社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

λコンピューティング環境構築のための Globus Toolkitを用いたMPIライブラリの実装と評価

井本　舞 †　谷口　英二†　馬場　健一††　村田　正幸†

† 大阪大学 大学院情報科学研究科　〒 565–0871 大阪府吹田市山田丘 1–5
†† 大阪大学 サイバーメディアセンター　〒 567–0047 大阪府茨木市美穂ヶ丘 5–1
E-mail: †{m-imoto,e-tanigu,murata}@ist.osaka-u.ac.jp, ††baba@cmc.osaka-u.ac.jp

あらまし　近年，グリッド計算に関する研究開発が盛んに行われている．現状のグリッド計算環境ではノード計算機間の通信に TCP/IP を用いているが、パケットを単位としたデータ交換ではオーバーヘッドの影響が大きく，大規模計算で必要な大量データの共有や交換を行うには十分な性能を得ることは難しい．そこで各ノード計算機に光ファイバを直結し，波長パスをノード計算機間の高速な通信チャネルとして活用するλコンピューティング環境を提案する．波長パスを利用することにより，ユーザに対して高速高信頼な通信パイプを提供可能になり，波長パスを用いたデータ交換，共有が可能になる．本稿では，グリッド計算を高速に行うλコンピューティング環境の実現形態として，WDM 技術に基づく AWG–STAR システムを用いてグリッド環境を構築した．すなわち，Globus Toolkit を導入できるように，MPI ライブラリである MPICH–G2 を AWG–STAR システム上で動作可能とした．そのために，AWG–STAR の共有メモリシステムを利用できるメッセージパッシング手法を提案し，実装している．また，実装したライブラリ上で MPI アプリケーションを実行し，分散計算の性能を評価した．その結果，AWG–STAR を用いた共有メモリ上のデータ交換の性能は，共有メモリへのアクセス回数，データサイズに大きく影響されることが明らかになった．これは，グリッド計算をより高速に実行するλコンピューティング環境の設計に指針を与えるものである．

キーワード　λコンピューティング環境，分散計算，AWG–STAR，Globus Toolkit，共有メモリアクセス手法

Implementation and Evaluation of MPI Library with Globus Toolkit for Establishing λ Conputing Environment

Mai IMOTO†, Eiji TANIGUCHI†, Ken-ichi BABA††, and Masayuki MURATA†

† Graduate School of Information Science and Technology, Osaka University
1–5 Yamadaoka, Suita, Osaka 565–0871, Japan
†† Cybermedia Center, Osaka University　5–1 Mihogaoka, Ibaraki, Suita, Osaka 567–0047 Japan
E-mail: †{m-imoto,e-tanigu,murata}@ist.osaka-u.ac.jp, ††baba@cmc.osaka-u.ac.jp

Abstract　In recent years, the Grid technology has being studied and developed by many researchers. In the conventional Grid environment, data is exchanged by using TCP/IP. However, as long as the architecture is based on packet switching, realization of high performance computing is difficult. Thus we propose a new architecture, the λ computing environment, that accomplishes high speed data transmission over optical fibers. In the λ computing environment, we can achieve highly reliable high-speed communication by establishing wavelength paths. Moreover, by making virtual channels, distributed computation and data sharing are enabled. In this paper, we use the Globus Toolkit to build a Grid environment and establish the λ computing environment using the AWG–STAR. Moreover, we implement the MPI library using services of Globus Toolkit and shared memory of AWG-STAR to exchange data, and evaluate performance of distributed computation in a λ computing environment. Our results show that the performance depends on the number of accesses to the shared memory and the size of exchanged data.

Key words　λ computing environment, distributed computing, AWG–STAR, Globus Toolkit, access to shared memory

1. はじめに

近年，グリッドコンピューティング環境の構築に対する期待が高まり，活発に研究開発が行われている．グリッドコンピューティングは，広域に分散した計算機やストレージ，さらには観測機器，さまざまなデバイスなど，多くの資源を，ネットワークを利用して統合的に接続し，ひとつの大規模な仮想計算機として機能させるインフラストラクチャを構成する技術である．このグリッドコンピューティングによって構築された仮想計算機を用いることにより，単体の計算機では解くことが難しい大規模な科学技術計算を行う，高性能観測機器から発生する大量のデータを高速に処理する，大規模データをリアルタイムに計算しながら可視化を行う，などより高度な処理能力を得ることができると期待されている．グリッドコンピューティングを実現するためには，地理的，組織的に分散したさまざまなコンピューティングに関する資源を動的に共有し，協調動作させることが重要な課題となっており，そのためのミドルウェアが必要である．. 現在では Globus Alliance によって開発された，グリッドコンピューティングミドルウェアの Globus Toolkit がデファクト標準となっている [1].

一方，グリッドコンピューティングを実現するために重要となる下位層の高速ネットワーク技術として，現在光伝送技術を用いた研究が活発に進められている．特に高速ネットワークを実現する手段として光の波長を用いて多重化を行う WDM (Wavelength Division Multiplexing) 技術が研究の中心であり，1000 波を利用できる新たな WDM 技術の研究も進められている [2]．また WDM を利用してインターネットの高速化を実現する IP over WDM ネットワークの研究もさかんに行われている．しかしながら，現状のネットワークにおいてはルーティングを行う際に，光信号を電気信号に変換し，もう一度光信号にかえる処理を行っており，光の高速性を損ねてしまう．そのため，WDM 技術以外のさまざまな光技術を下位レイヤの通信技術とする GMPLS (Generalized Multi-Protocol Label Switching) [3] と呼ばれるインターネットルーティング技術や，フォトニックネットワークの真の IP 化を実現するために光領域でパケット交換を行うフォトニックパケットスイッチの研究も行われている [4]〜[8] ．しかしながら，これらの技術は情報を扱う細粒度として IP パケットを扱い，ネットワーク上でそのパケットをいかに高速に運ぶかを研究開発の目標としている．そのため，このようなパケット交換技術に基づいたアーキテクチャをとる限り，個々のコネクションに対する高品質通信の実現は困難である．

そこで我々の研究グループでは、各計算機を接続している光ファイバを専用の通信路として利用し，WDM 技術を用いた高速な通信チャネルとして活用する λ コンピューティング環境を提案している [9], [10]．λ コンピューティング環境上の通信は，TCP/IP ではなく波長パスを利用するため，高速高信頼な通信を実現することができる（図 1）．

本稿では，λ コンピューティング環境に Globus Toolkit を導入することによって高速な分散計算環境を提供することを目指す．すなわち，Globus Toolkit により構築されたグリッド計算環境の下位層に λ コンピューティング環境におけるフォトニック技術を利用することにより，分散計算を行いたいユーザは，従来の利用法を変えることなく高速なコンピューティング環境を利用することが可能となる．具体的には，ユーザは，Globus Toolkit を利用して MPI (Message Passing Interface) を用いた分散計算アプリケーションのジョブを投入し，分散計算が λ コンピューティング環境上で実行される．また本稿では，λ コンピューティング環境を構築するために，各ノード計算機上に存在する共有メモリを高速にアクセスする手法を実装する．実際には，日本電信電話株式会社フォトニクス研究所が開発している「情報共有ネットワークシステム（AWG–STAR）」を用いる [11]〜[13]．この AWG–STAR システムは，各ノード計算機が波長可変光源を通じて光ファイバにより AWG (Arrayed Waveguide Grating) と呼ばれるルータに接続され，物理的にはスタートポロジを，論理的にはリングトポロジを形成している．また，各ノード計算機は共有メモリボードを搭載しており，共有メモリボード上のデータは全ノード計算機で同一のものになるよう設計されている．

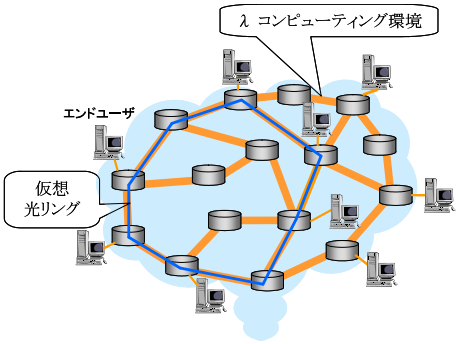

図 1　λ コンピューティング環境

本稿では，AWG–STAR システムを用いた λ コンピューティング環境を構築し，Globus Toolkit によるミドルウェアを実装した上で，MPI による分散計算アプリケーションを動作させることにより λ コンピューティング環境における分散計算性能を明らかにする．

以下，2 章では本研究で用いた AWG–STAR システムについて説明する．3 章では AWG–STAR システムにおける MPI ライブラリの実装方法について述べ，4 章では構築した λ コンピューティング環境において分散計算を行った場合の性能を評価する．最後に 5 章で本報告についてのまとめと今後の課題について述べる．

2. AWG–STAR システム

本報告では，λ コンピューティング環境構築のためのひとつの手法として，AWG–STAR システムを利用する．本章では AWG–STAR システムについて説明する．

2.1 AWG–STAR システムの概要

AWG–STAR システムは，日本電信電話株式会社フォトニクス研究所により開発されたシステムであり，WDM 技術によ

るデータ転送と AWG ルータによる波長ルーティング技術によって実現された情報共有ネットワークシステムである [11]〜[13] ．AWG ルータは波長に基づいたルーティングを行っており，電気信号に変換せず光信号をそのまま処理するため，高速なネットワークを構築することができる．また，各ノード計算機は，共有メモリボードを搭載し，共有メモリボード上で同一のデータを保持することでメモリを共有しており，高速な光リングネットワークを利用してデータ交換をリアルタイムに行うことができる．従来のシステムでは，データ共有するためには何らかの明示的なデータ転送が必要であったが，AWG–STAR システムでは共有メモリに書き込まれたデータは光リングネットワークに送出され，全ノード計算機の共有メモリが自動的に更新される．AWG–STAR システムを用いることにより，共有メモリ上のデータ共有は，共有メモリに書き込む処理によりハードウェアがバックグラウンドで行うため，高速に実行される．他ノード計算機が更新したデータの取得は，AWG–STAR システムを通じて共有メモリに配信され，自動的に更新されるため，共有メモリから読み込むことにより実現できる．

2.2 AWG–STAR における遅延時間

各ノード計算機の共有メモリを利用するには，ローカルメモリにアクセスする以上に遅延時間を要する．その要因は PCI バスを経由する遅延時間とデータ共有を行うための遅延時間である．すなわち，ローカルメモリのデータを共有メモリボードに転送する遅延時間と，データを全ノード計算機が共有するために，少なくともデータが光リングネットワークを一周回する時間である．表 1 に PCI バスを経由したデータアクセス速度を示す．

2.3 λ コンピューティング環境への適用

本稿では，λ コンピューティング環境の構築に AWG–STAR システムを用い，分散計算を行うことを考える．AWG–STAR システムの利用方法により，2 つのモデルが考えられる．ひとつは，共有メモリ上に計算対象となるデータを載せ，各ノード計算機が共有メモリとの間でデータの読み込み，書き込みを行うモデルである．この場合，共有メモリは全ノード計算機間で共有されるため，データ更新時にデータが光リングネットワークを 1 周するのを待つ必要があり，そのための遅延が生じる．もうひとつは，共有メモリに計算対象となるデータを載せるのではなく，主に AWG–STAR システムを高速チャネルとして利用するモデルである．すなわち，各ノード計算機のローカルメモリにデータを載せ，計算を行い，情報交換が必要な場合にのみデータを共有メモリ上に書き込み，他ノード計算機に高速に伝送するものである．本稿では，MPI ライブラリを用いた分散計算を対象にしており，親和性が高く，また共有メモリアクセスにおける遅延時間を軽減できるため，後者のモデルを対象とする．

3. 共有メモリを用いたメッセージパッシング手法の提案

3.1 分散計算のための MPI ライブラリ MPICH–G2

MPI (Message-Passing Interface) は，分散計算を行う際に

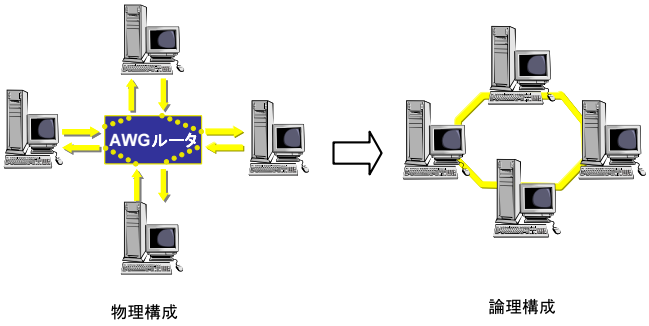

図 2　光リングネットワークの構成

表 1　共有メモリボードの仕様

光インターフェースの伝送速度	2.152Gbps
ノード計算機の 1 回あたりの転送データ量	1KByte
ノード計算機でのフレーム転送処理遅延	500ns
共有メモリへの書き込み	64MBytes/s
共有メモリから読みだし	80MBytes/s

ネットワークを通じてメッセージ交換を行い，計算に必要なデータを共有，あるいは他のプロセスとの協調動作を行うための規定である．MPI は API の仕様を定めただけであるため，実際の処理をどのように行うかは実装によって異なり，その実装のひとつに Globus Toolkit 上で動作する MPI ライブラリ，MPICH–G2 [14] がある．MPICH–G2 は通信とジョブ管理に Globus Toolkit が提供する API を用いる．本稿では MPICH–G2 の実装方法を参考にして，ジョブ管理は Globus Toolkit を利用し，メッセージパッシングのための通信は AWG–STAR を利用する MPI ライブラリを実装することにより，λ コンピューティング環境で MPI アプリケーションを実行できるようにした．以下，MPI アプリケーションにおいて実行される MPI プロセスを単にプロセスと呼び，データを送信するプロセスを送信プロセス，データを受信するプロセスを受信プロセスと呼ぶ．

3.2 AWG–STAR システムを利用した MPI ライブラリの実装方法の提案

AWG–STAR システムを用いてメッセージパッシングを実現するため，各プロセスが自プロセスも含めた全てのプロセスへデータを送受信するためのキューを共有メモリ上に確保する．具体的には，プロセス数を n とすると，共有メモリの領域を n^2 に分割し，各送受信プロセス間ごとの送受信キュー用領域に割り当てる．つまり，プロセスランク i からプロセスランク j へデータを送信する場合，共有メモリの $[i, j]$ 領域へデータを書き込む．送信プロセスでは送信関数が呼び出されると共有メモリ上の送信先プロセス用領域に送信データを書き込む．共有メモリ上にデータを書き込む際，前のデータが書き込まれているアドレスの次から書き込む必要があるため，前のデータが書き込まれている最後尾のアドレスを送信プロセスのローカルメモリに保持している．一方，受信プロセスでは共有メモリに新しい書き込みがあると受信データを共有メモリからローカルメモリに読み込む．このとき，受信プロセスは送信プロセスが共有メモリにデータを書き込んだことを知る必要があるため，AWG–STAR システムが提供するシグナル機能を用い

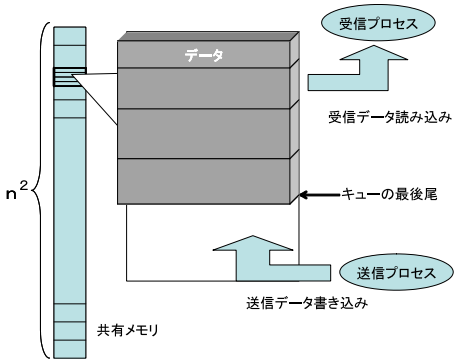

図 3　提案手法における共有メモリの利用方法

る．つまり，送信プロセスがデータを共有メモリに書き込んだ後，送信プロセスは受信プロセスに対してシグナルを発する．シグナルを受けたプロセスは，シグナルを発したプロセスからのデータを共有メモリから読み込む．図 3 に共有メモリの利用方法を示す．送受信キューを共有メモリ上に実装することにより，MPICH–G2 ではローカルメモリとネットワーク上の通信ソケットに分かれていたデータ送信の作業を，ひとつの作業としてとらえることができる．

また，各プロセスのローカルメモリには，MPICH–G2 の実装方法を参考に、posted キューと unexpected キューと呼ばれる 2 種類の受信用キューを設ける．これは共有メモリからデータを受信するタイミングと，プロセス上で受信関数が呼ばれるタイミングが前後するためである．つまり，受信関数が呼ばれたにも関わらず受信データが共有メモリに書き込まれていない場合，データを受信するまで受信要求をバッファリングしておく必要がある．データを受信するまで受信要求をバッファリングするためのキューが posted キューである．逆に，共有メモリからデータを読み込んだにも関わらずそのデータを受信するべき受信関数が呼ばれていない場合，受信関数が呼ばれるまで受信データをバッファリングしておく必要がある．受信関数が呼ばれるまで受信データをバッファリングするためのキューが unexpected キューである．受信データと受信要求は，データに付加されたヘッダの情報を元にして要求に一致するデータであるかどうかを判定する．

3.3　MPICH–G2 と提案方式の比較

MPICH–G2 による MPI アプリケーションの実行方法と，AWG–STAR を用いた提案方式による MPI アプリケーションの実行方法を図 4 に示す．プロセスを動かすジョブ実行ノードに対する認証とジョブ実行要求は Globus Toolkit の API を用いて行う．MPI アプリケーションを動作させメッセージパッシングを行う際は，MPICH–G2 を用いた場合は，ローカルメモリから OS のシステムコールを通じてソケットバッファにコピーされ，次にソケットバッファから NIC のメモリ上に転送され，NIC 内で MAC フレームを生成し，イーサネットを通じて転送される．AWG–STAR を用いた場合は，ローカルメモリから共有メモリボードに転送，バッファされ，ネットワークに送出される．AWG–STAR を用いた構成では，メモリコピーとパケット生成にかかるオーバーヘッドが少なく，高速なデータ交換が可能であると考えられる．

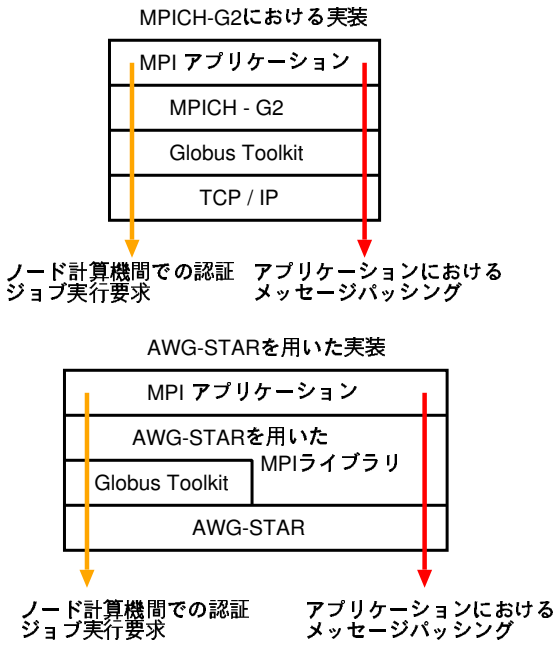

図 4　MPICH–G2 と提案方式の比較

4.　MPI アプリケーションによる評価

本章では，並列アプリケーションのベンチマークである姫野ベンチマーク [15] を MPI アプリケーションとして動作させ，処理速度を測定することにより，AWG–STAR を用いて構成した共有メモリシステムとそのメモリアクセス手法の性能を評価する．ただし，今回の実装ではそれぞれのノード計算機上ではひとつの MPI プロセスを動作させることを前提としている．これは，AWG–STAR システムでは，共有メモリボードを複数のアプリケーションやプロセスで共有することを想定していないためである．また，MPICH–G2 上でも同様のアプリケーションを走らせて比較対象としている．

4.1　実験システム環境

評価に用いた計算機の仕様を表 2 に示す．実験に使用した計算機の台数は 1 台から 4 台の範囲で行い，全て同じ性能の計算機を用いた．今回の実験では，ノード計算機数に応じて光リングの長さを変えている．具体的には，ノード計算機数を N とすると，光リングネットワークの長さは $10N$m としている．MPICH–G2 を利用する際はネットワークとして 100Mbps の Ethernet を用いている．

4.2　評価に用いるアプリケーションプログラム

姫野ベンチマークは，理化学研究所の姫野龍太郎氏が非圧縮流体解析コードの性能評価のために考案したベンチマークで，ポアソン方程式をヤコビの反復法で解く場合に主要なループの処理速度を計るものである．ヤコビ法は領域分割法を用いることで，三次元配列の領域をプロセス数で等分割し，並列化を行うことが可能である．今回ベンチマークとして走らせた問題サイズはあらかじめ姫野ベンチマークで設定されている問題サイズを変更している．

表 2　実験に用いた計算機の仕様

CPU	Xeon 2.80 GHz
メインメモリ	SDRAM 512MB
1 次キャッシュ	512KB
2 次キャッシュ	512KB
NIC	Intel PRO/100MT
PCI バス	64 bit/66MHz
PCI 転送速度	533MBytes/sec
OS	Redhat Linux 7.3
コンパイラ	gcc 2.96
グリッド環境ミドルウェア	Globus Toolkit 2.4

4.3　MPICH–G2 との比較による評価

AWG–STAR システムを用いたシステムで実装した MPI ライブラリでの処理速度と，Ethernet を使用している MPICH–G2 での処理速度の比較を行った．図 5 に 2 プロセスでアプリケーションを実行したときの処理速度を，図 6 に 4 プロセスでアプリケーションを実行したときの処理速度を示す．横軸には問題サイズ，縦軸には処理速度 (MFLOPS) をとっている．問題サイズの大きさと一度に交換するデータの大きさは比例関係にあり，問題サイズの大きさとデータの交換回数は反比例関係にある．

図 5，図 6 で示されているように，AWG–STAR を用いた MPI アプリケーションでは，問題サイズが小さいときは処理速度が遅い．これは，問題サイズが小さいときは共有メモリへのアクセスが頻繁に発生するため，共有メモリボードを利用するアクセス遅延が大きなボトルネックになっていることが要因と考えられる．問題サイズが大きくなり，共有メモリへのアクセス回数が減少すると，共有メモリボードのアクセス遅延が減少するため，処理速度が速くなる．しかしながら，依然として MPICH–G2 に比べて処理速度が遅く，共有メモリボードのアクセス速度の改善が必要と考えられる．

4.4　プロセス数による評価

同じ大きさの問題サイズを，異なるプロセス数で実行したときの処理速度を図 7 に示す．1 プロセスと 2 プロセスでの処理速度を比べた場合，問題サイズが $129 \times 129 \times 257$ より小さいときは，2 プロセスで実行したときよりも 1 プロセスで実行したときの方が処理速度が速い．これは，1 プロセスで実行する際はメッセージパッシングを行わないため，メッセージパッシングのオーバーヘッドがないためである．しかし，問題サイズが大きくなるにつれて 1 プロセスと 2 プロセスの処理速度の差は小さくなり，問題サイズが $161 \times 161 \times 321$ のときは 2 プロセスで実行した方が処理速度が速くなる．この理由は，問題サイズが大きくなるに従って，データ交換の回数が減少し，メッセージパッシングのオーバーヘッドが少なくなるからである．

また，2 プロセスと 4 プロセスでの処理速度を比べた場合，2 プロセスで実行した方が処理速度が速い．これは，1 プロセスと 2 プロセスを比較した場合と同様に，2 プロセスで実行した方がメッセージパッシングの回数が少ないからである．ただし，問題サイズが $193 \times 193 \times 385$ 以上のとき，問題サイズが大き過ぎるために 1 プロセスでは実行不可能であった．同様に，2

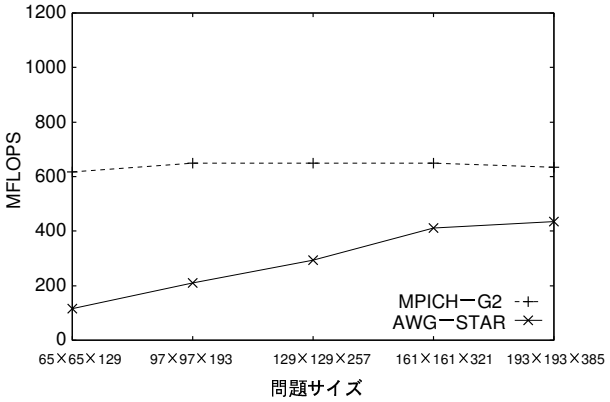

図 5　2 プロセスで動作させたときの処理速度

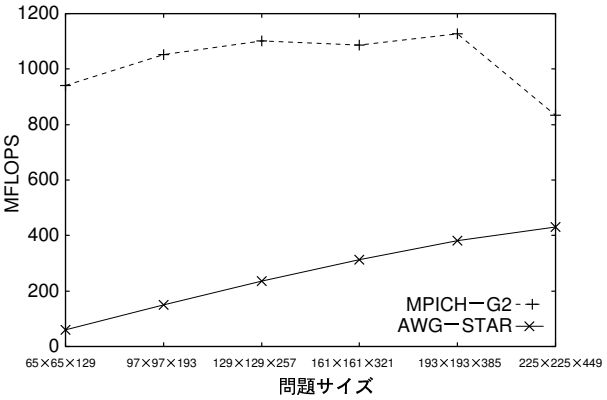

図 6　4 プロセスで動作させたときの処理速度

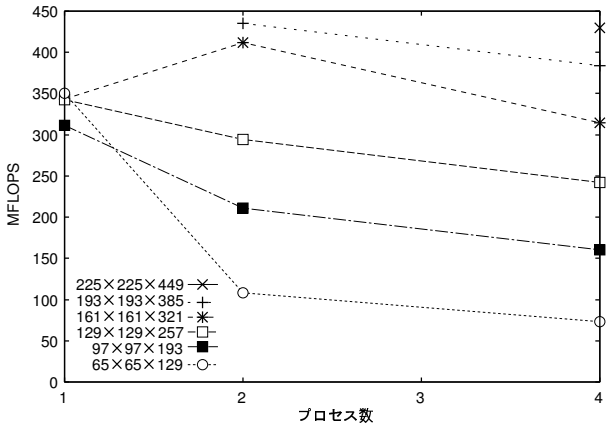

図 7　プロセス数の違いによる処理速度の変化

プロセスで問題サイズ $225 \times 225 \times 449$ を実行することもできなかった．すなわち，1 つのプロセスでは計算できないような大きな問題サイズの場合も，複数の計算機を利用し分散計算することにより結果を得ることができることがわかる．

4.5　問題分割の違いによる評価

姫野ベンチマークでは，3 次元配列をプロセス数で等分割し，プロセス間でデータ交換を行うことで並列計算を実行する．図 8 に示すように，交換をするデータサイズは，おおよそ問題分割により境界を共有している面積の大きさに比例する．すなわち，問題分割が $1 \times 4 \times 1$ の場合は，問題分割が $1 \times 1 \times 4$ の場合の 2 倍の大きさのデータを交換する．問題分割が $1 \times 4 \times 1$

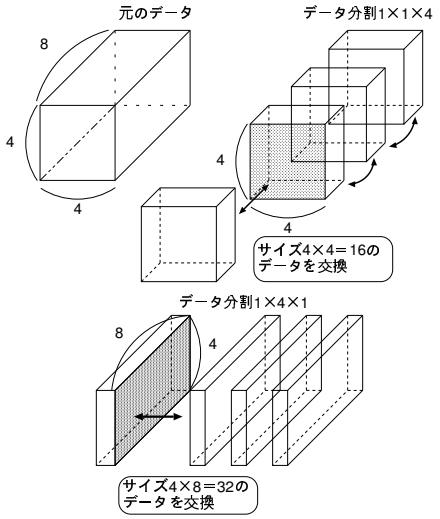

図 8　問題分割と交換データサイズ

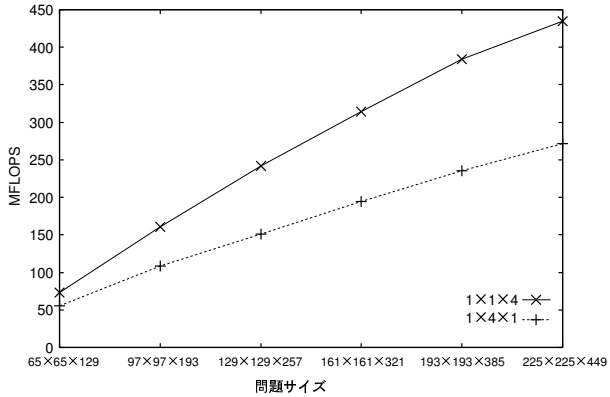

図 9　問題分割の違いによる処理速度変化

の場合と問題分割が $1 \times 1 \times 4$ の場合の処理速度を図 9 に示す．問題サイズ $1 \times 1 \times 4$ の処理速度が問題サイズ $1 \times 4 \times 1$ の処理速度の 1.3 倍から 1.6 倍になっており，交換するデータの大きさに処理速度が大きく影響していることがわかる．つまり，ローカルメモリと共有メモリ間の転送時間が処理速度に現れていることから，転送速度の遅さがボトルネックになっていることがわかる．

5. おわりに

本報告では，λ コンピューティング環境の実現形態のひとつとして WDM 技術に基づくフォトニックネットワークである AWG–STAR システムを用いてグリッド計算環境を構築し，グリッド計算環境でデファクト標準となっている Globus Toolkit をミドルウェアとして実装するため，分散計算を行うための MPI ライブラリの実装手法を提案した．

さらに，MPI アプリケーションを実行することにより，構築したシステムが動作することを確認し，構築システムにおける分散計算性能を評価した．その結果，AWG–STAR システムに基づく共有メモリシステムを λ コンピューティング環境として利用する場合，共有メモリへのアクセス回数と，共有メモリへ書き込むデータサイズが性能に影響を与えることが分かった．

今後の課題として，共有メモリへの動的なメモリの割り当てがあげられる．共有メモリの動的な割り当てが可能となれば，より柔軟で効率のよい MPI ライブラリの実装が可能となる．また，1 台のノード計算機上で複数の MPI プロセスや他の共有メモリを用いたアプリケーションを実行させた際に競合しない機能を持たせることも今後の課題となる．

謝辞　本研究を進めるにあたり，日本電信電話株式会社フォトニクス研究所の岡田顕氏，大阪大学大学院情報科学研究科の藤本典幸助教授に多大なご支援を頂いた．深く謝意を示す．

文　献

[1] “Globus Toolkit”. available at `http://www.globus.org/`.
[2] M. Murata and K. Kitayama: “A 1,000-channel WDM network can resolve network bottleneck”, Proceedings of the 7th Asia-Pacific Conference on Communications (APCC 2001) (Tokyo), pp. 113–116 (2001).
[3] E. L. Berger: “Generalized multi-protocol label switching (GMPLS) signaling functional description”, IETF RFC3471 (2003).
[4] S. L. Danielsen, C. Joergesen, B. Mikkelsen, and K. E. Stubkjaer,: “Analysis of a WDM packet switch with improved performance under bursty traffic conditions due to tuneable wavelength converters”, IEEE Journal of Lightwave Technology, **16**, pp. 729–735 (1998).
[5] D. Hunter, M. C. Chia, and I. Andonovic: “Buffering in optical packet switches”, IEEE Journal of Lightwave Technology, **16**, pp. 2081–2094 (1998).
[6] K. L. Hall and K. A. Rauschenbach: “All-optical buffering of 40-gb/s data packets”, IEEE Photonic Technology Letters, Vol. 10, pp. 442–444 (1998).
[7] K. Baba, R. Takemori, M. Murata and K. Kitayama: “A packet scheduling algorithm for the 2x2 photonic packet switch with FDL buffers”, Proceedings of 28th Europian Conference on Optical Communication 2002 (ECOC2002) (2002).
[8] T. Yamaguchi, K. Baba, M. Murata and K. Kitayama: “Scheduling algorithm with consideration to void space reduction in photonic packet switch”, IEICE Transactions on Communications, **E86-B**, 8, pp. 2310–2318 (2003).
[9] H. Nakamoto, K. Baba and M. Murata: “Shared memory access method for a λ computing environment”, Proceedings of IFIP Optical Networks and Technologies Conference (OpNeTec), pp. 210–217 (2004).
[10] E. Taniguchi, K. Baba and M. Murata: “Implementation and evaluation of shared memory system for establishing lambda computing environment”, Technical Report 255, IEICE (2004).
[11] Y. Sakai, K. Noguchi, R. Yoshimura, T. Sakamoto, A. Okada and M. Matsuoka: “Management system for full-mesh WDM AWG–STAR network”, 27th European Conference on Optical Communication, 2001, Vol. 3, pp. 264–265 (2001).
[12] A. Okada, H. Tanobe and M. Matsuoka: “Dynamically reconfigurable real-time information-sharing network system based on a cyclic-frequency AWG and tunable-wavelength lasers”, in Proceedings of ECOC2003 (2003).
[13] 岡田顕，田野辺博正，松岡茂登：“波長ルーティング技術を用いたダイナミックに再構成可能な情報共有ネットワーク”, 電子情報通信学会技術研究報告 (IN2003-332), **第 103 巻**, 692 号, pp. 423–427 (2004).
[14] “MPICH-G2”. available at `http://www3.niu.edu/mpi/`.
[15] “姫野ベンチマーク”. available at `http://accc.riken.jp/HPC/HimenoBMT/`.