	●○○○ ●○○○ ●○○○ ●○○○
4	●●●● ○○○○ ○○○○ ○○○○
5	●●○○ ●●○○ ○○○○ ○○○○
6	●●●● ○○○○ ○○○○ ○○○○
7	●●●● ○○○○ ○○○○ ○○○○
8	●●○○ ●●○○ ○○○○ ○○○○
9	●○○○ ●○○○ ●○○○ ●○○○

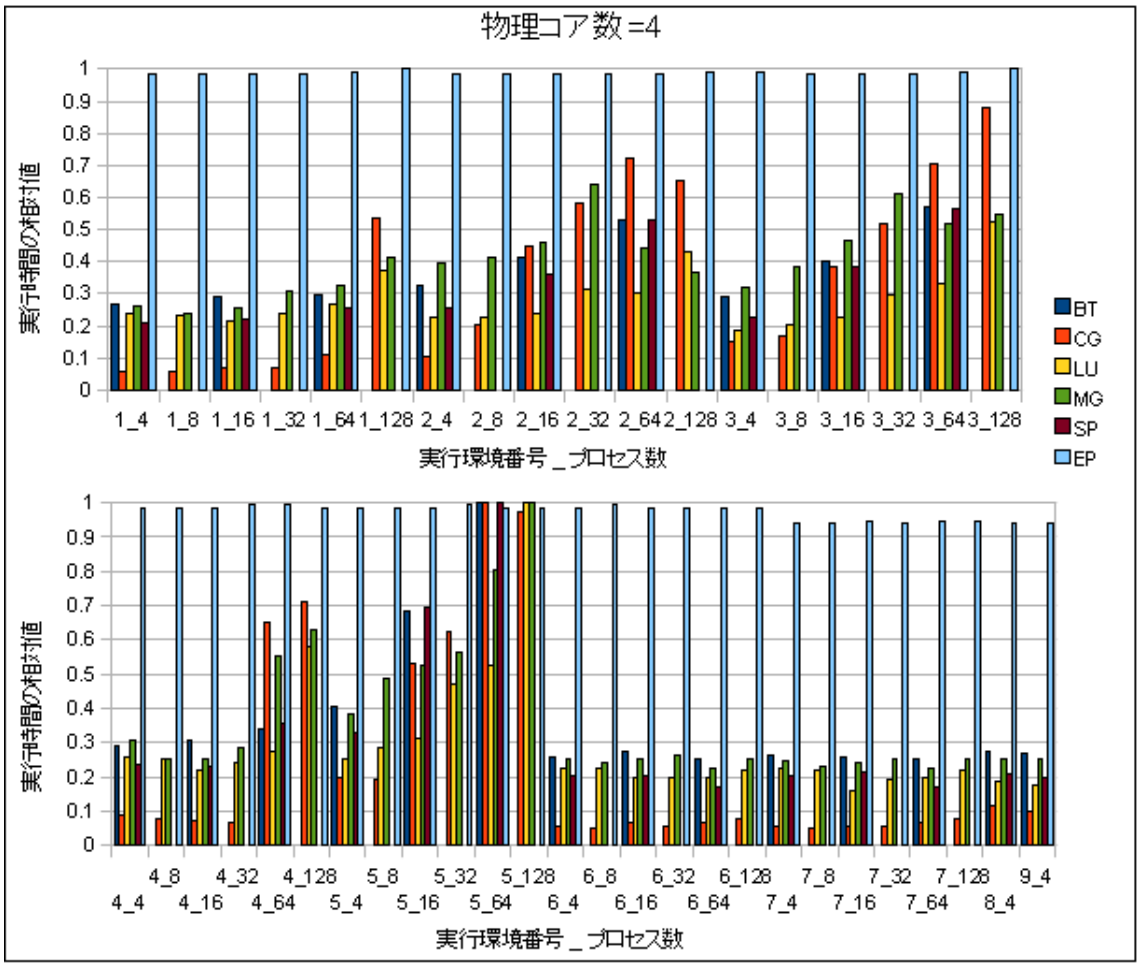


図 34：実行時間の相対値(class C,物理コア数＝4)

表 21：実行環境対応表(class C,物理コア数＝8)

実行環境番号	実行に使用した8つの物理コア
1	●●●● ●●●● ○○○○ ○○○○
2	●●○○ ●●○○ ●●○○ ●●○○
3	●●●● ●●●● ○○○○ ○○○○
4	●●○○ ●●○○ ●●○○ ●●○○
5	●●●● ●●●● ○○○○ ○○○○
6	●●●● ●●●● ○○○○ ○○○○
7	●●○○ ●●○○ ●●○○ ●●○○

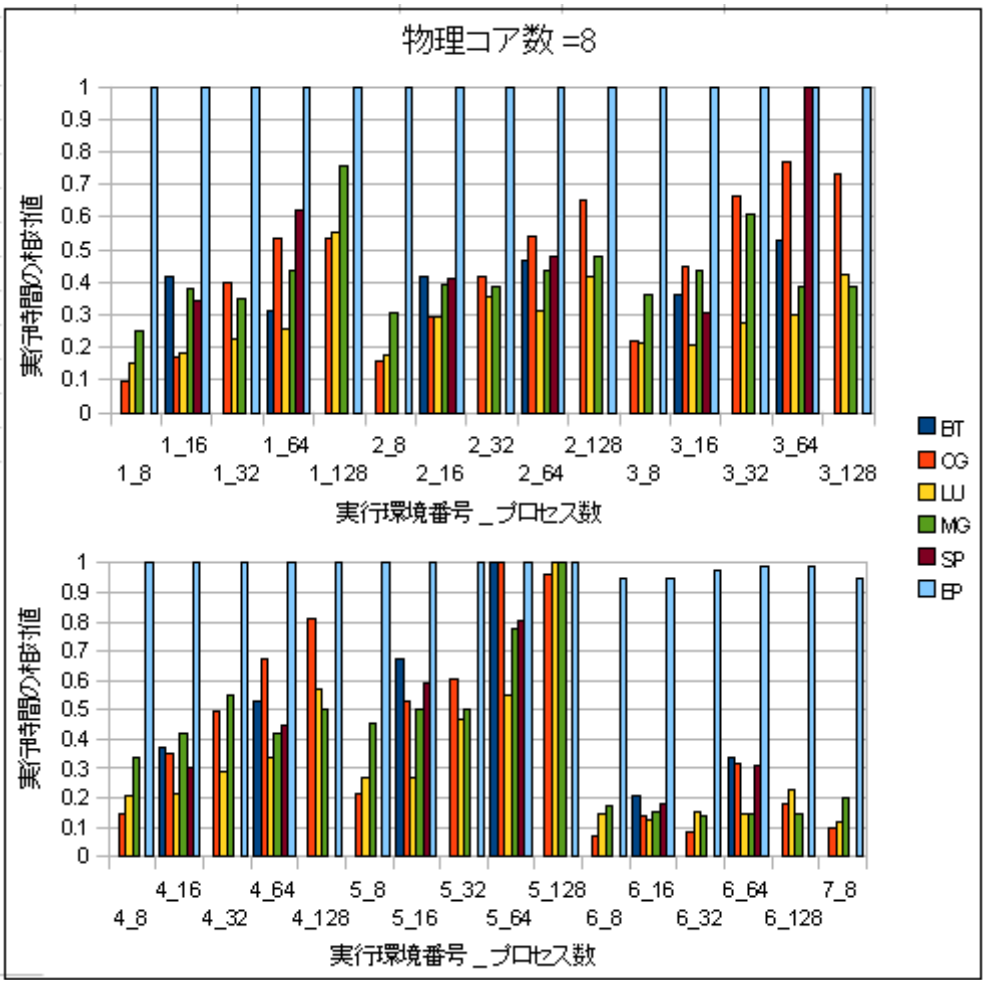


図 35：実行時間の相対値(class C,物理コア数＝8)

表 22：実行環境対応表(class C,物理コア数＝16)

実行環境番号	実行に使用した 16個の物理コア
1	
2	
3	
4	

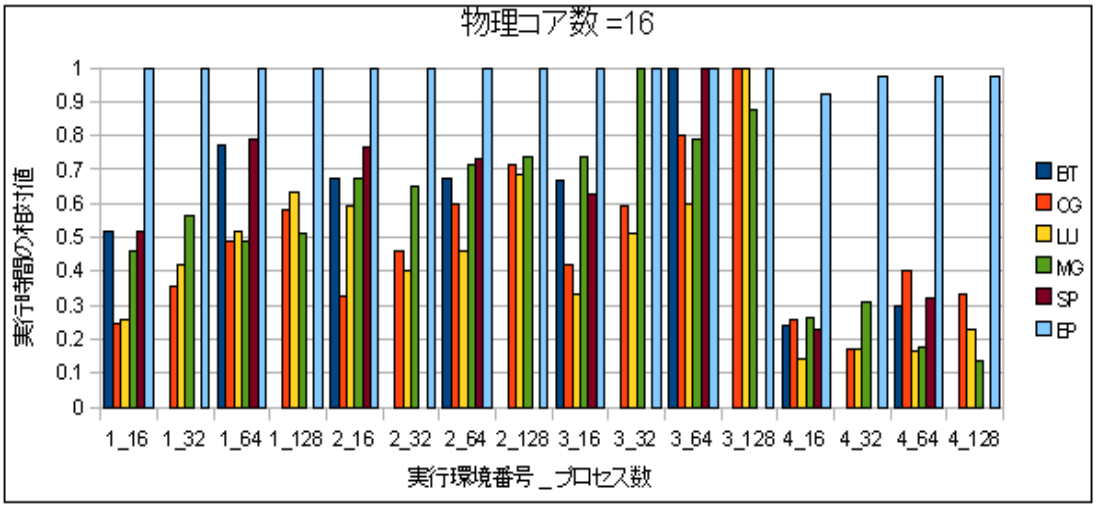


図 36：実行時間の相対値(class C,物理コア数＝16)