特別研究報告

題目

λコンピューティング環境構築のための
共有メモリシステムの実装と評価

指導教官

宮原 秀夫 教授

報告者

谷口英二

平成 16 年 2 月 19 日

大阪大学 基礎工学部 情報科学科

平成 15 年度 特別研究報告

λコンピューティング環境構築のための共有メモリシステムの実装と評価

谷口英二

内容梗概

近年、ネットワークに接続された複数の計算機を用いて大規模な科学技術計算を行うためのグリッド技術がさかんに研究および開発されている。グリッド環境では、現在のインターネットの TCP/IP 上の Globus や MPI を用いてデータ交換を行いながら計算を行う。しかしながら TCP/IP のようなパケット単位のデータ交換ではパケット処理に要するオーバヘッドが大きく、大規模計算で行われる大量のデータ共有やデータ交換を行うには十分な性能を得ることは非常に難しい。

従って、各ノード計算機を接続している光ファイバを、インターネットとして利用するのではなく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。すなわち、各ノード計算機と接続しているネットワークを仮想的なリングネットワークとして利用し、このリング上にデータを載せる、あるいは伝送することにより、通信を意識することなくデータ共有あるいはデータ交換を可能とする技術である。

本報告では、分散計算を行う場合に、これらの技術のうちの一つである、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し、その性能を明らかにする。具体的には、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。このシステムでは、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アレイ導波路回折格子 (AWG) 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メモリは、AWG–STAR 上でリングネットワークを構成している全ノード計算機で同一のデータを保持している。すなわち、このシステムは AWG ルータと波長可変光源をベースとした動的な波長ルーティングを使用し、複数端末ノード計算機の共有メモリを共有する、多対多マルチキャストシステムである。

本報告では、AWG–STAR を用いた実験システム上で、実際のアプリケーションを動作させることにより共有メモリのアクセス手法の性能を明らかにしている。アプリケーションとして、分散計

算のベンチマークとして利用される SPLASH2 を用いた。MPI を用いた従来の TCP/IP による結果と比較することにより、共有メモリシステムおよびそメモリアクセス手法の評価を行った。その結果、AWG–STAR による分散計算は、共有メモリへの書き込み回数に大きく依存し、現状ではボトルネックとなっていることがわかった。そこで、効率よく共有メモリへの書き込みを行うことで AWG–STAR の性能を向上させることが可能であることを示した。

主な用語

λ コンピューティング環境、AWG–STAR、分散計算、共有メモリアクセス手法

目 次

図 目 次

表 目 次

1 はじめに

近年、画像処理や遺伝子解析、地球環境のシミュレートなど 1 台の計算機では実用的な時間内で解を算出できないような問題や 1 台の計算機では保持できない膨大なデータを扱う問題を計算する要求が生じている。

このような計算を実現する方法として、多数の計算機を高速なネットワークで接続して計算機間で協調動作を行いながら計算する PC クラスタや、インターネット上の多数存在する遊休 PC を利用するグリッドコンピューティングと呼ばれる技術がある。グリッド環境では、現在のインターネットの TCP/IP 上の Globus や MPI (Message Passing Interface) を用いてデータ交換を行いながら計算を行う。しかしながら、TCP/IP のようなパケット単位のデータ交換ではパケット処理に要するオーバヘッドが大きく、大規模計算で行われる大量のデータ共有やデータ交換を行うには十分な性能を得ることは非常に難しい。さらにこのような技術では高速かつ高品質な通信が要求される。

高速ネットワークの実現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、開発されている。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が行われている。しかしながら、現在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おこなっており、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フォトニック技術を下位のレイヤの通信技術とする GMPLS (Geralized Multi-Protocol Label Switching) と呼ばれるインターネットの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 化を実現するためのフォトニック技術に基くフォトニックパケットスイッチの研究がさかんに研究されている。しかしながれこれらの技術はパケットを情報を扱う粒度として用いおり、いかにして高速に転送するかに焦点をおくベストエフォート型通信であるため高品質性を達成するのは困難である。

光ネットワークを用いた高速な分散環境システムを目標とするミドルウェアとして、OptIPuter[1] がある。OputIPuter は地球規模での光ネットワークによるグリッド環境を構築するために現在研究、開発されている。OptIPuter ではネットワークの端末ノード計算機にまで光ファイバで接続され、各端末のアプリケーションレベルで、ネットワーク資源を発見、配置、調整を行い動的に端末間の専用光パスを設定し、小さなデータをバーストで送信するのでなく、巨大のデータをそのまま送信することを可能としている。専用の光パスを設定するために SLCP (Simple Lightpath Control Protocol)

と呼ばれるシグナリング手法を用いている。そしてまた、これらの光パスを有効利用してデータを転送するためのプロトコルも提案されている。例えば、各ルータにおいても輻輳制御を行う XCP (eXplicit Contorol Protocol) がある。さらにアプリケーションレベルでの様々な資源監理を行うためのミドルウェアの QUANTA が開発されている。しかしながら、やはり OptIPuter においても現在のインターネット技術をベースとしており情報の粒度としてパケットを用いるために、先に挙げたようなパケット処理の問題が生じうる。

そこで各計算機を接続している光ファイバを、インターネットとして利用するのではなく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。すなわち、各計算機と接続しているネットワークを仮想的なリングネットワークとして利用し、このリング上にデータを載せる、あるいは伝送することにより、通信を意識することなくデータ共有あるいはデータ交換を可能とし、高速性、高品質性を両立する技術である。

本報告では、分散計算を行う場合に、これらの技術のうちの一つである、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し、その性能を明らかにする。具体的には、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)」[2, 3, 4] を用いる。

このシステムでは、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メモリは、AWG–STAR 上でリングネットワークを構成している全ノード計算機で同一のデータを保持している。すなわち、このシステムは AWG ルータと波長可変光源をベースとした動的な波長ルーティングを使用し、複数端末ノード計算機の共有メモリを共有する、多対多マルチキャストシステムである。

本報告では、AWG–STAR を用いた λ コンピューティング環境を構築し、実際の分散計算アプリケーションを動作させることにより共有メモリのアクセス手法の性能を明らかにしている。具体的には、AWG および計算機ノードを接続して光リングネットワークを構築し、アプリケーションとして、分散計算のベンチマークとして利用される SPRASH2 を使用した。また AWG–STAR 上で分散計算を行うにために、分散計算を行うために必要ないくつかの関数を設計、実装している。AWG–STAR の共有メモリシステムを分散計算に利用した場合の性能を評価するために、比較対象として MPI を用い

た際の分散計算を動作させ、AWG–STAR を用いた場合の実行時間と MPI を用いた従来の TCP/IP による実行時間とを比較することにより、AWG–STAR を用いた λ コンピューティング環境における共有メモリシステムおよびそメモリアクセス手法の評価を行った。

以下、2 章では分散計算手法とそのシステムについて述べる。3 章では本報告で用いた AWG–STAR のシステムについて説明する。4 章で AWG–STAR を用いて分散計算を行った場合の AWG–STAR 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　分散計算手法とそのシステム

一般に、分散計算を行うシステムは、二つの種類に大別できる。ひとつは、1 台の計算機に複数のプロセッサを搭載し、演算処理を行うためのシステムである。この方式では、全てのプロセッサがアドレス空間を共有しており、全プロセッサから単一の物理アドレスによりアクセスされる。このようなモデルとしては UMA (Uniform Memory Access) モデルが挙げられる。もうひとつは、ネットワークを通じて複数の計算機を接続して演算を行うシステムである。この方式では、全てのプロセッサが同一のアドレス空間を共有する場合と各プロセッサが互いに独立したアドレス空間のメモリを持つ場合とがある。前者は、全てのプロセッサが同一のアドレス空間を共有するため、他のプロセッサのメモリへのアクセスはバスやネットワークなどの結合網を経由してアクセスすることができる。例として NUMA (Non-Uniform Memory Access) モデルがあげられる。一方、後者は、ネットワークを通じたメッセージの交換によって計算を進め、それぞれの計算機のメモリは、全体の共有メモリとしては機能しない。このようなモデルとしては NORA (NO Remote Memory Access) モデルがあげられる。

前章でも述べたように、本報告では、広域に分散した複数の計算機を用いて分散計算を行う環境を想定する。従来のシステムにおいては、複数の計算機で分散計算を行う場合は、メッセージ交換に基づいた手法を用いて計算機間で必要なデータ交換を行い、計算を実行していた。ネットワークは、インターネットなどのパケット交換を用いているため、転送確認処理やパケット損失処理などのオーバーヘッドが大きく、十分な高速計算ができないことが問題となる。そこで、計算機間を接続する光ファイバをインターネットなどのパケット網として利用するのではなく、高速な計算機間の通信チャネルとして、あるいはファイバ自体を共有メモリとして分散計算を行うことができる、λ コンピューティング環境を提案する。

本章では、従来用いられているメッセージ交換による手法として MPI を用いた分散計算システム、および我々が提案する新たな分散計算環境である λ コンピューティング環境における分散計算システムついて述べる。

2.1　MPI による分散計算システム

本節では、従来の分散計算システムにおける MPI を用いた計算手法とその分散計算システムにおけるデータ共有のためのアクセス手法について述べる。

分散計算システムとしては、複数の計算機をネットワークで接続し、メモリアクセスへの同一仮想空間を持たず、各計算機はそれぞれのメモリをローカルメモリとして利用し、複数の計算機間では、MPI によるメッセージ交換を行うことにより、分散計算を実行する（図 1)。MPI は、分散メモリ型並列処理の基本であるメッセージ交換ライブラリの規格である。すなわち、MPI を用いたプログラムの実行においては、各計算機は共有メモリを持たないため、その動作を明示的なメッセージの送信および受信によって協調動作する。

従来の広域に分散した分散計算システムでは、ネットワークとしてインターネットを利用し、メッセージ交換を行う。インターネットにおいては、パケット交換によるデータ転送を行うため、その処理のオーバヘッドが問題となる。まず、送信側計算機では、交換すべきメッセージをパケットに分割し、ヘッダをつけ、ネットワークインターフェースを通じてパケットを送出する。インターネット内では、各中継ルータにおいてストアアンドフォワード方式により転送するため遅延が発生する。受信側計算機で受信されたパケットは再びネットワークインターフェースを通じてメッセージに組み立てられる。また MPI の送受信プロトコルには、TCP プロトコルを利用するため、パケットの送受信確認が必要となる。さらに、パケットが途中経路で失われると再送が行われ、多大な遅延が発生する。このように、インターネットを利用したデータ転送にはいくつかのオーバヘッドの要因が存在し、これらが分散計算の性能に影響を与えると考えられる。

2.2 λ コンピューティング環境における分散計算システム

分散計算を行う複数の計算機を接続するネットワークを、パケット交換に基づく既存のインターネットではなく、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する。すなわち、λ コンピューティング環境では、各ノード計算機を接続するネットワークを仮想的な光リングネットワークとして利用し、このリングネットワーク上にデータを載せるあるいは高速に伝送することにより、通信を意識することなくデータ共有あるいはデータ交換を可能とする技術である。

λ コンピューティング環境においては、次の二つのシステムを対象としている。ひとつは、λ コンピューティング環境における仮想光リングネットワークを、共有メモリとして利用し、各ノード計算機内のローカルメモリをキャッシュなどに利用する場合（共有メモリ型アーキテクチャ）[5]、もうひとつは、高速な伝送路として利用し、共有すべきデータは各ノード計算機のメモリ内におく場合

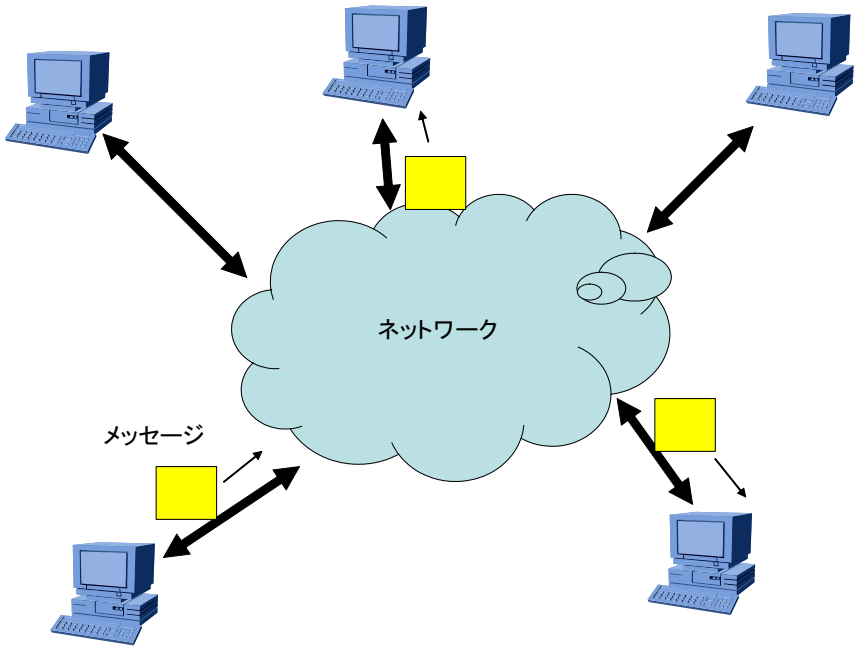

図 1: MPI による分散計算

（高速チャネル型アーキテクチャ）である。以下では、それぞれのシステムとメモリアクセス手法について述べる。

2.2.1 共有メモリ型アーキテクチャ

各ノード計算機を接続する仮想的な光リングネットワークを共有メモリとして用い、各ノード計算機のローカルメモリやプロセッサにおけるキャッシュを共有メモリに対するキャッシュとして利用する場合を考える。

光リングネットワークを用いた共有メモリシステムは、WDM により多重化された波長パスを共有メモリとして用いる。その波長パスへのアクセスは、各波長へアクセス可能なインターフェースを介して行う。その際、利用可能な波長パスのうち、ノード計算機間の通信用に数波を割り当て、残りの波長を共有メモリとして用いる。

また、光リングネットワークの共有メモリシステムは、各計算機のプロセッサが一つの共有メモリにアクセスするシステムであるたため、従来の共有メモリシステムに近く UMA 型であるといえる。しかしながら、光リングネットワークを共有メモリとして利用する場合、同一計算機内の共有メモリバス結合とは異なり、長距離の光ファイバ上に展開しているため共有メモリに対するアクセスのタイ

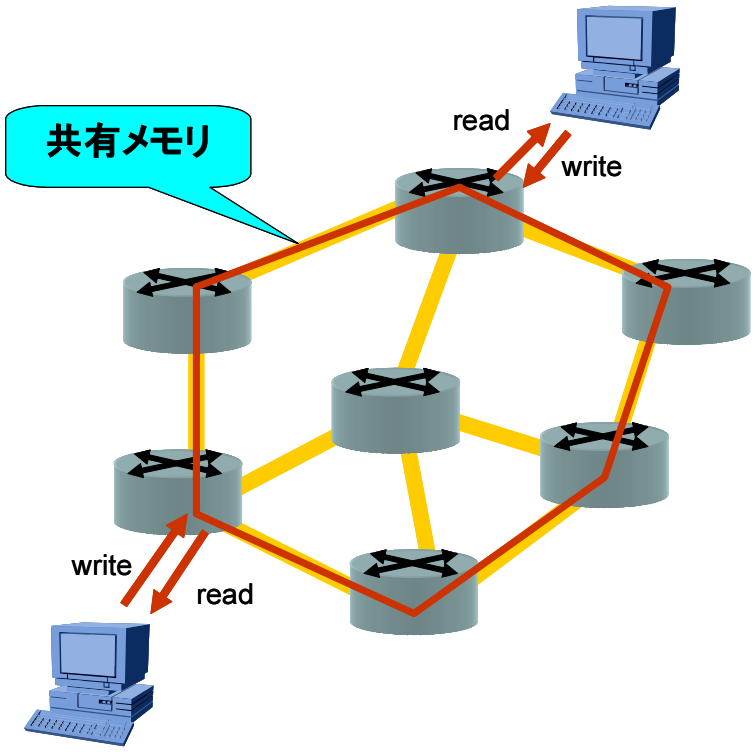

図 2: 共有メモリ型アーキテクチャ

ミングや頻度に制約を受ける。従って、通常の共有メモリシステム以上に、光リングにおける共有メモリと各計算機群のキャッシュの整合性を十分考慮する必要がある。

具体的には、従来の共有メモリシステムでは、共有バスにより共有メモリにアクセスするが、光リングネットワークによる共有メモリシステムでは共有バスはなく、各計算機が直接共有メモリにアクセスする。さらに、従来の共有メモリシステムは共有メモリにランダムアクセス可能であるが、光リング共有メモリシステムでは、リングネットワーク上をメモリ空間が展開しているためランダムアクセスができない。すなわち、共有メモリにアクセスする場合、共有バスにおける競合は存在しないが、当該のメモリ空間にアクセスできるまで待つ必要がある。さらに、光リングが広域に展開している場合、従来の共有メモリシステムに比べ、アクセス時間が非常に大きくなる可能性がある。また、各ノード計算機が共有メモリのコピーをローカルメモリにキャッシュとして持つため、キャッシュの整合性を考慮し、また全てのノード計算機が一つの共有メモリにアクセスするため、書き込み競合についても考慮する必要がある。

2.2.2　高速チャネル型アーキテクチャ

共有メモリ型アーキテクチャは、データを共有する場合に有効であると考えられるが、まったく新しいアーキテクチャであり、ハードウェアの開発にも時間がかかると思われる。そこで、光リングネットワークを共有メモリではなくデータ共有を図るための高速通信チャネルとして利用するアーキテクチャが考えられる。このアーキテクチャにおいても二つの手法が考えられる。ひとつが各ノード計算機のローカルメモリを統合して共有メモリとし、光リングを通じて他ノード計算機のローカルメモリアドレスへアクセスする手法（図 3)、もうひとつが各ノード計算機にそれぞれ共有メモリ領域を用意し、すべてのノード計算機が同じデータを持つ手法である（図 4)。

前者は、光リングネットワークを結合網とする NUMA 型のモデルであるといえる。この方式では、ノード計算機間のデータ転送、キャッシュの整合性などの制御用データ転送に光リングを用いる。すなわち、各ノード計算機が共有メモリの異なる領域を持つため、あるノード計算機が共有メモリの他のノード計算機の持つ領域にアクセスする場合に、高速チャネルを利用してデータにアクセスする。また、ローカルメモリやプロセッサキャッシュをキャッシュとして利用する場合は、それらの更新に合わせ、分散した共有メモリの更新も必要になる。

後者は、各ノード計算機がすべて同じデータを持つため、データ更新の際に高速チャネルを利用して、全ノード計算機に対して更新されたデータの配送を行う。従って、共有メモリのアクセスは、共有メモリへ書き込む場合は、高速チャネルへのアクセスが生じるが、読み出し時には各ノード計算機の共有メモリからデータを取得すればよく、高速チャネルへのアクセスは発生しない。

後者のシステムを実現するためのひとつの例として、日本電信電話株式会社フォトニクス研究所が中心となって開発した「情報共有ネットワークシステム」を利用することが考えられる。この「情報共有ネットワークシステム」は、複数波長を利用することにより多対多の通信を実現し、明示的にデータ転送することなく、複数の計算機間でデータの共有ができる特長を有する。この「情報共有システム」は AWG-STAR と呼ばれるネットワークシステムで構成されている。具体的な利用例としては、多地点で撮影した映像を他の計算機でも共有できる「映像共有アプリケーション」がある。

本報告では、分散計算のためのシステム構築に AWG–STAR ネットワークシステムを用いて、分散計算に利用した場合の性能を測り、共有メモリシステムの有効性を明らかにすることを目的としている。

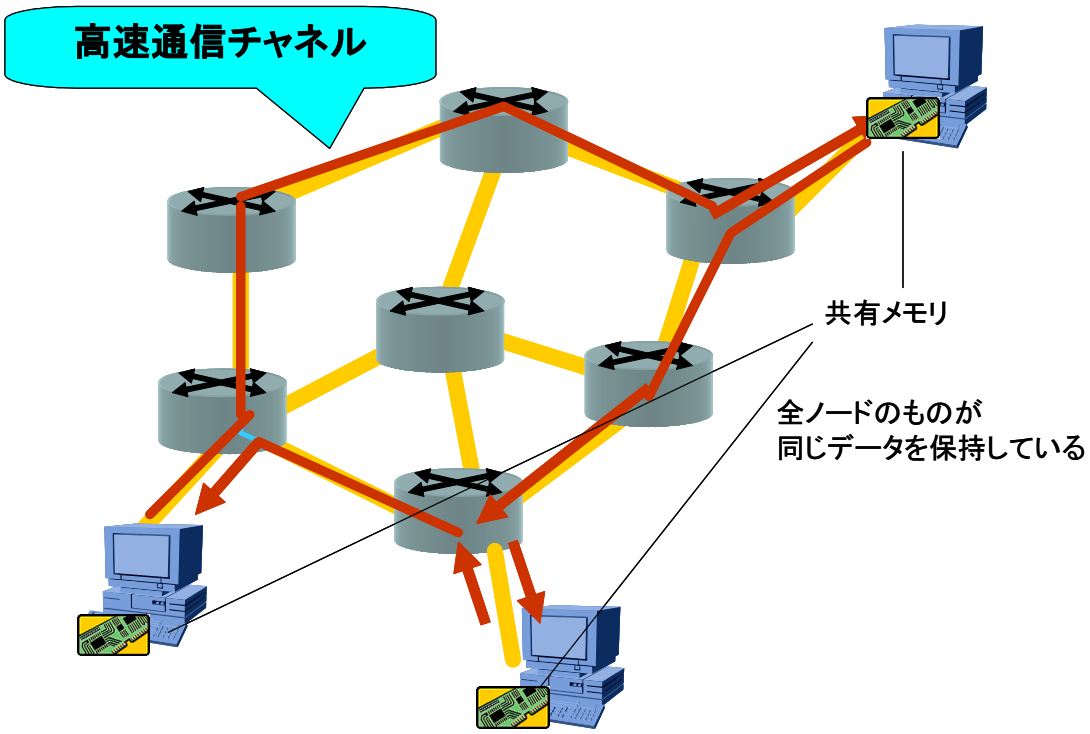

図 3: 高速チャネル型アーキテクチャの構成 1

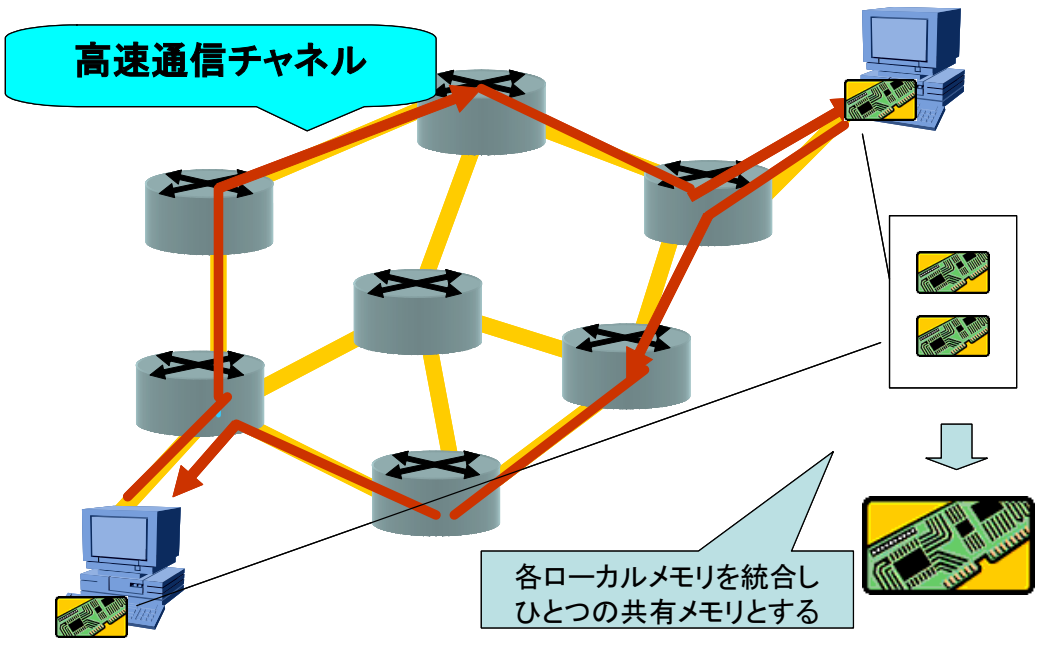

図 4: 高速チャネル型アーキテクチャの構成 2

3　AWG–STAR を用いた共有メモリアクセス手法の実現

本報告では、λコンピューティング環境を構築するために、ひとつの手法として AWG–STAR ネットワークシステムを利用する。本章では、AWG–STAR ネットワークシステムについて述べ、AWG–STAR を用いた分散計算手法について説明する。

3.1　AWG–STAR ネットワークシステムの概要

AWG–STAR ネットワークシステムは、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 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。AWG 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、すべてのノード計算機で同一のデータを保持する共有メモリを持ち、AWG および WDM を利用して構成された高速な光リングネットワークを利用することによりノード計算機間で共有メモリ上のデータ交換をリアルタイムに行うことができる。

2.2.2 節でも紹介したが、このシステムを用いたアプリケーション例として、「映像共有アプリケーション」がある。このアプリケーションにおいて、AWG–STAR 上の各ノード計算機は、カメラから取りこんだ画像データを共有メモリ上の自ノード計算機に割り当てられたアドレス範囲に書き込む。従来ならば、画像データを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 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を流れ、全ノード計算機の共有メモリの更新が行われる。このように AWG–STAR を用いることにより、共有メモリ上のデータの共有は、共有メモリに書き込む手続き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 を通じて共有メモリに配信される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。

そこで、本報告では、この共有メモリを分散計算に用いることを考える。すなわち、各ノード計算機においてはそれぞれが独自に計算を行い、計算に必要な共有すべきデータは共有メモリを用いて分散計算を行う。しかしながら、全ノード計算機間でデータを共有するためには、データ更新時にデータが光リングネットワークを 1 周回する必要があり、そのための遅延が生じる。したがって、分散計算を行うにはこの遅延時間を十分考慮する必要がある。

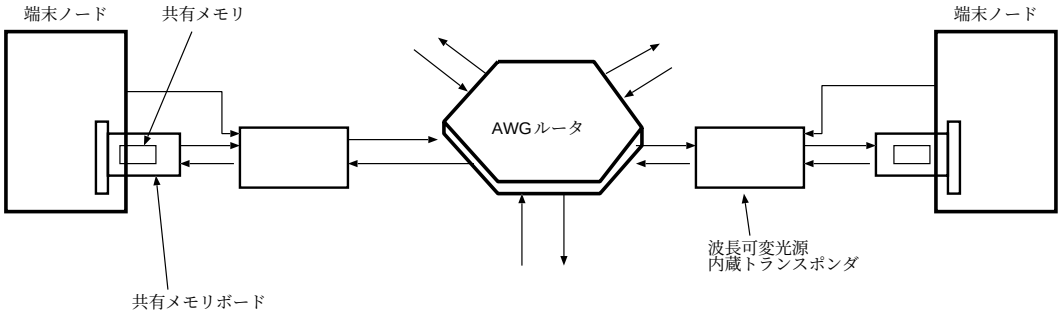

図 5: AWG–STAR システム構成

表 1: 共有メモリボードの仕様

光インタフェースの伝送速度	2.152 Gbps
ノード計算機の 1 回当たりの転送データ量	1 KByte
ノード計算機でのフレーム転送処理遅延	500 ns
共有メモリへの書き込み	170 MBytes/s
共有メモリから読み出し	380 MBytes/s

3.2 AWG–STAR の構成

AWG–STAR ネットワークシステムの概略図を図 5 に示す。このシステムでは、各ノード計算機は波長可変光源を通じて光ファイバにより AWG 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、光リングネットワークを形成している（図 6)。AWG–STAR に使用している光ファイバはシングルモード光ファイバを使用している。AWG–STAR 上の全ノード計算機は、共有メモリボードを搭載し共有メモリはこのボード上にある。以降、特に断らない限り共有メモリは共有メモリボード上のものを指す。表 1 に共有メモリボードの仕様を示す [4]。

AWG ルータは波長による動的なルーティングを行うルータである。AWG ルータは 32 個の入力ポートと 32 個の出力ポートを持っており、入力ポートに入力された光はその波長によって出力ポートが決定される。波長の割り当てに例ついて表 2 に示す。たとえば、入力ポート 2 に波長 52 の光が入力されれば出力ポート 1 に出力される。ただしこの数値は AWG ルータのために定められている独自の波長番号である。

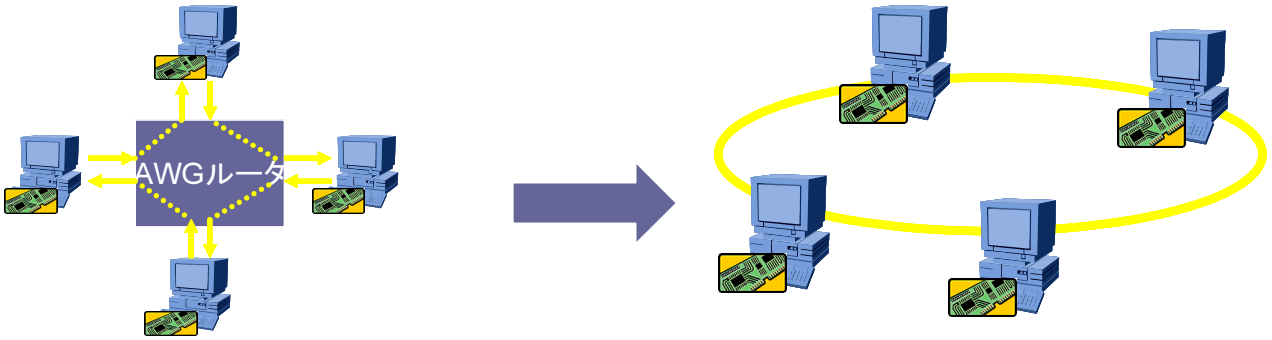

図 6: 論理的光リング

表 2: AWG ルータの入出力ポートと波長の対応

出力 / 入力	ポート 1	ポート 2	ポート 3
ポート 1	51	52	53
ポート 2	52	53	54
ポート 3	53	54	55

3.3 共有メモリへのアクセス

共有メモリ上のデータは、光リングネットワークを構成している全ノード計算機で同一のものを保持している。あるノード計算機が共有メモリのデータを更新を行うと、この時更新されたデータが光リングネットワークを周回し、光リングネットワークに接続されたノード計算機上の共有メモリの同一アドレスのデータの更新を行う。すなわち、共有メモリ上のデータは、光リングネットワークを周回しながら他ノード計算機の共有メモリの更新を行うことによりすべてのノード計算機でデータの共有を実現している。

3.3.1 共有メモリへのアクセス手法

共有メモリへのアクセス手法は二通りある。ひとつは共有メモリボードの機能を用いた DMA (Direct Memory Access) アクセスであり、もうひとつはポインタを用いたアクセスである。これらは、共有

メモリの先頭からのオフセットもしくは直接アドレスを指定することでアクセスが可能である。

共有メモリは、共有メモリボード上にあり、計算機とはPCIバスで接続されている。そのため、共有メモリへの読み出しおよび書き込みはPCIバスを経由して行われるため、ローカルメモリへアクセスする場合よりも遅延時間が大きくなる。すなわち、共有メモリへのアクセスによる生じる遅延は、共有メモリへの書き込みもしくは読み出し時間とPCIの転送時間である。

共有メモリからのデータ取得については、自ノード計算機の共有メモリから読み出すため、光リングネットワークの通信路に負担をかけない。一方、共有メモリへの書き込みに際しては光リングネットワークへのアクセスが発生する。

3.3.2 高速チャネルにおける通信手法

光リングネットワーク上では、常にひとつのトークンが流れており、各ノード計算機はそのトークン上に更新を行ったデータに関する送信フレーム（アドレス、データ、制御コード、CRC）を付加することにより通信を行う。共有メモリの更新には二つの場合がある。ひとつは自ノード計算機の共有メモリに書き込む場合であり、もうひとつは他ノード計算機からの共有メモリの更新情報を受信した場合である。

1. 自ノード計算機の共有メモリに書き込む場合

   この場合、まず自ノード計算機の共有メモリに書き込み、その後トークンが回ってきた際に、送信フレームをトークンに附随している送信フレームの最後尾に付加し、次のノード計算機にトークンを転送する。リングを1周し、トークンが再度回ってきたら先ほど付加した送信フレームを削除する。ただし、送信中にエラーが発生すれば、リトライが行われる。

2. 他ノード計算機からの更新情報を受信した場合

   トークンが回って来ればトークンに付加されている他ノード計算機の送信フレームを確認する。他ノード計算機の更新情報がトークンに附随していれば、データを読み込み自ノード計算機の共有メモリを更新し、次のノード計算機に向けてトークンを流す。

3.3.3 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。例えば、更新されたデータを全ノード計算機が共有するためには、少なくともデータが光リングネットワークを 1 周は周回しなれけばならない。従って、ノード計算機間で同期をとる場合には、その遅延時間のため、性能に影響を与える。

光リングネットワークを周回する際に生じる遅延の原因は二つある。ひとつは光ファイバによる伝搬遅延であり、もうひとつは共有メモリボードにおける転送処理遅延である。

- 光ファイバによる伝搬遅延時間

  光ファイバをによる伝搬遅延時間は 5ns/m である。

- データ転送のため処理遅延時間

  各ノード計算機において前のノード計算機から転送されてきたフレームを、次のノード計算機に転送するために処理時間を必要とする。具体的には送信フレームの削除と追加、共有メモリへの反映のために 500ns の時間を必要とする。

従って、光リングの長さを L、ノード計算機数を N とすると、光リングを 1 周するのに必要な遅延時間は $500N + 5L$ns となる。

以上のことから AWG–STAR を用いた共有メモリシステム上で分散計算を行う場合、データ共有のための通信時間、および通信は共有メモリへの書き込みにより発生するため、共有メモリへの書き込み回数が性能を左右すると考えられる。次章では実際にアプリケーションを動作させ、検証する。

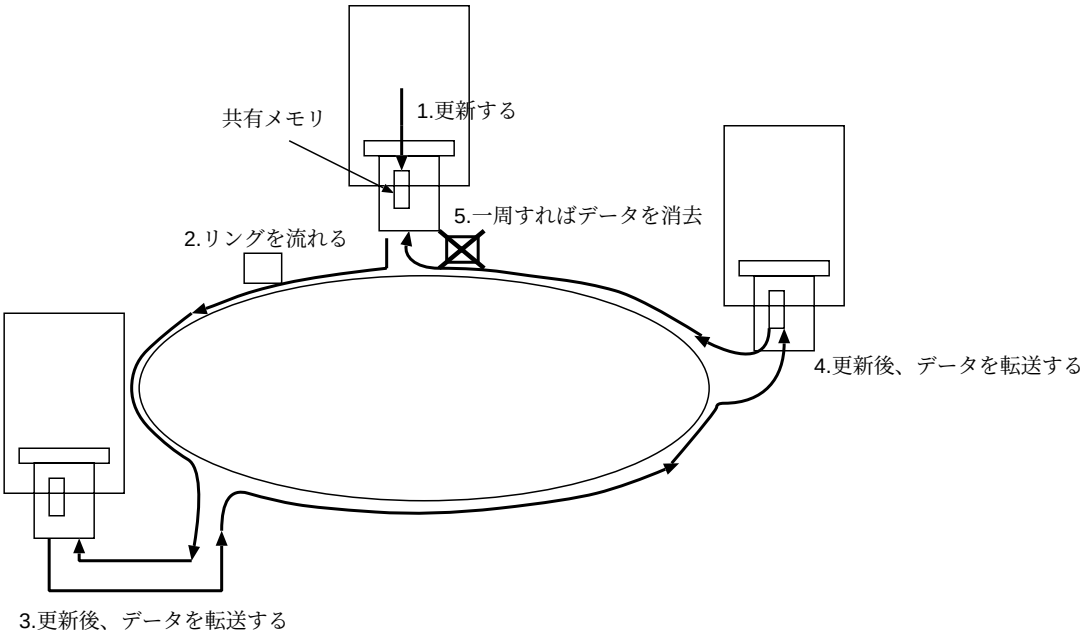

図 7: データ共有の流れ

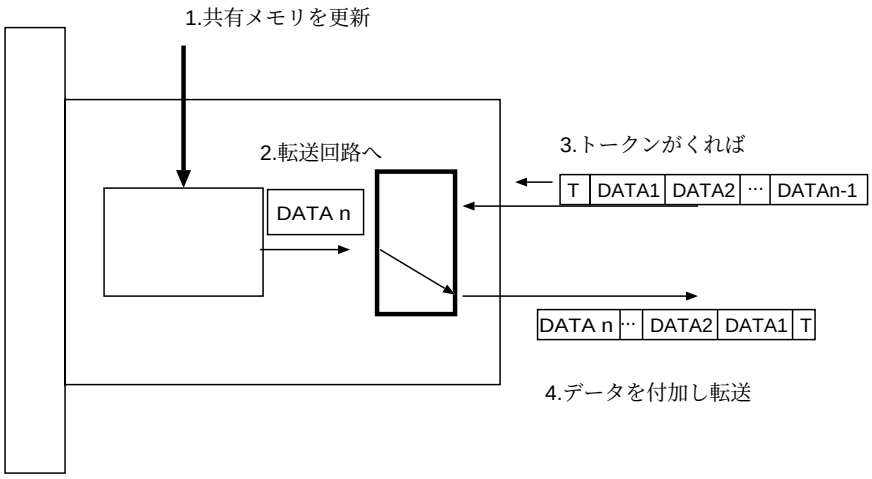

図 8: 更新データの送信

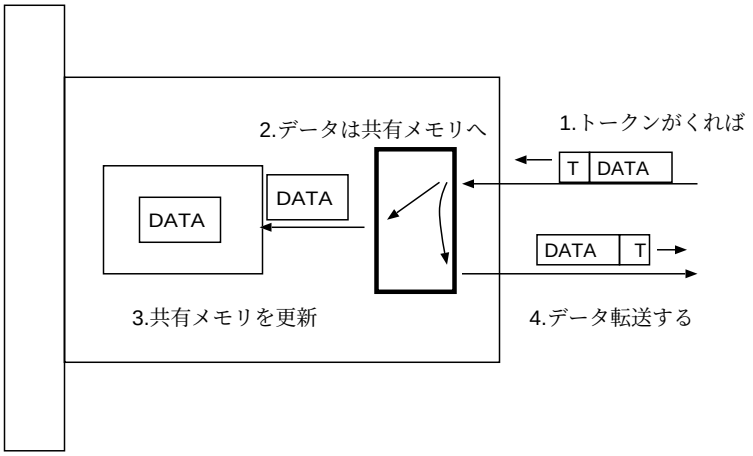
2.データは共有メモリへ
1.トークンがくれば
T
DATA
DATA
DATA
DATA
T
3.共有メモリを更新
4.データ転送する

図 9: 更新データの受信

4 実験と評価

本章では、並列アプリケーション集である SPLASH2[6] の中のいくつかのプログラムを動作させ、実行時間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。

4.1 実験システム環境

評価に用いた計算機の仕様を表 3 に、実際に構築した光リングネットワークを図 10 に示す。実験に使用した計算機の台数は 1 台から 3 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。表 4 に今回の実験で構成した光リングネットワークと、1 周に要する時間を示す。MPI を用いた方式による実験でも、使用した計算機は AWG–STAR による実験と同じ計算機を使用した。また、計算機は、100Mbps の Ethernet で接続され、ひとつのスイッチングハブに全て接続されている。

4.2 評価に用いるアプリケーション

SPLASH2 は、スタンフォード大学で開発された分散計算用のベンチマークアプリケーションである。

プログラムは分散計算を行うために必要となるバリア同期関数などは実装されておらず、実験環境に合わせてユーザ側で作成する必要がある。そのため AWG–STAR を用いる方式では AWG–STAR の機能を利用した命令を作成し、MPI を用いる方式では MPI ライブラリ関数で対応する命令に置き換えて実現している。プログラムにおいて共有するデータは、全て共有メモリ上に配置する。プログラムの高速化を図るためのプログラムのチューニング、例えばプログラムの実行中にローカルメモリにコピーし、ローカルメモリ上で処理をしてから共有メモリに書き戻すといった処理は行っていない。これはプログラムの改変を最低限にとどめ、性能を測るためである。

本報告では、SPLASH2 のアプリケーションの中から次の三つのプログラム、基数ソートプログラム、LU 分解プログラム、高速フーリエ変換プログラムを実験に使用した。

- 基数ソートプログラム（以下、RADIX）

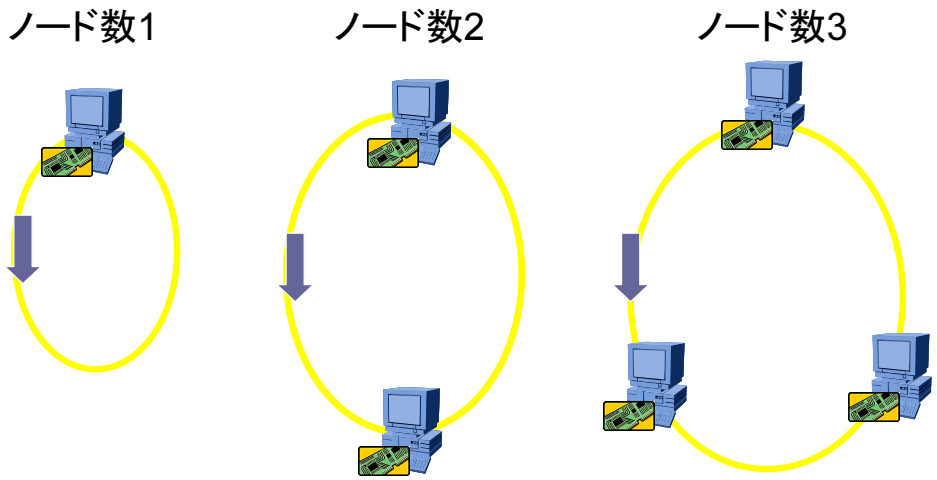

図 10: 実験で構成した光リングネットワーク

計算を行うノード計算機数の分だけソートの対象となるキーの配列を等分割し、各ノード計算機に割り当てる。各ノード計算機は割り当てられた配列について処理を行うことで、並列化を実現している。RADIX は共有メモリへの書き込み回数が少なく、またほぼ全ての処理において並列化が達成されている。

- LU 分解プログラム（以下、LU）

  対象とする行列をブロックに分割し、各ブロックに処理を担当するノード計算機を割り当てる。各ノード計算機は割り当てられたブロックについてのみ処理を行うことで並列化を実現している。LU では共有メモリへの書き込みが頻繁に発生する。あるデータを書き込んだ後に、次のデータを書き込むことが発生する。また、LU においては全ての処理において完全な並列化は行われておらず、ある処理においては特定のノード計算機のみが処理を行う。

- 高速フーリエ変換プログラム（以下、FFT）

  変換の対象となるデータを行列状に配置し、行列を行についてノード計算機数の分だけ等分割を行い、各ノード計算機に割り当てる。各ノードは割り当てられた行列についてのみ高速フーリエ変換を行うことにより並列化を実現している。また、FFT も処理の過程で共有メモリへの書き込みが頻繁に発生する箇所がある。

表 3: 実験に用いた計算機の仕様

CPU	Xeon 2.80 GHz
メインメモリ	SDRAM 512 MB
1 次キャッシュ	512 KB
2 次キャッシュ	512 KB
NIC	Intel PRO/1000MT
PCI バス	64 bit / 66 MHz
PCI 転送速度	533 MBytes/sec
OS	Redhat Linux 7.3
コンパイラ	gcc 2.96
MPI ライブラリ	MPICH 1.2.5

表 4: 実験に用いた光リングネットワークの仕様

ノード計算機数	リング長 [m]	1 周にかかる時間 [ns]
1	10	550
2	20	1100
3	30	1050

4.3 共有メモリシステムの性能評価

本節では、実際に RADIX、LU、FFT を使用して実験を行い、共有メモリシステムとそのアクセス手法の評価を行う。各プログラムの実行結果の測定において、実時間および CPU 時間を測定している。実時間とは、計測の開始時から終了時までの経過時間であり、CPU 時間とは計測の開始時から終了時までの間に CPU を消費した時間である。実時間には通信のための時間が含まれるが、CPU 時間には含まれない。

4.3.1 基数ソートプログラムによる実行結果

図 11 および図 12 に RADIX の実行時間を示す。MPI を用いた場合の結果も比較の対象として併せて示す。これらの図より、MPI を用いてノード計算機数 2 の場合が実行実時間が最も多くかかっ

ている。これは各ノード計算機の結果を分配および集約に要する時間、特に集約する際の通信時間が影響する。処理の過程において共有するデータ量は定数量であるため、それほど通信時間はかからない。しかし結果の集約には各ノード計算機からの通信が必要となり、これは並列化ができない。従って、最後の集約に必要な通信量はソート対象の要素数を n とすると $O(n)$ である。AWG–STAR では結果の集約は各ノード計算機がそれぞれ独立に行えるのに加え、他の処理に行われる書き込み回数はパラメータとして与える基数の値によって決定される。しかしこの値は、実行に際し定数として与えられるため要素数とは無関係である。よって、AWG–STAR を用いた場合の書き込み回数は後から説明する LU や FFT に比べると共有メモリへの書き込み回数は少なく、通信が頻繁に行われる訳ではないため実行時間が短縮される。ノード計算機数が 1 の場合に MPI を用いる方式が実時間、CPU 時間がともに最も高速なの理由は、全ての処理をローカルメモリ上で行い、共有メモリへのアクセス遅延や通信が発生しないためである。

4.3.2 LU 分解プログラムによる実行結果

図 13、図 14 に LU の実行時間を示す。LU の実時間における実行時間は AWG–STAR を用いた場合はノード計算機数の増加に伴い実行時間が減少しているが、MPI を用いた場合はノード計算機数の増加に伴い、実行時間が増えている。これは MPI のデータ共有のための通信が並列化できないこと、およびノード計算機数が増加することによりデータ共有のための送信回数が増えるためである。

LU を用いた場合の実行時間は AWG–STAR を用いた場合が MPI を用いた場合に比べて性能が十分でない。その理由は次のように考えられる。AWG–STAR において LU を用いた場合、共有メモリへの書き込みアクセスが多く行われる。ノード計算機数が 2 の時に行列サイズ 480 の場合を考えると、共有メモリへの書き込み回数が 1 ノード計算機あたり約 1800 万回あるが、そのうちの 1700 万回が共有する必要のないデータの書き込みである。これは書き込み回数全体の 94%にあたる。したがってこの 9 割の通信による遅延のため AWG–STAR の性能が十分に生かせずに性能の低下を招いたといえる。プログラムのチューニングにより共有メモリへのアクセス回数を減らすことにより AWG–STAR の性能の向上が可能であると考えられる。

4.3.3 高速フーリエ変換プログラムによる実行結果

図 15、図 16 に FFT の実行時間を示す。FFT を用いた場合の実行結果も LU を用いた時と同様に、AWG–STAR を用いた場合が MPI を用いた場合に比べて性能がよくない。これも LU の時と同様に、実行中に共有メモリに 1 要素づつ書き込む処理が多いため、通信量が多くなってしまい、それによる遅延のためである。

一般に FFT の計算量は、要素数を N とすると、$O(N \log N)$ である。今回は並列化を行っているのでノード計算機数を P とすると $O(\frac{N}{P} \log \frac{N}{P})$ となる。ここで、SPLASH2 の FFT は常に要素数は 2 の巾乗を要求しているため、要素数に対してノード計算機数が小さくなりノード計算機数は無視できる。従って計算量は $O(N \log N)$ となる。SPLASH2 の FFT の要素数は $N = 2^k$ と表せるので、計算量 $O(k2^k)$ となりこれが通信量となる。一方、MPI を用いた場合では通信は分割されたデータを集約、分配のために行われ、通信量は $O(N) = O(2^k)$ である。したがって、k が小さい範囲ならばこれらの差は小さいが、k が大きくなるにつれて AWG–STAR の方が通信量が多くなる。そのために、AWG–STAR の性能が十分に発揮されなくなり、MPI に対する性能の差が表れる。

4.4 共有メモリアクセス手法の高速化

前節において、LU および FFT において AWG–STAR による共有メモリシステムでは十分な性能が得られないことがわかった。その主な要因が共有メモリへのアクセスの多さによる遅延である。すなわち、ある処理において共有メモリへの書き込み回数が増大するため、光リングネットワークの周回数が増大し実行時間の増加につながっている。さらに光リングネットワーク 1 周に生じる遅延時間は光ファイバによる伝搬遅延と共有メモリボードによるフレーム転送遅延がある。そこで本節では、共有メモリアクセス手法の性能向上のための手法について考察する。

4.4.1 共有メモリボードの高速化による改善

AWG–STAR を用いた共有メモリシステムの性能が十分でない要因のひとつに、共有メモリボードにおける処理遅延時間がある。共有メモリボードにおいて、ネットワークから入力されたデータを共有メモリへ反映する時間や、次のノード計算機へデータを転送するための波長の変換時間のための時間に約 500ns を要する。この遅延は、共有メモリへデータを書き込む際に必ず発生するため、データの書き込み回数に比例して遅延が増大している。そのため今後の開発により、共有メモリボー

ドでの遅延を小さくすることができれば、それに見合った性能の向上が期待できる。共有メモリボードの高速化が行われ、その処理遅延時間が現在の $\frac{1}{10}$ となったと仮定したときに LU 分解をノード計算機数 3 で実行した場合の実行時間の理論値を図 17 に示す。この図より共有メモリボードの高速化により AWG–STAR の性能改善が可能であり、約 50%改善されていることがわかる。

4.4.2 プログラムのチューニングによる改善

AWG–STAR を用いた共有メモリシステムの性能が十分でないもうひとつの要因として、共有メモリへの集中的な書き込みアクセスがある。このような共有メモリへの書き込みが頻繁に発生するとデータの周回に伴う遅延が発生する。この遅延を減らすには、共有メモリにデータをまとめて書き込むことが考えられる。まとまったデータを一度に書き込むことで周回の回数が減り遅延の減少、ひいてはプログラムの実行時間の減少につながる。共有メモリに連続して書き込んでいたものをまとめて書き込むように変更するには、書き込むデータをローカルメモリ上に一時的にスプールし、一定量になれば共有メモリに書き込むようにすればよい。

LU について考える。LU においては各ノード計算機はブロックを割り当てられ、ブロック単位で処理を行う。4.3 節における実験では、ブロック内のデータを共有メモリから読み出し演算を行い共有メモリに書き込むようになっているため、この操作がボトルネックとなっている。このボトルネックを解消するための実装の改変として、ブロックを共有メモリからローカルメモリにコピーし、必要な演算はコピーしたローカルメモリ上のデータを用いて演算を行い、ブロックの処理が終了すれば共有メモリに書き込む。ブロックのサイズは 16×16 でありブロックの各要素は double 型 (8bytes) であるので、1 ブロックの大きさは 2KB である。2KB のデータを共有メモリに書き込んだ場合、リングを 2 周すればよい。連続で書き込む場合は、実際に行う処理に依存するが最大で 16^3 回書き込むため、この回数がそのままリングの周回回数につながる。図 18 に実際に先に述べた改善例を実装した時の実行結果を示す。この方法によるプログラムのチューニングより、ノード計算機数 3 で行列サイズが 480 の場合の書き込み回数を 1160 万回から 3000 回に、実行時間を改善前の約 20%にまで減少することに成功した。

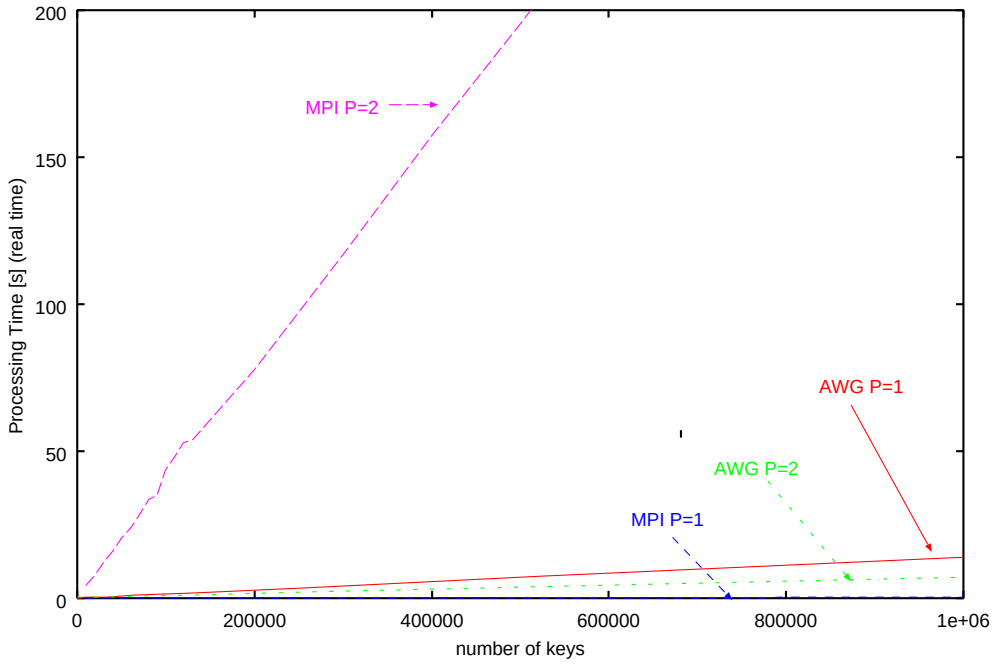

図 11: 基数ソートの実行時間（実時間）

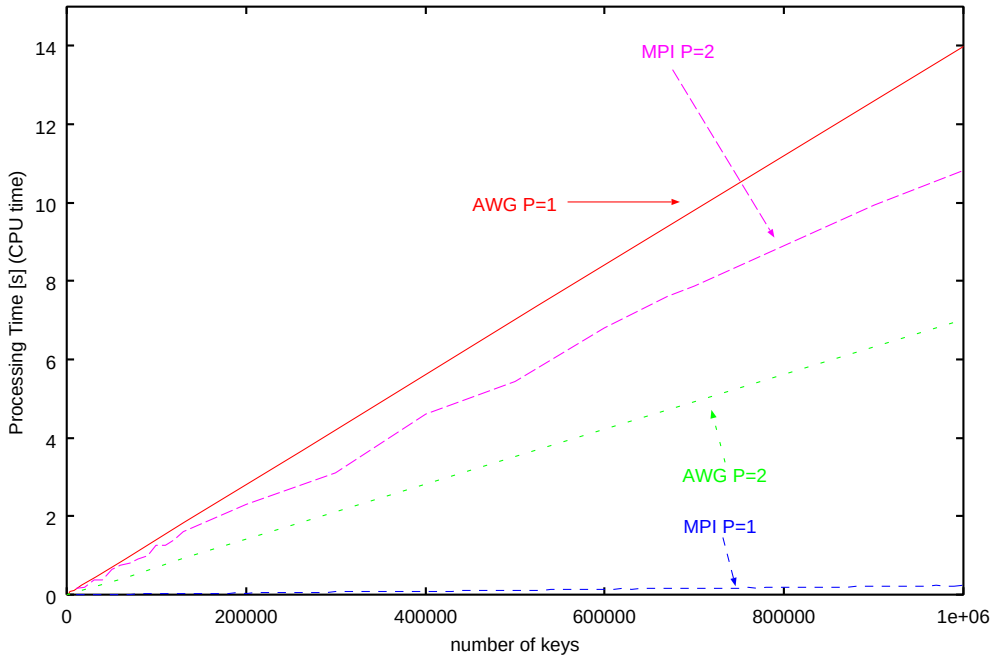

図 12: 基数ソートの実行時間（CPU 時間）

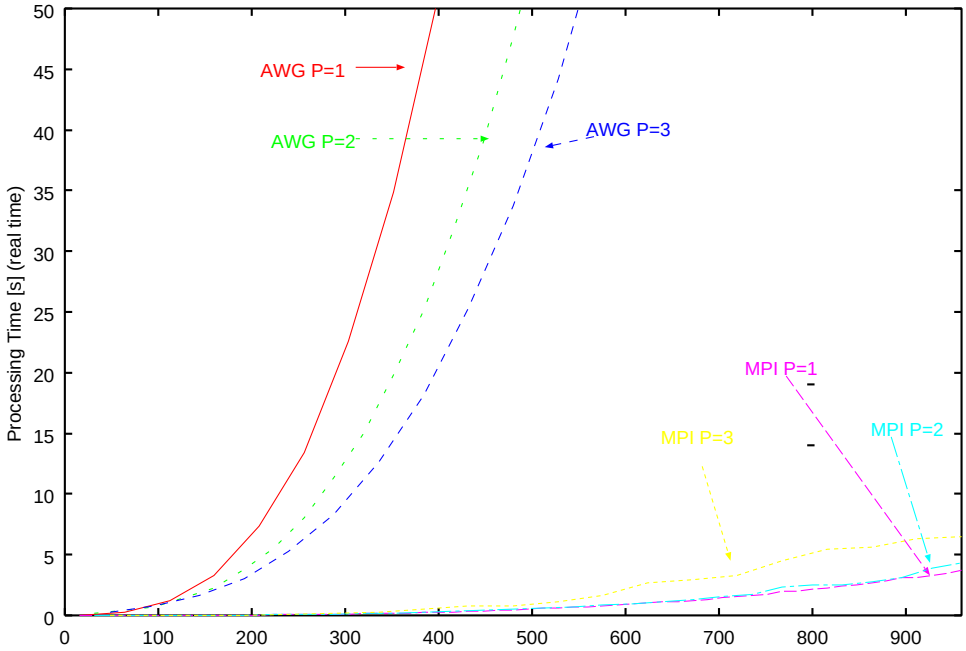

図 13: LU 分解の実行時間（実時間）

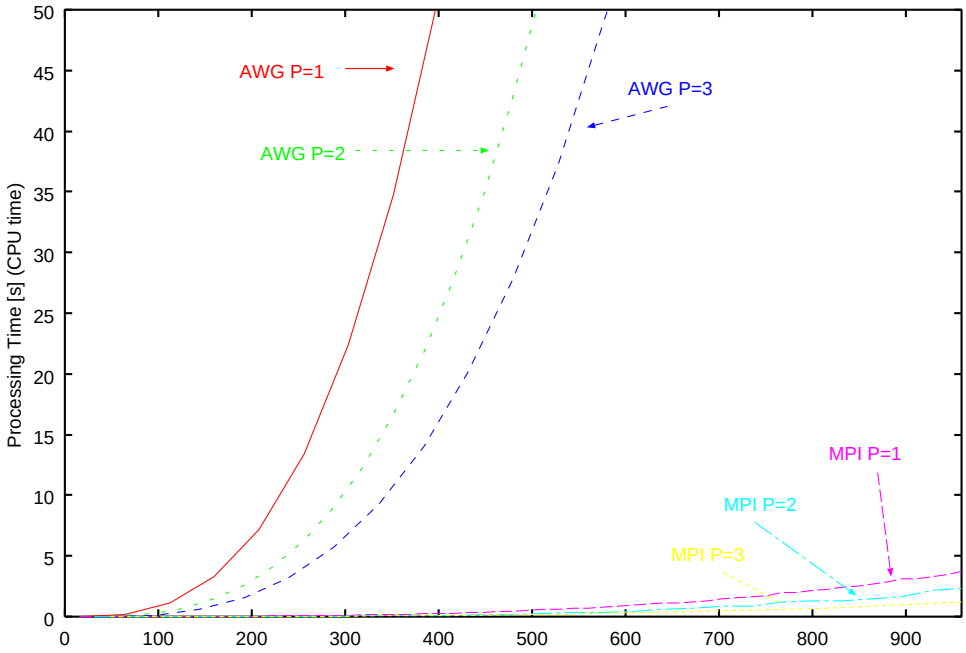

図 14: LU 分解の実行時間（CPU 時間）

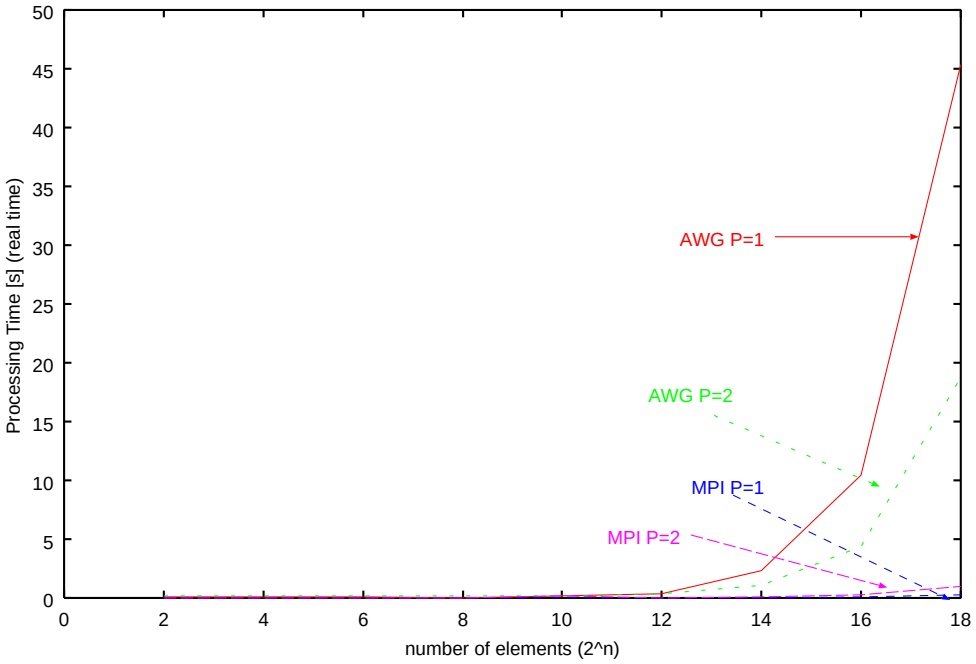

図 15: FFT の実行時間（実時間）

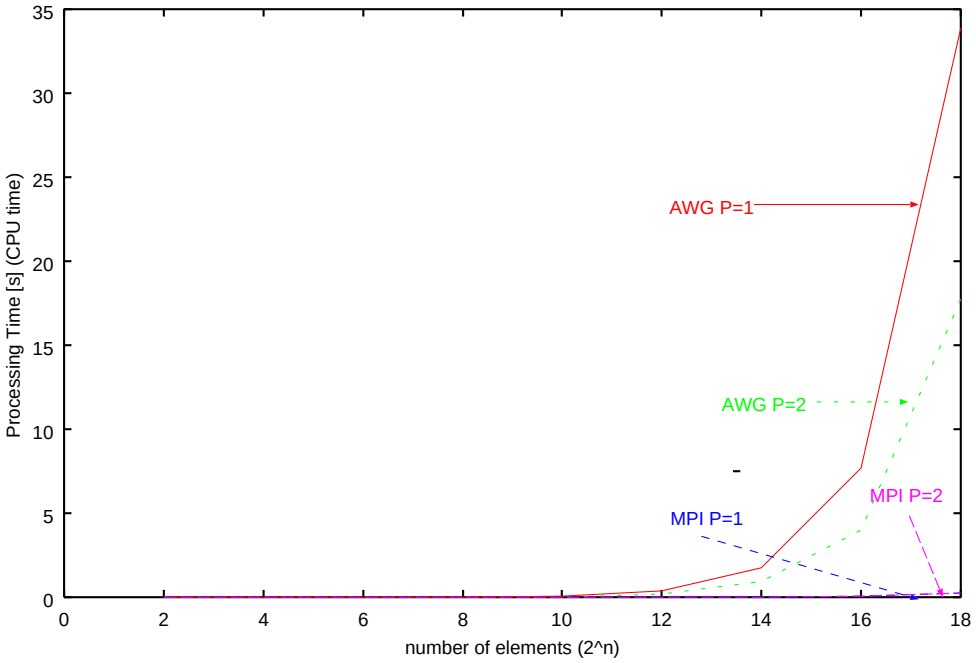

図 16: FFT の実行時間（CPU 時間）

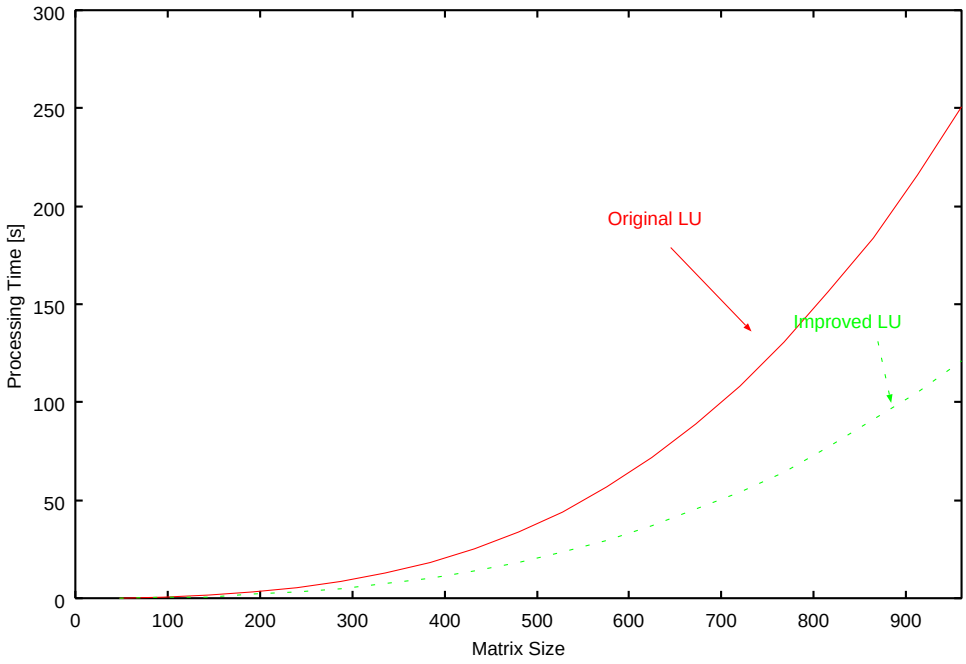

図 17: 共有メモリボードの高速化による改善後の実行時間

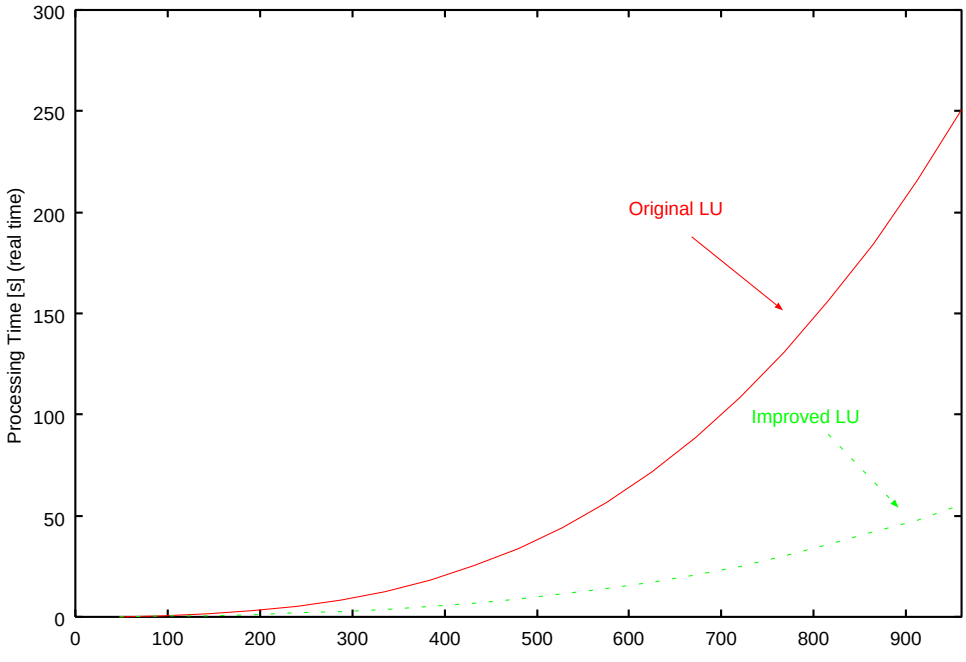

図 18: プログラムのチューニングによる改善後の LU 分解の実行時間（ノード計算機数 3）

5 おわりに

本報告では、λコンピューティング環境として、AWG–STAR を利用した場合の共有メモリシステムの性能の評価を行った。AWG–STAR を用いて光リングを構成し高速な通信チャネルとして利用し、各ノードの共有メモリを分散計算におけるデータ共有手段として用いて分散計算のベンチマークアプリケーションを実際に実行することでその評価を行った。その結果、AWG–STAR のようなモデルの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、共有メモリへの書き込みアクセス回数が性能に影響を与えることがわかった。

今後の課題としては、ノードが広域に分散し伝搬遅延の影響が大きいようなモデルでの評価を行うことが考えられる。またそのような環境でも高速にデータ共有ができるような効率のよい共有メモリのアクセス手法を考案しなければならない。また今回はベンチマークアプリケーションを用いて評価を行ったが、実用的な分散計算を行うアプリケーションを用いた場合の性能評価も課題のひとつである。

謝辞

本報告を終えるにあたり、御指導、御教授を頂いた大阪大学大学院情報科学研究科の宮原秀夫教授に深く感謝致します。また、直接、御指導、御教授頂いた大阪大学サイバーメディアセンターの村田正幸教授に心から感謝致します。また終始、御指導、御助言を頂いた大阪大学サイバーメディアセンターの馬場健一助教授に深く感謝致します。本報告において、多大な御協力を頂いた 日本電信電話株式会社フォトニクス研究所の松岡茂登氏、岡田顕氏、小西邦昭氏に心から御礼を申し上げます。

また日頃から適切なご助言を頂いた大阪大学大学院情報科学研究科の若宮直紀助教授、大崎博之助教授、牧一之進助手、大阪大学サイバーメディアセンターの長谷川剛助教授、大阪大学大学院経済学研究科の荒川伸一助手、大阪府立看護大学の菅野正嗣助教授、大阪市立大学の阿多信吾講師に心から感謝致します。

また、本報告のためにいろいろとお世話して頂いた中本博久氏に厚く御礼を申し上げます。最後に、日頃から御協力を頂いた宮原研究室および村田研究室の皆様に心からお礼申し上げます。

参考文献

[1] T. DeFanti, M. Brown, J. Leigh, O. Yu, E. He, J. Mambretti, D. Lillethun, and J. Weinberger, "Optical Switching Middleware for the OptIPuter," *IEICE Transaction on Communication*, vol. E86-B, Aug 2003.

[2] K. Kato, A. Okada, Y. Sakai, K. Noguchi, T. Sakamoto, S. Suzuki, A. Takahara, S. Kamei, A. Kaneko, and M. Matsuoka, "32×32 full-mesh (1024 path) wavelength-routing WDM network based on uniform-loss cyclic-frequency arrayed-waveguide grating," *Electronics Letters*, vol. 36, pp. 1294–1296, July 2000.

[3] Y. Sakai, K. Noguchi, R. Yoshimura, T. Sakamoto, A. Okada, and M. Matsuoka, "Management system for full-mesh WDM AWG–STAR network," in *27th European Conference on Optical Communication, 2001*, vol. 3, pp. 264–265, Sep 2001.

[4] 日本電信電話株式会社フォトニクス研究所, **情報共有ネットワークシステム説明書**.

[5] 中本博久, "フォトニックグリッド環境における共有メモリアクセス手法," **大阪大学 特別研究報告**, 2003.

[6] S. Cameron, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH–2 Programs: Characterization and Methodological Considerations," in *Proceedings of the 22nd Annual International Symposium on Computer Architecture*, pp. 24–36, June 1995.

[7] 天野英晴, **並列コンピュータ**. 昭晃堂, 1996.

[8] P. パチェコ 著 秋葉博 訳, *MPI* **並列プログラミング**. 培風館, 2001.