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における共有メモリアクセス手法の提案

中本　博久† 　馬場　健一†† 　村田　正幸†

† 大阪大学 大学院情報科学研究科 〒 565-0871 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 茨木市美穂ヶ丘 5-1
E-mail: †{h-nakamt, murata}@ist.osaka-u.ac.jp, ††baba@cmc.osaka-u.ac.jp

あらまし　ネットワークにおける高速かつ大容量な伝送を可能とする技術への要求を満たすために、光伝送技術を用いた研究が活発に進められているが、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非常に難しくなっている。そこで、ネットワークノードや計算機群を光ファイバで接続したフォトニックネットワーク上に仮想チャネルをメッシュ状に張ることにより、Storage Area Network やグリッド計算など新しい応用技術に必要な、高速かつ、高信頼な通信パイプをエンドユーザに提供することができるλコンピューティング環境を提案する。本稿では、フォトニックネットワーク上に仮想リングを構成し、リング上にデータを載せることにより、波長を仮想的な共有メモリとして利用することを考え、この共有メモリに対するアクセス方式を提案し、評価している。その結果、光リングネットワークによる共有メモリが有効であること、特に同期処理が少ないプログラムにおいて並列化効果の高いことがわかった。

キーワード　λコンピューティング環境、光リングネットワーク、共有メモリ、メモリアクセス競合、キャッシュの整合性

Proposal of Shared Memory Access Methods for Lambda Computing Environment

Hirohisa NAKAMOTO†, Ken-ichi BABA††, and Masayuki MURATA†

† Graduate School of Information and Engineering Science, Osaka University, Suita, Osaka 565-0871, Japan
†† Cybermedia Center, Osaka University, Ibaraki, Osaka 567-0047, Japan
E-mail: †{h-nakamt, murata}@ist.osaka-u.ac.jp, ††baba@cmc.osaka-u.ac.jp

Abstract　Optical transmission technology is studied actively in order to realize high-speed transmission and broadband networks. However, conventional packet-based switching technology cannot realize the true high quality communication for each connection. Then we propose λ computing environment which has the virtual channels utilizing optical fibers connecting computing nodes. So we can offer the high-speed and reliable connection path/pipe which is necessary for SAN and Grid computing, to the users with such virtual channels. In this paper, we propose and evaluate an access methods to the virtual ring network which consists of such channels when we use the ring network as a shared memory. As a result, we can show the performance of the proposed method to access the shared memory on photonic networks.

Key words　λ computing environment, phtonic ring network, shared memory, meory access contention, cache coherency

1. はじめに

近年のインターネットをはじめとするネットワークの利用者の増大により、ネットワークを流れるトラヒック量は増大する一方である。特に、映像などを利用したさまざまなアプリケーションが利用されるようになり、ネットワークにおける高速かつ大容量な伝送を可能とする技術への要求はますます高まっている。これらの要求を満たすために、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、光の波長を多重化して利用する WDM (Wavelength Division Multiplexing) 技術が開発の中心であり、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 [1]。さらに、WDM 技術を基盤としてインターネットの高速化を図る、いわゆる IP over WDM ネットワークの研究開発が、現在さかんに進められている。また、それを一歩進めて WDM 技術以外のさまざまなフォトニック技術を下位レイヤの通信技術とした、GMPLS (Generalized 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の標準化も IETF (the Internet Engineering Task Force) で進められている [2]。さらに、フォトニックネットワークの真の IP 化を狙って、フォトニック技術に基づいたフォトニックパケットスイッチに関する研究も始められつつある [3], [4]。

しかし、これらの諸技術は現在のインターネット技術を是としている。すなわち、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れ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非常に困難である。例えば、SAN (Strage Area Network) やグリッド計算など新しい応用技術では、高速かつ、高信頼な通信パイプをエンドユーザに提供する必要があり、そのためには、エンドユーザ間に大容量波長パスを設定し、ユーザに提供することが考えられる。すなわち、既設のファイバを利用し、あるいは必要に応じて、ファイバを新たに敷設し、ファイバおよびファイバ内に多重化された波長を最小粒度として情報の交換を行うフォトニックネットワークを構築することによって、超高速かつ高品質な通信パイプをエンドユーザに提供することが可能である。

光ネットワークを用いた高速な分散計算環境システムを目標とするミドルウェアとして、OptIPuter [5] がある。OputIPuter は広域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目指している。しかしながら、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る。

そこで、本稿では、ネットワークノードや計算機群を光ファイバで接続したグリッド環境において、グリッド上での通信を波長パスを利用して行う新たなアーキテクチャの一つとして λ コンピューティング環境を提案する。従来のグリッド環境においては、TCP/IP を用いたメッセージパッシング (MPI; Message Passing Interface) を利用してデータ交換を実現していたが、λ コンピューティング環境においては、グリッド上の通信を従来の TCP/IP を用いて実現するのではなく、あらかじめ設定した波長パスを利用することにより高速かつ高信頼な通信を実現することができる。すなわち、グリッドを構成するフォトニックネットワーク上にデータ通信用の仮想チャネルをメッシュ状に張ることにより、高速チャネル上での分散計算が可能となる。また、λ コンピューティング環境を構成するネットワークノードおよび計算機群を結ぶ仮想リングを想定し、仮想リング上の波長を高速な共有メモリとして利用することも可能である。その結果、広域分散システムにおける共有メモリと通信チャネルの区別の必要がなくなり、コンピュータ間の高速なデータ交換が可能になる（図 1）。

以上より、本稿では、λ コンピューティング環境を実現する波長パスを用いてデータ交換を行う新たなフォトニックネットワークアーキテクチャを対象に、これらのフォトニックネットワークへのアクセス手法を提案する。具体的には、先に述べた仮想光リングを共有メモリとして利用し、各計算機群のローカルメモリや CPU 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ことを考える。仮想光リング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の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アクセスのタイミングや頻度に制約を受け、通常の共有メモリシステム以上に、仮想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コヒーレンシを十分考慮する必要があるが、広域分散システムにおける共有メモリと通信チャネルの区別の必要がなくなり、計算機間の高速なデータ交換が可能になる。以上、述べたような特徴を考慮し、λ コンピューティング環境を想定した光共有メモリのアクセス手法を提案し、シミュレーションを用いてその性能を明らかにする。

以下、2 章では、従来の共有メモリシステムのキャッシュの整合性、メモリアクセスの競合回避について述べ、3 章では、対象とする λ コンピューティング環境と提案する共有メモリアクセス手法について述べる。4 章では、並列計算のベンチマークプログラムを用いて提案手法の評価を行い、5 章で、まとめと今後の課題について述べる。

2. 従来の共有メモリシステム

共有メモリシステムにおける重要な問題は、複数のプロセッサから共有メモリにアクセスする際に発生する競合を回避することである。すなわち、各プロセッサから同一の共有メモリにアクセスする場合、あるいは同時にプロセッサと共有メモリを結ぶバスなどの伝送路を利用する場合に競合が発生するため、ロック機構などを利用して回避する必要がある [6]〜[8]。

また、それらの競合回避のためのオーバヘッド処理によりメモリアクセス時に制約を受け、プロセッサの性能を十分に引き

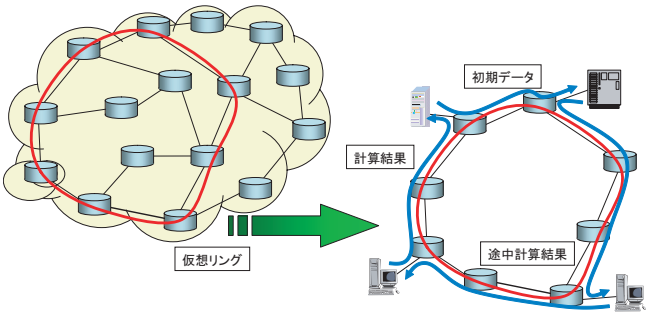

図 1　フォトニックネットワーク上での仮想リング

出せなくなる可能性がある。そこで、高速なキャッシュを用いて共有メモリへのアクセスを減らし、性能向上を図る。その際、キャッシュとメモリの整合性を保つ機構が重要になる。本章では、これらの手法について簡単に述べる。

2.1　共有メモリにおける競合回避

まず、共有メモリにおける競合について述べるため、各プロセッサがキャッシュを持たない場合を想定する。各プロセッサが必要なデータを取得する基本的な手順は、各プロセッサ間で同じデータを参照しても差し支えないため、同一の共有メモリに対する競合は考慮しなくてもよい。しかしながら、共有バス上では競合が生じるため、ロック機構を用いて競合回避を図る。すなわち、取得の制御メッセージを制御バスに転送し、データを取得する。

各プロセッサが処理したデータを書き出す、あるいは書き換える場合の基本的な手順も、共有メモリ上、および共有バス上で競合が発生するため、ロック機構を用いて競合を回避する。すなわち、データを書き込みの制御メッセージを制御バスに転送し、共有バス、共有メモリとも保護し、データを書き込む。

2.2　キャッシュの整合性

各プロセッサから共有メモリへのアクセスは、共有バスにおける競合回避のため、アクセスに制約を受ける。そこで、共有メモリへのアクセスを減らすためにキャッシュを用いた高速化が図られるのが一般的である。通常、プロセッサごとにキャッシュをもち、1 次キャッシュ、2 次キャッシュ、3 次キャッシュと多段に構成される。1 次キャッシュは、プロセッサ内に配置され、容量は小さいが高速に動作する。2 次、3 次キャッシュはプロセッサ外に配置され、1 次キャッシュより容量は少し大きくなるが、プロセッサへの転送速度は 1 次キャッシュに比べ遅くなる。

各プロセッサがキャッシュを持つ場合は、キャッシュ上のデータと共有メモリ上のデータの整合性を十分に考慮する必要がある。一般にキャッシュの整合性を解決する手法にディレクトリ方式とスヌープキャッシュ方式の 2 つの方式がある。本稿では、ディレクトリ方式を採用した場合、ディレクトリテーブルに対するアクセスがボトルネックになる可能性があるので、スヌープキャッシュ方式を採用する。

2.3　スヌープキャッシュ方式

スヌープキャッシュ方式は、キャッシュ間のデータの一致制御の目的で、共有バス上のメモリアクセスを監視し、必要に応じて自キャッシュブロックに対する一貫性制御を分散的手法で行う。プロセッサとメモリ間でデータ交換する際の手順については、各プロセッサがキャッシュを持つ場合、データへのアクセス手法はかなり複雑になる。各プロセッサがデータを取得する基本的な手順は、まず、キャッシュを検索し、キャッシュにデータが存在しない場合にはじめて共有メモリにアクセスする。ただし、取得するデータが他のプロセッサの持つキャッシュになければ、共有メモリから取得すればよいが、他のプロセッサのキャッシュに存在する場合には、対応の仕方によっていくつかの手法が考えられる。

また、各プロセッサが処理したデータを書き出す、あるいは書き換える場合は、さらに複雑になり、キャッシュに書き出す、共有メモリにも書き出す、他のプロセッサのキャッシュに対象となるデータが存在する場合の対応などによって、いくつかの手法がある。

このようなキャッシュ一致プロトコルは、一致させるタイミング（ライトスルー、ライトバック）と方法（無効化、更新）によって 4 つに分類される。そのうち、無効化型ライトバックプロトコルは最も共有メモリアクセスの少ないプロトコルであり、共有メモリへのアクセス遅延が大きい場合に非常に有効である。そこで、以下で無効化型ライトバックプロトコルについて説明する。

2.4　無効化型ライトバックプロトコル

各データは、無効 (I: Invalid)、共有メモリと一致 (C: Clean)、共有メモリと不一致 (D: Dirty) の 3 状態を持つ。複数あるいはひとつのプロセッサがあるアドレスを参照すると、データが共有メモリからキャッシュにコピーされ、複数のプロセッサのキャッシュ上で C 状態になる。C 状態のデータは、共有メモリと内容が一致しているので、このラインに対する読み出しはバス操作を伴わない。ここで、C 状態のデータに対して書き込みを行うと、そのデータは D 状態になる。この時、バス上には無効化を示す信号と、ブロックに対応するアドレスが送出される。他のプロセッサのキャッシュコントローラは、バスを監視し、対応するデータを保持していればそれを無効化する。以後、D 状態のデータに対する読み書きは、バスを介さずに行える。D 状態のデータのアドレスに対して他のプロセッサが読み出し要求を出した場合、D 状態のデータを共有メモリに対して書き戻して一致をとる。次に、要求を出したプロセッサに対して共有バスから転送が行われ、両方のキャッシュは C 状態になる。一方、D 状態のラインのアドレスに対して他のプロセッサが書きこみ要求を出した場合、読み込み同様、D 状態のデータの書き戻しが起こり、次に要求を出したキャッシュに対して転送が行われる。最後に要求を出したプロセッサは自キャッシュ上のデータに書きこみを行い、状態は D 状態になる。もともとデータを保持していたキャッシュは無効化される。

3.　λ コンピューティング環境における光共有メモリとメモリアクセス手法

3.1　対象とするネットワークモデル

次に示すネットワークモデルを対象とする。λ コンピューティ

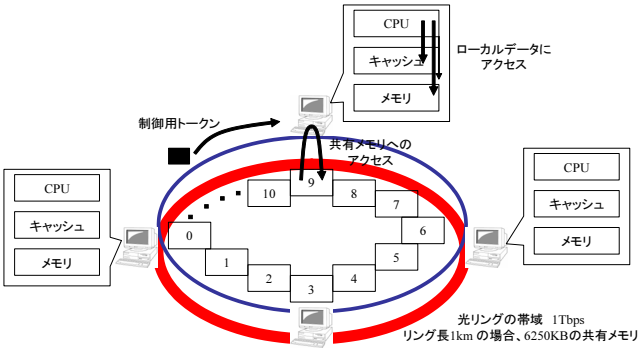

図 2　ネットワークモデル

ング環境を構成するノード計算機群は、光ファイバで接続され、それらの光ファイバにより、仮想的にリングネットワークを構成しているとする。本稿では、各ノード計算機は、1 台の CPU、1 次キャッシュ、ローカルメモリを持つとする。ネットワークモデルの構成を図 2 に示す。光リングネットワークは、波長パスとして、共有メモリ用の波長パスと制御信号用の波長パスを持つ。共有メモリ用の波長パスの帯域は 1Tbps とし、伝搬遅延時間は 5ns/m とする。光リングネットワークを構成するネットワーク機器等の中間ノードでの処理遅延時間はここでは考慮せず、伝搬遅延時間に含まれるものと想定している。従って、光リングネットワークを共有メモリとして用いる場合、例えば、距離 1km で 6250KByte の容量に相当する。

3.2　光共有メモリへのメモリアクセス手法

λ グリッド環境における共有メモリへのアクセスの際にも、前章に示したキャッシュの整合性の問題が生じる。今回、評価に用いるアプリケーションはノード計算機内で集中的に処理を行い、同期処理後、データ交換を行うため、ノード計算機内での処理中はデータ交換を行わない。そこで、共有メモリへの書き戻しが少ない無効化型ライトバックプロトコルを採用し、メモリアクセスのタイミングに制約があるため、その制約とキャッシュの整合性を考慮したプロトコルを 3.2.1 節で提案する。この問題を解決するために、制御用トークンを用意する。また、並列計算機では、プロセッサがお互いに協調動作をするための同期操作が重要である。同期操作の種類としては、不可分命令、同期変数のキャッシング、共有メモリ上での待ち合わせ、メモリロック、バリア同期法などがある。本稿では、共有メモリ上で動作するアプリケーションの性質、光リングネットワークに対するアクセス制約からバリア同期法を採用した。光リング上でのバリア同期法の説明を 3.2.2 節に示す。

3.2.1　光リングネットワークの特性を考慮した無効化型ライトバックプロトコル

光リングネットワークを共有メモリとして利用する際の、無効化型ライトバックプロトコルにおいて、従来手法と異なる点は、制御メッセージを送信する際に制御用トークンを用いるため、制御用トークンが光リングネットワークを 1 周するなどの待ち時間が生じること、リードミス、ライトミスの処理が異なることである。

従来手法においては、リードミスが起きた場合、他ノード計算機が該当データを持っていた場合、他ノード計算機から要求ノードにコピーされる。しかし、光リングネットワークを用いた共有メモリの場合、他ノード計算機から該当データの転送を待つより、共有メモリに直接アクセスする方が制御が容易で遅延時間を短縮できる。そこで、各ノード計算機は、ラインコピー要求が来た場合、該当データが C 状態の場合は応答を返さず、D 状態のときだけ応答を返すようにする。この場合は共有メモリへのアクセスを伴わない。

次に、キャッシュに対する書き込みの処理の場合を考える。データが C 状態でキャッシュに存在し、プロセッサがそのデータを書き出す場合、キャッシュに対してデータを書き出すとそのデータは C 状態から D 状態になる。続いて該当データへの無効化要求メッセージを制御用トークンに付加し、制御用の波長パスに送出する。このとき、該当アドレスのデータを持つ他のノード計算機は該当データの書き込みが起きたことを知り、そのノード計算機の自キャッシュに持つデータを I（無効）状態にする。この際、複数のノード計算機が同時に C 状態の同じアドレスのデータに対する書き込みを行うと、複数のノード計算機が D 状態のデータを保持する可能性がある。この問題を解決するために、ノード計算機がキャッシュを C 状態から D 状態に更新する際は、制御用トークンを獲得し、他ノード計算機が該当アドレスに対して無効化メッセージを付加していないことを確認してから、無効化メッセージを付加し、光リングの制御用の波長に送出し終わった後に該当データのキャッシュを C 状態から D 状態に変更するようにしなければならない。また、制御用トークンを獲得した際に、既に他ノード計算機が該当アドレスに対して無効化メッセージを付加していた場合は該当キャッシュを I 状態にし、キャッシュの更新を他ノード計算機に譲る。

3.2.2　バリア同期

光リングネットワークによる共有メモリにおいて、バリア同期を実現する方法について説明する。まず、同期メモリ用に共有メモリの一部を当てる。従来の手法と同様に、同期メモリへのアクセスは Fetch&Decrement 操作を伴う。つまり、同期メモリへのアクセスの際は、必ず不可分で減算処理が行われるように制御を行う。光リングネットワークを同期メモリに用いる場合、同期メモリに同時にアクセス可能なノード計算機は 1 台に限られるので不可分命令の実行は容易である。アプリケーションプログラム上でノード計算機間での待ち合わせが必要な場合は、各ノードは同期メモリにアクセスする。同期メモリには、プロセッサ数がセットされており、アクセスがあると 1 減算され、全てのノードがアクセスすると 0 になる。同期メモリにアクセスした際に値が 0 でなければ、ノードは他のノードの処理が終わるのを待つ。全てのノードが同期メモリにアクセスし終わると、同期メモリの値が 0 になり、全てのノードは同期処理を終えて、次の処理に移る。

4.　性 能 評 価

本章では、前章で提案した共有メモリアクセス方式をシミュレーションにより評価する。その際、λ コンピューティング環境上での共有メモリを用いて分散計算を行う場合に加え、従来

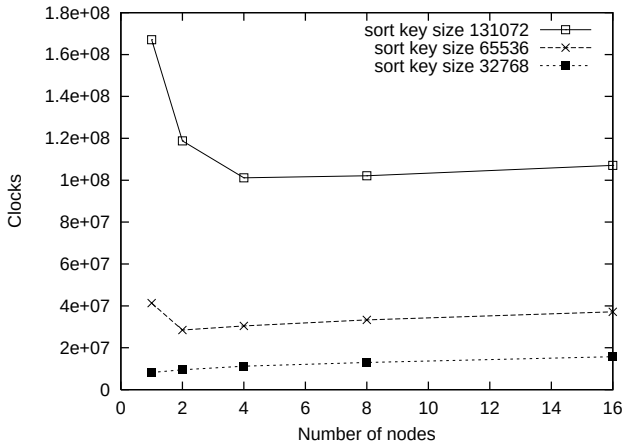

図 3　光共有メモリを用いた基数ソートプログラム実行時間

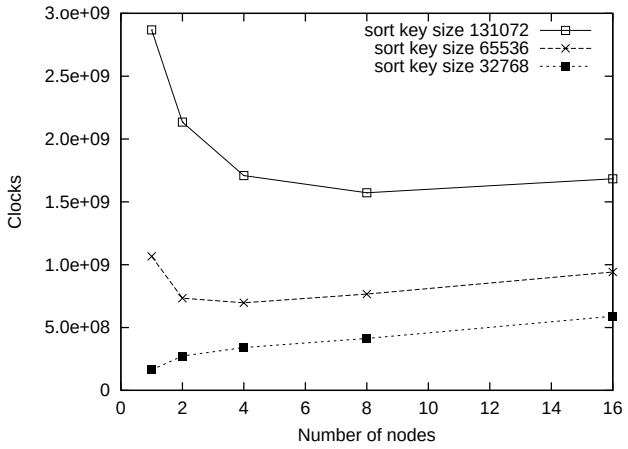

図 4　TCP 転送を用いた基数ソートプログラム実行時間

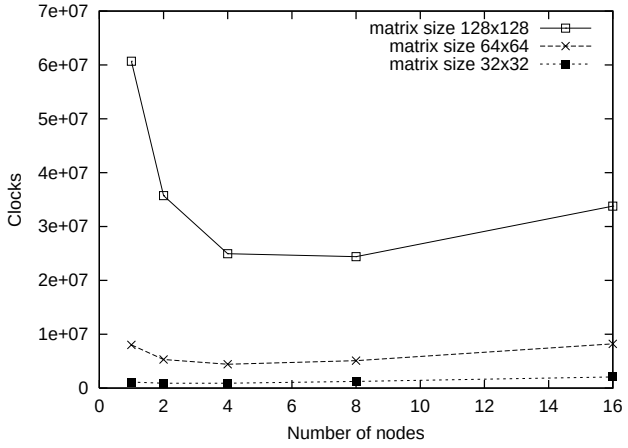

図 5　光共有メモリを用いた行列積プログラム実行時間

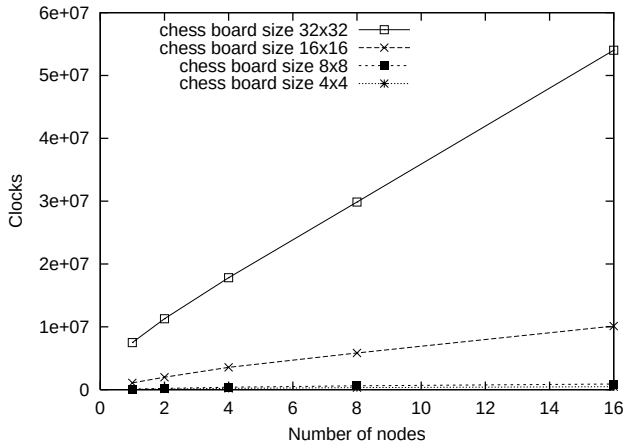

図 6　光共有メモリを用いたクイーン問題プログラム実行時間

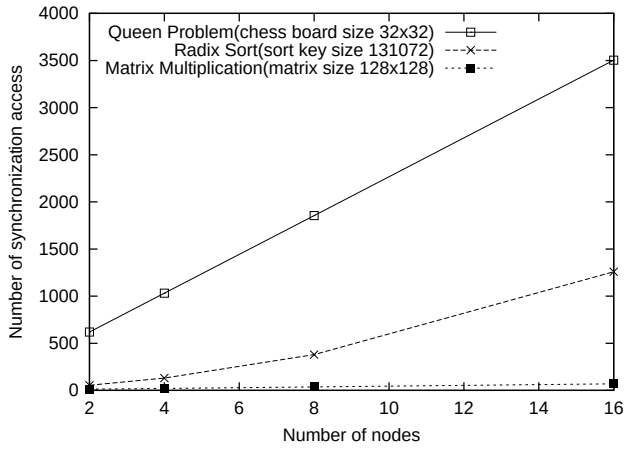

図 7　ベンチマークプログラムごとの同期アクセス回数

の TCP を用いて通信を行って分散計算を行った場合の性能を併せて示す。シミュレーションプログラムの作成の際に、慶應大学の天野研究室で開発している ISIS ライブラリを参考にしている [9]。

4.1　シミュレーションモデル

次に示すネットワークモデルを対象とする。λ コンピューティング環境を構成するノード計算機群は、光ファイバで接続され、それらの光ファイバにより、仮想的にリングネットワークを構成しているとする。各ノード計算機は、1 台の CPU、1 次キャッシュ、ローカルメモリを持つとする。CPU の動作周波数は 3GHz、1 次キャッシュの容量は 512KB、ローカルメモリは 2GByte。各ノード計算機から光リングネットワークへのインターフェースにおける処理遅延はここでは考慮していない。ノード計算機は、光リングネットワークをノード計算機数で割った均等な距離に配置されていることを想定している。光リングネットワークは、波長パスとして、共有メモリ用の波長パスと制御信号用の波長パスを持つ。共有メモリ用の波長パスの帯域は 1Tbps とし、伝搬遅延時間は 5ns/m とする。比較対象として TCP によるデータ交換を用いた共有メモリシステムでは、1 台の共有メモリサーバに共有メモリが存在するものと考え、ノード計算機群は、Ethernet で共有メモリサーバと接続されているとする。ノード計算機の性能は、光共有メモリモデルの場合と同様であり、各ノード計算機から共有メモリサーバへの距離は 1km とする。Ethernet の転送速度は 1Gbps とする。光リングネットワークを構成するネットワーク機器等の中間ノードでの処理遅延時間はここでは考慮せず、伝搬遅延時間に含まれるものと想定している。

また、シミュレーションの評価には、ベンチマークプログラムである SPLASH2 アプリケーション集から、整数値の列を

基数ソートするプログラム (基数ソートプログラム)、乱数を要素とする $n \times n$ の行列積を求めるプログラム (行列積プログラム)、n-Queen 問題を解くプログラム (クイーン問題プログラム) を用いた。それぞれのプログラムは慶応義塾大学の天野研究室で用いられているプログラムを基に扱える問題サイズの最大値を改変している。

4.2 基数ソートプログラムによる実行結果

図 3 に、λ コンピューティング環境を想定した共有メモリシミュレータ上で、基数ソートプログラムを実行した場合の CPU における実行クロック数を示す。ソート対象のキーの数は、32768、65536、131072 個である。問題サイズ 32768 の場合は、ノード計算機数を増やしても、ローカルでの処理が少なく、同期処理が増加するので、並列化の優位性は見られない。しかしながら、問題サイズが 65536 や 131072 のように大きくなると、ノード計算機数が 8 程度までは並列化の効果が出ていることがわかる。しかし、ノード数が増えていくにつれて並列化の効果は弱まる。また、図 4 に、TCP を用いた場合の共有メモリシミュレータ上で、基数ソートプログラムを実行した場合の CPU における実行クロック数を示す。λ コンピューティング環境上での共有メモリシミュレータと同様の傾向を示すが、全体として λ コンピューティング環境上での共有メモリシミュレータより実行クロック数が大きくなっている。この結果から、十分な問題サイズに対して分散処理を行う際に、λ コンピューティング環境を想定した共有メモリ方式の方が TCP を用いたデータ交換方式より性能が良いことがわかった。

4.3 行列積プログラムによる実行結果

図 5 に、λ コンピューティング環境を想定した共有メモリシミュレータ上で、行列積プログラムを実行した場合の実行クロック数を示す。行列のサイズは 32×32 から 256×256 である。基数ソートプログラムによる結果と同様に、問題サイズが小さい場合は、ノード計算機数を増やす効果はないが、問題サイズが大きくなると、ノード計算機数が 8 程度までは並列化の効果が出ていることがわかる。TCP を用いた場合の共有メモリシミュレータ上で、行列積プログラムを実行した場合の実行クロック数を示す。紙面の都合上、掲載していないが、TCP を用いた場合の共有メモリシミュレータは、問題サイズを増やしても並列化の効果が出ておらず、全体として λ コンピューティング環境上での共有メモリシミュレータより実行クロック数が大きくなった。

4.4 クイーン問題プログラムによる実行結果

図 6 に、λ コンピューティング環境を想定した共有メモリシミュレータ上で、クイーン問題プログラムを実行した場合の実行クロック数を示す。問題サイズは 4×4 から 32×32 である。クイーン問題の場合は問題サイズを大きくしても、並列化の効果がない。これは、クイーン問題の同期処理回数が他のアプリケーションと比べて多いためである。

4.5 同期アクセス回数の実行クロック数への影響

図 7 に、λ コンピューティング環境を想定した共有メモリシミュレータ上で、基数ソートプログラム、行列積プログラム、クイーン問題プログラムを実行した場合のノード計算機間の同期回数（同期メモリへのアクセス回数）を示す。基数ソートプログラムはソート対象のキー数が 131072 個、行列積プログラムは行列サイズが 128×128、クイーン問題プログラムはチェス盤のサイズが 32×32 である。図より、クイーン問題プログラムの同期メモリアクセス回数が基数ソートプログラム、行列積プログラムと比べてかなり多いことがわかる。従って、基数ソートプログラム、行列積プログラムの場合は、同期処理が並列化に与える影響は小さいが、クイーン問題プログラムでは、その影響が大きいと考えられる。その結果が図 3, 5, 6 に示した並列化効果の有無に現れている。

5. ま と め

本稿では、 フォトニックネットワーク上で共有メモリを実現した際の、共有メモリアクセス手法を提案した。また、計算機上でシミュレーションを行い、並列計算用のベンチマークプログラムを用いて評価を行った。その結果、光共有メモリの有効性と同期処理が少ない場合のノード数の増加による並列計算の有効性を示した。今後は、効率の良い共有メモリアクセス方式、ならびにローカルメモリの活用方法を検討していく予定である。さらに、今回のシミュレーションでは、インターフェースでの処理遅延や、複数のノード計算機から光リングネットワークへの同時アクセスなどを考慮していないので、改善すべき課題である。また、我々は、光リングネットワークを高速チャネルとして利用した場合の共有メモリシステムとメモリアクセス手法を検討しており、今後、報告する予定である。

謝　　辞

シミュレータプログラム作成の際に、慶應義塾大学理工学研究科の若林正樹様 (現ソニー株式会社) に助言をいただいた。ここに記して謝意を表す。

文　　献

[1] M. Murata and K. Kitayama: “Ultrafast photonic label switch for asynchronous packets of variable length”, IEEE INFOCOM 2002 (2002).
[2] E. L. Berger: “Generalized multi-protocol label switching (GMPLS) signaling functional description”, IETF RFC3471 (2003).
[3] 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).
[4] 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).
[5] 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, **E86-B**, 8 (2003).
[6] 天野英晴：“並列コンピュータ”, 昭晃堂 (1996).
[7] 鈴木則久, 清水茂則, 山内長承：“共有記憶型並列システムの実際”, コロナ社 (1993).
[8] 富田眞治：“並列コンピュータ工学”, 昭晃堂 (1996).
[9] 若林正樹, 天野英晴：“並列計算機シミュレータの構築支援環境”, 電子情報通信学会論文誌, **J84-D-I**, 3, pp. 1–10 (2001).