特別研究報告

題目

光リングネットワークにおけるλコンピューティング環境に適した共有メモリアーキテクチャの評価

指導教員
村田 正幸 教授

報告者
久保 貴司

平成 19 年 2 月 20 日

大阪大学 基礎工学部 情報科学科

平成 18 年度 特別研究報告

光リングネットワークにおける λ コンピューティング環境に適した 共有メモリアーキテクチャの評価

久保 貴司

内容梗概

一台の計算機では実行できないような大規模計算を複数の計算機資源を利用して計算するグリッドコンピューティング技術に関する研究・開発がさかんに進められている．グリッドコンピューティング環境では計算に使用するデータ量が大きいことが多く，現在の TCP/IP によるインターネットではパケット処理などの通信に要するオーバヘッドが大きく十分な性能を得るのは難しい．そこで，ネットワーク上のノードや計算機群を光ファイバで接続し，エンドホスト間に光波長パスを設定することにより高速かつ高品質な通信パイプを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している．λ コンピューティング環境における共有メモリアーキテクチャでは，従来のマルチプロセッサシステムとは異なり計算機が広域に展開しているため，ネットワーク特性がその性能に大きな影響を与える．

以前の研究において，λ コンピューティング環境における共有メモリアーキテクチャのモデル化を行い，ネットワークやキャッシュ一貫性制御のための処理がシステムの性能にどのような影響をあたえるかを解明し，どのような共有メモリアーキテクチャが，λ コンピューティング環境に適しているかについて報告されている．この報告において対象とされたネットワークモデルは単一波長のリングモデルとフルメッシュモデルである．単一波長モデルにおいては，各ノード計算機での処理時間等の影響が大きくなり，フルメッシュモデルではハードウェアの制約上，実現が難しいと考えられる．

そこで，本報告では，λ コンピューティング環境において波長数を考慮したリングネットワークにおける共有メモリアーキテクチャを提案し，その性能を評価する．具体的には，複数波長を用いてデータ転送やキャッシュ一貫性制御を行うリングネットワークの設計を行い，制御にかかる遅延時間を求め，セミ・マルコフ過程を用いて解析を行った．その結果，共有メモリアクセス頻度が大きい場合など性能向上が得られるパラメータ領域が存在することを明らかにした．

主な用語

λ コンピューティング環境，共有メモリアーキテクチャ，キャッシュ一貫性制御，セミ・マルコフ過程

目 次

図 目 次

表 目 次

1　はじめに

近年，遺伝子解析，画像処理，地球環境シミュレーションなど，大量のデータ処理が必要となる分野が広がり，大規模・高速計算に対する要求が高まっている．こうした大規模なデータを扱う計算は，通常の 1 台の計算機では実用的な時間で実行することが困難であるため，グリッドコンピューティング技術がさかんに研究されている．グリッドコンピューティング環境では，ネットワークに接続されたノード計算機の CPU やストレージなどの計算資源を共有し，計算処理を多数のノード計算機に分散させて並列処理することにより，計算時間の短縮が期待できる．現在，多くのグリッドコンピューティング環境では，ノード計算機へのジョブ投入やデータ転送などに TCP/IP プロトコルを使用している．TCP/IP を用いると，既存のネットワーク技術の上に容易にグリッドコンピューティング環境を構築できる．しかしながら，TCP/IP をグリッドコンピューティング環境へ応用する場合，データ通信の過程で様々な弊害も生じる．例えば，グリッドコンピューティングではノード計算機間で送受信されるデータ量が膨大であるため，送信元から送信先へ至る経路途中で輻輳が発生し，パケットが廃棄されてしまうことがある．このパケットの廃棄に伴い，パケットを再送することによるスループットの低下，あるいは通信経路を共有する他のネットワークサービスやネットワークユーザに対してサービスの品質低下を招く．また，TCP/IP では，ルーティングやパケットのフラグメンテーションなど，パケット処理による遅延も大きい．これら TCP/IP を使用する場合のマイナス面が，グリッドコンピューティング技術による高性能計算環境の実現を困難にしている．このため，ネットワークにおける高速かつ大容量なデータ転送を可能とする新しい技術が必要とされている．

現在，高速ネットワークを構築する技術として，光伝送技術を用いる研究が活発に進められている．特に，光の波長を多重化することで帯域を増大させる WDM (Wavelength Division Multiplexing) 技術が開発の中心である．さらに，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進んでいる．しかし，これらの技術では，情報を扱う最小粒度は IP パケットであり，ネットワーク上でそれをいかにして高速で転送するかを研究開発の目標としている．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かぎり，ノード計算機間で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．また，大容量データ転送を可能とするフォトニックネットワーク上でグリッドコンピューティングなどの分散計算を効率よく行うには，既存のネットワークアーキテクチャとは異なるアーキテクチャが必要である．

パケットにかわり，光ファイバおよび光ファイバ内に多重化された波長を最小粒度として情報交換を行うことにより，高速，かつ高品質な通信を実現することが可能であると考えら

れる．すなわち，既設の光ファイバを利用し，あるいは必要に応じて光ファイバを新たに敷設し，エンドユーザ間に光ファイバによる大容量波長パスを設定すれば，高速、かつ高品質なフォトニックネットワークを構築することが可能となる．そこで，我々は，ノード計算機群やネットワーク上のスイッチを光ファイバで相互接続することでエンドノード計算機間に情報伝送専用の波長パスを提供し，この波長パスを利用して分散計算を行う新たなアーキテクチャとしてλコンピューティング環境を提案している．従来の TCP/IP を用いたグリッドコンピューティング環境とは異なり，あらかじめ設定した波長パスを専用の通信チャネルとして利用することにより，分散並列計算の際のデータ交換において高速・高品質な通信が実現可能となる．

我々はこのλコンピューティング環境上に分散共有メモリシステムを構築する研究を行っている．グリッドコンピューティング環境上でデータ共有を行うモデルとしてはメッセージパッシングと共有メモリの 2 種類があげられる．前者は，各ノード計算機上のアプリケーションプログラムが明示的にメッセージ通信を行うことによりデータを共有するモデルであり，後者はノード計算機間で同一の内容が保持された共有メモリを利用してデータ共有を行うモデルである．前者の方法では，プログラマがノード計算機間でのメッセージの交換を最適化することによって高い計算性能を達成できる可能性があるが，アプリケーションプログラムの開発が容易ではない．一方，後者の方法は，グリッドコンピューティング環境上で共有メモリを実現する分散共有メモリと呼ばれるシステムによってノード計算機の通信が隠蔽されるため容易にアプリケーションの開発を開発できるが，データ共有を自動化するための複雑なプロトコルなどにより通信量が増加し，一般的にグリッドコンピューティング環境上では十分な性能を達成できないと考えられてきた．しかしながら，我々の提案するλコンピューティング環境では，波長パスを専用の通信チャネルとして利用することにより高速・高品質なネットワークを実現しており，高性能な分散共有メモリシステムを構築できる可能性がある．

関連研究 [1, 2] では，λコンピューティング環境を実現する一つの手法として，NTT フォトニクス研究所が開発した情報共有システム（AWG-STAR システム） [3, 4] を利用している．AWG-STAR システムでは，各ノード計算機が共有メモリボードを有し，共有メモリボード間を AWG ルータを介して波長パスで接続した上で共有すべきデータを転送し，すべてのノード計算機間で同一のデータを共有する．これらの研究では，AWG-STAR システムを用いたλコンピューティング環境において，MPI や OpenMP の設計と実装を行い，アプリケーションを用いてその性能を評価している．しかしながら，AWG-STAR システムによる共有メモリアーキテクチャにおいては，並列計算アプリケーションの実行時間に基づいて評価を行っているため，ネットワーク特性やキャッシュプロトコルの違いによる性能への影響などについては十分な評価ができていない．

λコンピューティング環境上の共有メモリでは，ノード計算機が広域に展開しているため，マルチプロセッサシステムやクラスタにおける分散共有メモリなどに比べ，共有メモリシステムを用いた計算環境の性能にネットワーク特性が大きな影響を与えるものと考えられる．このため，λコンピューティング環境に適した共有メモリアーキテクチャを設計するには，ネットワークモデルが共有メモリの性能に与える影響を検討しなければならない．また，現在の計算機アーキテクチャでは CPU の計算性能にプロセッサのキャッシュメモリが大きな影響を与える．このキャッシュメモリの制御方式と共有メモリシステムの相互作用についても検討の必要がある．

これらの課題に取り組んだ文献に [5] がある．文献 [5] では，λコンピューティング環境における共有メモリアーキテクチャのモデル化を行い，ネットワーク特性やキャッシュ一貫性制御のための処理がシステムの性能にどのような影響を与えるかを解明し，どのような共有メモリアーキテクチャが，λコンピューティング環境に適しているかについて報告されている．そこでは，共有メモリアーキテクチャのネットワークモデルとして，単一波長のリングモデルとフルメッシュモデルを取り上げている．しかしながら，単一波長モデルでは，各ノード計算機での処理時間等の影響が大きくなることが明らかになっている．また，フルメッシュモデルでは，性能は改善されるものの，使用する波長数が多いためハードウェアに実装するのは難しいと考えられる．

そこで，本報告では，λコンピューティング環境において波長数やハードウェアの制約を考慮した実現可能なリングネットワークにおける共有メモリアーキテクチャを提案し，その性能を評価する．具体的には，リングネットワークにおいて複数波長を用いてデータ転送やキャッシュ一貫性制御を行う方式の設計を行い，制御にかかる遅延時間を求め，セミマルコフモデルを用いて解析を行う．

2　λコンピューティング環境における共有メモリアーキテクチャ

本章では，λコンピューティング環境について説明し，次に文献 [5] で用いた共有メモリアーキテクチャとその解析結果について述べる．最後に本報告で用いる共有メモリアーキテクチャについて説明する．

2.1　λコンピューティング環境

λコンピューティング環境においては，WDM 技術を利用して各ノード計算機，光スイッチを光ファイバで接続し，ノード計算機間に波長パスを設定する．このノード計算機間に設定した波長パスを利用して分散並列計算を行う．すなわち，従来の TC P/IP を用いたグリッドコンピューティング環境とは異なり，あらかじめ設定した波長パスを専用の通信チャンネルとして利用することにより，分散並列計算のデータ交換において，高速かつ高信頼な通信が実現できる（図 1）．

具体的には，各ノード計算機に搭載された共有メモリを，これらの高速な通信チャンネルで接続する．そのため，従来は密接続であった共有メモリシステムを広域なネットワーク上に展開することが可能となる．

次に，あるノード計算機で更新されたデータが他のノード計算機に反映される方法について説明する．λコンピューティング環境では波長パスを用いて通信を行っている．波長パスの設定によって論理リングトポロジを構成することができる（図 4）．あるノードがデータを更新すると，更新されたデータはこのリングネットワークに送信される．更新されたデータは他のノード計算機に波長パスを用いて送信され，他のノード計算機のデータが更新される．更新すべきデータは，順次，転送され，これによりすべてのノード計算機は更新データを受け取ることができる．したがって，λコンピューティング環境ではリング上に更新データを送信することによってノード計算機が計算中にデータの更新を行うことができる．

一方，光ファイバで多重送信された波長パスを適切な波長に切り替えることにより動的なメッシュトポロジを構成することもできる．また，メッシュトポロジでは各ノード計算機がデータ送信とマルチキャストを行うために個々に波長パスを設定する必要がある．この場合，各ノード計算機はノード計算機間で直接通信を行い，データを更新する．

2.2　従来の共有メモリアーキテクチャの特性

本節では，共有メモリアーキテクチャの特性について説明した後に文献 [5] で設計された共有メモリアーキテクチャ，ならびに評価結果について説明する．

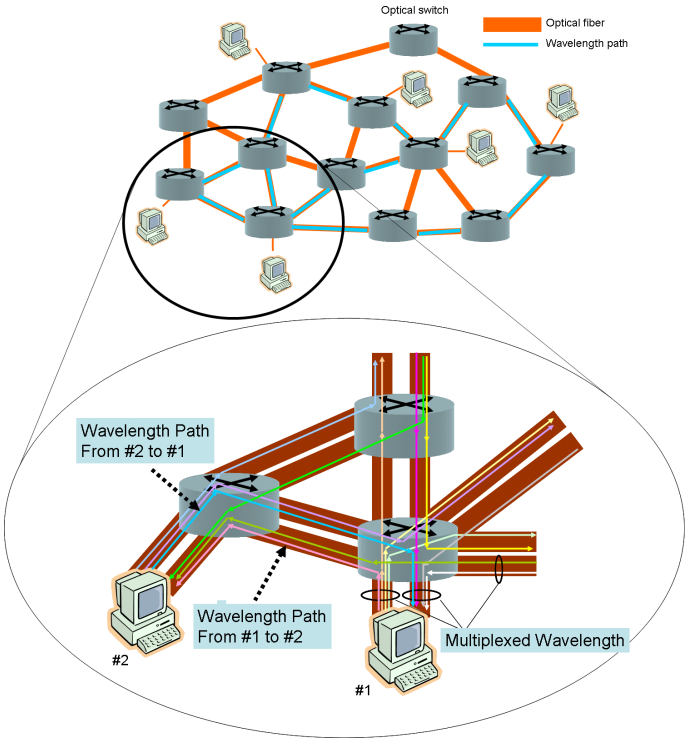


図 1: 波長パスの設定

2.2.1 共有メモリアーキテクチャの特性

λコンピューティング環境における共有メモリアーキテクチャはネットワークトポロジやメモリアクセスモデル，キャッシュプロトコルなどにより性能を大きく左右される可能性があり，どのような共有メモリアーキテクチャが適しているのかは，一概に決定することはできない．

ネットワークトポロジ

λコンピューティング環境において，ネットワークトポロジはノード計算機間の処理遅延に影響を与える．さらに，データ送信のための波長パスの設定方法や使用する波長数の問題がある．

- リングトポロジ
  ブロードキャストが容易となるトポロジであるが，伝搬遅延としてリング 1 周分を最低でも要することになる，また，各ノード計算機は単一方向にしかデータを送信することができない．

- メッシュトポロジ

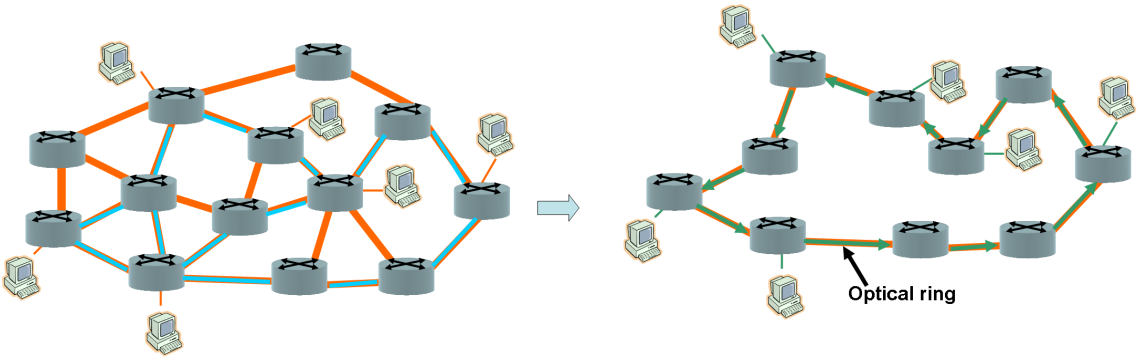


図 2: 論理リングトポロジ

リングトポロジに比べると伝搬遅延は短くなるが，ブロードキャストのために各ノード計算機で送受信されるデータの複製が必要となる，また，リングトポロジに比べ多くの波長が必要とする．

メモリアクセスモデル

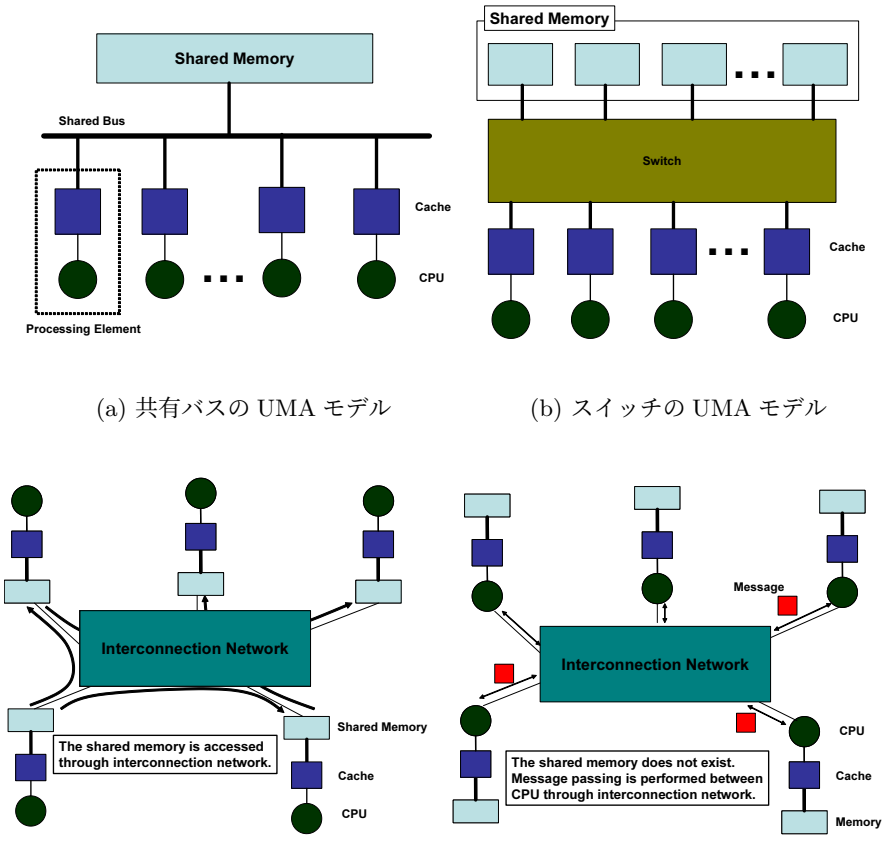
メモリアクセスモデルは，ノード計算機が共有メモリへのアクセス方法を提供する．メモリアクセスモデルは，ノード計算機が直接共有メモリにアクセスできるか否かを決定する．ノード計算機がネットワークと通して共有メモリへアクセスする場合，ネットワークを介さずに共有メモリへアクセスする場合に比べて多くの時間がかかる．メモリアクセスモデルは通常 3 種類に分けられる．

- UMA (Uniform Memory Access) モデル
  すべてのプロセッサがアドレス空間を共有し，同一時間でアクセス可能なモデルまたはそのようなメモリを持つ計算機（図 3(a)，図 3(b)）.

- NUMA (Non-Uniform Memory Access) モデル
  すべてのプロセッサが，アドレス空間を共有するメモリを持つが，あるプロセッサから見た時のアクセス速度は，メモリの番地によって異なるモデルまたはそのようなメモリを持つ計算機 (図 3(c)).

- NORMA (NO Remote Memory Access) モデル
  各プロセッサは互いに独立したアドレス空間のメモリを持ち，メッセージのやりとりによって計算を進めていく，つまり共有メモリをもたないモデルまたは計算機 (図 3(d)).

λ コンピューティング環境において，UMA モデルでは読み込みアクセスの場合にはネットワークを使用しないが，書き込みアクセスの場合は他のノード計算機の共有メモリを更新



(a) 共有バスの UMA モデル　　(b) スイッチの UMA モデル

(c) NUMA モデル　　(d) NORMA モデル

図 3: メモリアクセスモデル

するためにネットワークを利用する必要がある．また NUMA モデルでは他のノード計算機の共有メモリに読み込み，書き込みアクセスを行う場合にはネットワークを利用する必要がある．NORMA モデルでは，共有メモリが存在せず，ネットワークにメッセージを送信するによってデータの交換を行う．

共有メモリの一貫性プロトコル

キャッシュ一貫性プロトコルを実現する方法は 2 つあり，それはスヌープ法とディレクトリ法である．

- スヌープ法
  マルチプロセッサシステムで多用されており，各キャッシュが共有メディアを監視することにより，データや制御信号の発生などシステムの挙動を把握することでキャッ

シュ一貫性プロトコルを実現している．

- ディレクトリ法
  NUMA 型システムで使用されており，どのキャッシュがどのキャッシュラインを保持しているかという情報を各ノードのディレクトリに保持し，必要が生じた時にそれを検索することで直接相手のキャッシュの無効化などの処理を行う．

またキャッシュと共有メモリ間の一貫性プロトコルを行うキャッシュ一貫性プロトコルとしては，キャッシュを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，

- 無効化型ライトスルー
  書き込みが行われるたびに，共有メモリの更新を行い，また同一キャッシュラインを保持する各 CPU のキャッシュメモリではキャッシュラインを無効化することでキャッシュの内容の一致性を保証する，制御が容易である一方で，書き込みの度に共有バスを使用するため，共有バスの混雑が激しくなり一般的にライトバック型プロトコルよりも性能が低下する，接続可能なプロセッサ数の制限も厳しい．

- 更新型ライトスルー
  書き込みが行われるたびに，共有メモリの更新を行い，また同一キャッシュラインを保持する他の CPU）のキャッシュメモリ上の該当キャッシュラインの更新を行いキャッシュラインの内容の一致をとる，ただし，ライトスルー型プロトコルのため，無効化型ライトスルーと同様の問題がある．

- 無効化型ライトバック
  書き込みが行われると，書き込まれたキャッシュラインと同一のコピーを持つ他のキャッシュメモリ上の該当キャッシュラインを無効化することで，キャッシュの一貫性を保持している，書き込み後，直ちに共有メモリに書き込まないため，最新のデータは書き込んだ CPU のキャッシュメモリ上にしか存在しない，他のキャッシュメモリが読み出し要求を発行した場合は，このキャッシュライン内のデータに対してリードミスとなるため，最新キャッシュラインもつ CPU がキャッシュラインを共有メモリに書き戻してから，要求元の CPU は共有メモリからキャッシュメモリにコピーするか，もしくは最新キャッシュブロックをもつ CPU のキャッシュメモリから直接コピーするかのどちらかが取り得る方法である．

- 更新型ライトバック
  書き込みが行われると，書き込まれたキャッシュラインと同一のコピーを持つ他のキャッ

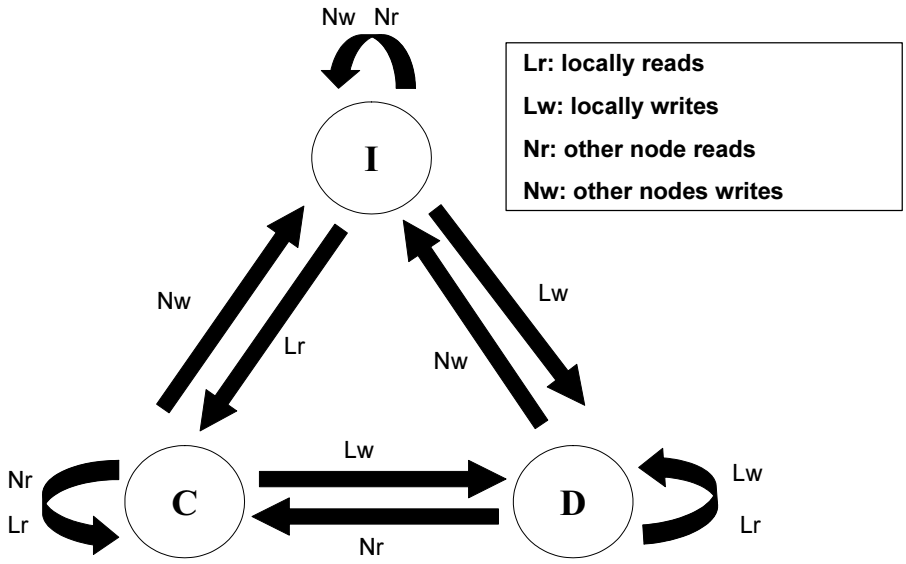


図 4: 無効化型ライトバックの状態遷移図

シュメモリ上の該当キャッシュラインを更新することで，キャッシュの一貫性を保持している，書き込み後，直ちに共有メモリに書き込まず，複数のキャッシュメモリが最新の状態を保持することになる，他のキャッシュのリードミス，ライトミスの際に最新状態のキャッシュラインを保持するキャッシュメモリからコピーするが，最新状態のキャッシュラインを保持するキャッシュメモリが複数個存在するため，どのキャッシュからコピーをするかをプロトコルに含める必要がある．

ここでは，本報告で用いる無効化型ライトバックの状態と状態遷移図 4 を示す．無効化型ライトバックには 3 個の状態があり，それぞれキャッシュと共有メモリとの間でデータが一致している Clean 状態（以下，C 状態），キャッシュと共有メモリとの間でデータが一致していない Dirty 状態（以下，D 状態），キャッシュが無効化されている Invalid 状態（以下，I 状態）である．

2.2.2 従来の共有メモリアーキテクチャの構成とその評価

文献 [5] で設計された共有メモリアーキテクチャの構成を表 1 に示す．

共有メモリアーキテクチャにおいて重要なことは各プロセッサ間のキャッシュの一貫性と，キャッシュと共有メモリ間のデータの一致である．また，並列計算アプリケーションの観点ではプロセス間の同期も重要な点である．このようなデータの一貫性や同期の際にはブロー

表 1: 共有メモリアーキテクチャの構成

ネットワークトポロジ	メモリアクセスモデル	キャッシュプロトコル
リングトポロジ	UMA モデル	スヌープ法
リングトポロジ	NUMA モデル	スヌープ法
メッシュトポロジ	NUMA モデル	ディレクトリ法

ドキャストが多用される．そこで文献 [5] においては，ブロードキャストが容易となるリングトポロジを構成する共有メモリアーキテクチャについての解析が行われた．このモデルではリングトポロジを構成しているため，制御トークン用の波長を用意し，これを監視するスヌープキャッシュプロトコルが採用されている．またキャッシュ一貫性プロトコルのうち，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有効である．

スヌープキャッシュプロトコルを用いるこれら 2 つのアーキテクチャは，トポロジとしてリング型を採っているために，伝搬遅延としてリング 1 周分を最低でも要する．そのため伝搬遅延の影響を低くすることができるメッシュトポロジをとる共有メモリアーキテクチャについても解析が行われた．ただしこの場合，ブロードキャストを行うには各ノードにおいてフレームの複製が必要となり負荷が増大する．そのため，ブロードキャストではなく特定のノードのキャッシュおよび共有メモリに対してのみ処理を行うことが可能となる NUMA 型のメモリアクセスモデルが採用されている．また，メッシュトポロジを構成しているためにリング型における制御用の波長の代替となる手法を用意することが困難である．そのためこの場合は，キャッシュプロトコルはディレクトリ法を用いている．

共有メモリアーキテクチャの評価はネットワーク利用率，平均メモリアクセス時間，スループットを指標として行われている．リングトポロジについて，ネットワーク利用率はメッシュトポロジより高くなっている．これはリングトポロジにおいてデータはリングを 1 周する必要があり，そのためデータの伝播に時間がかかるためである．平均メモリアクセス時間についてはリング長が短い場合は高速なアクセスが可能である．一方リング長が長くなればデータ通信の遅延時間が大きくなり高速なアクセスが困難になり平均アクセス時間は長くなる．スループットについては共有メモリへのアクセス率が高くなるとスループットが低下する．これは先に述べたようにデータ通信の遅延時間が大きいことが原因となっている．また共有メモリのキャッシュヒット率が高い場合，スループットは高いが，キャッシュヒット率が低い場合，スループットは大きく減少する．解析結果としては UMA 型と NUMA 型では同様のスループットの結果が出ている．しかし，実際のキャッシュヒット率を考えると，

NUMA 型はキャッシュヒット率がそれほど低くなることがないため，スループットが大きく減少することはない．

メッシュトポロジについては高速通信の行える結果となっている．ネットワーク利用率については，リングトポロジに比べて低い結果となっているが，これは，データ交換が必要なノード間のみの通信が行われているためであり，また，データ通信の遅延時間が小さいためでもある．平均メモリアクセス時間については，CPU が光リングを回る遅延時間を待つ必要がないため高速な値が出ている．またスループットについてはノード数が増えた場合についてもメッシュ構造を採ることにより，高い値が得られている．

2.3 対象とする共有メモリアーキテクチャ

文献 [5] の共有メモリアーキテクチャではネットワークトポロジとして 1 波長リングトポロジとフルメッシュトポロジが用いられている．しかし，これらのネットワークトポロジには以下で述べるような問題点が挙げられる．共有メモリアーキテクチャはノード間のデータ共有に要する時間によって大きく性能が左右される．単一波長リングトポロジは，トポロジがリング型を採っているために，伝播遅延としてリング一周分を最低でも要する．そのためノード間のデータ共有に多くの時間を要し，共有メモリアーキテクチャの性能が低下している．また，フルメッシュトポロジではノード間のデータ共有に要する時間を低く抑えることができる．しかし，フルメッシュでノード間に波長を設定しているため，ノード数が増加すると必要となる波長数が多くなる．現在のネットワークにおいて大規模な波長数を設定することは困難である．そのため多数のノード数を扱う共有メモリアーキテクチャではフルメッシュトポロジを用いることが難しい．

そこで，本報告ではこれらの問題点を踏まえた，実現可能で高性能な共有メモリアーキテクチャの構成方法を提案し，その性能を評価する．リングトポロジではノード数が増加した場合にリング長が長くなるため，データ共有における遅延時間も長くなり，共有メモリアーキテクチャの性能が低下することがわかっている．この遅延時間には 2 つの種類が存在する．ひとつは伝播遅延であり，もう一方はノードによる処理遅延である．リングトポロジでは全てのノードが連なり 1 つのリングを形成している．そのためリングトポロジをとる限り 1 周分のデータの伝播遅延を短くすることはできない．一方，ノードによる処理遅延は，ホップ数の小さな波長を利用することで減らすことが可能となる．したがって，波長を追加することによりノードでの処理遅延を減らし共有メモリアーキテクチャの性能を向上させることができる．そこで，本報告では，まずリングトポロジを採る共有メモリアーキテクチャの性能を評価する．また評価する共有メモリアーキテクチャでは，メモリアクセスモデルとして UMA モデルと NUMA モデルを，キャッシュプロトコルとしてスヌープ法を用いる．

3　複数波長を用いた光リングネットワークの構成

本章では，共有メモリアーキテクチャにおいて用いるネットワークの構成とそのネットワークにおける通信の遅延時間について述べる．

本報告では，対象とするリングネットワークを単一方向リングとし，対象とする波長数は 2 波長と 3 波長とする．リングトポロジとして，階層化したリング構造を有する Hierachical ring [6-8] を採用する．このネットワークはノード数によらず少ない任意の波長数で実現することができ，また，ノードからネットワークへの接続に必要となる波長インターフェース数が少なく，ネットワークの構築コストを抑えることができる．

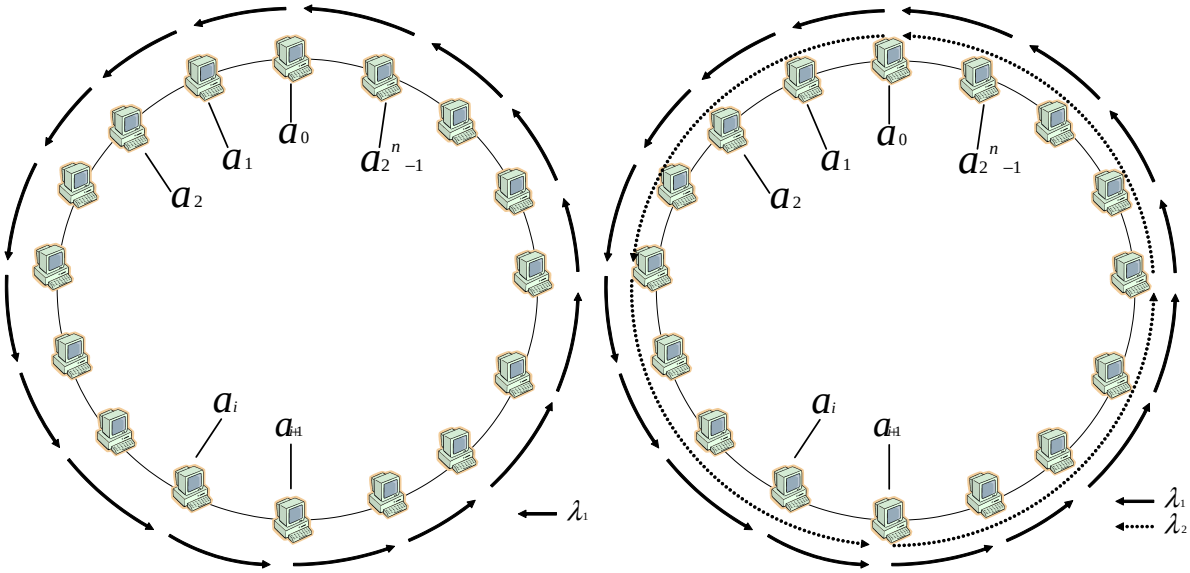
3.1　光リングネットワーク構成と記号の定義

ここでは，簡単のためにリングトポロジを形成するノード数 N を 2^n とする．これらのノードをリング状に配置し，各ノードに対して，ノード番号を 0 から順に反時計回りにつけ，$a_0, a_1, \cdots, a_{2^n-1}$ と表すものとする（図 5(a) 参照）．この状態のリングトポロジに，波長 λ_2，波長 λ_3 を，それぞれ分割数 $I = 2^i$，$I = 2^{i+1}$ で全ノードを均等に分割できるように設定する（図 5(b) 参照）．分割数を 2 のべき乗にすることによって分割されたノード数を均等にすることができる．

複数の波長を利用できるノードを複数波長ノードと呼び，複数波長ノードから半時計回りに次の複数波長ノード手前までのノードを 1 セットとして扱い，$\mathcal{S}_k$ $(k = 0, 1, \cdots I-1)$ と表す．ここで，複数波長ノードを a_{w_k} と定義する（図 6(b)）．すなわち，ノード数 $N = 2^n$，分割数 $I = 2^i$ とすると，a_{w_k} となるノードはノード番号が $a_0, a_{2^{n-i}}, a_{2\cdot 2^{n-i}}, a_{3\cdot 2^{n-i}}, \cdots, a_{(i-1)\cdot 2^{n-i}}$ のノードとなる．つまり $w_k = k \cdot 2^{n-i}, 0 \leq k \leq i-1$ と表すことができる．3 波長を利用する場合，分割数 $I = 2^{i+1}$ のため，$w_k = k \cdot 2^{n-i}, 0 \leq k \leq i$ となる．次に，ノード a_{w_k} から反時計回りに隣のノード $a_{w_{k+1}}$ の手前までを 1 セットとして区切る．このとき各セットに属するノード数は 2^{n-i} となる．3 波長の場合は 2^{n-i-1} となる．

本章では，ノード間の転送遅延時間を求めるため，送信ノードを a_s とし目的ノードを a_d $(0 \leq s \leq 2^n-1, s \neq d)$ とする．各セットはそれぞれ対称であるので送信ノードが属するセットを $\mathcal{S}_0$ に固定する．そのため，$0 \leq s \leq 2^(n-i)$ である．3 波長の場合は $0 \leq s \leq 2^(n-i+1)$ である．

遅延時間の要素には，データが光リングを伝わるのに要する伝播遅延，ノードにおける処理遅延，波長変換に要する遅延がある．したがって，a_s, a_d 間における通信の遅延時間は，各遅延時間に対する定数 T_{TD}, T_{PD}, T_{CD} と各遅延時間に対する係数を $F_{TD}(a_s, a_d), F_{PD}(a_s, a_d), F_{CD}(a_s, a_d)$ とすると，式 (1) と表すことができる．



(a) 1 波長のノード構成図　　(b) 2 波長のノード構成図

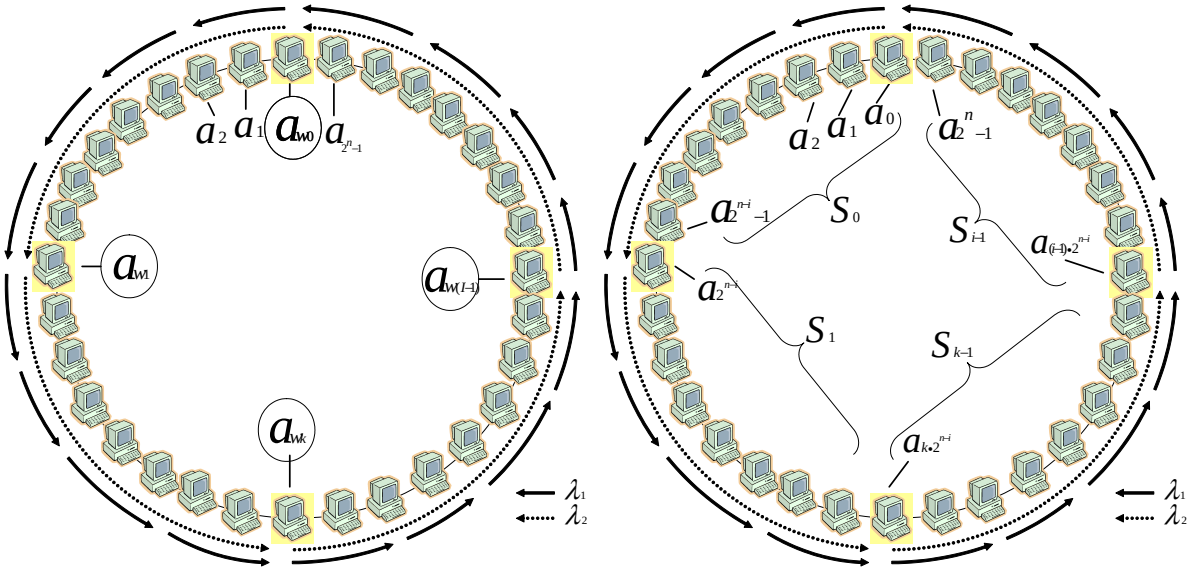
図 5: ノードの構成図

$$T_{Delay}(a_s, a_d) = F_{TD}(a_s, a_d) \cdot T_{TD} + F_{PD}(a_s, a_d) \cdot T_{PD} + F_{CD}(a_s, a_d) \cdot T_{CD} \qquad (1)$$

3.2　2 波長を用いた光リングネットワーク構成

2 波長を用いてリングネットワークを構成する．すなわち，波長 λ_2 を用いて全ノードをセットに分割する．波長 λ_2 を利用できる複数波長ノードを $a_{w_k}, 0 \leq k \leq I-1$ と定義し，ノード番号 a_0 のノードが a_{w_0} となるように波長 λ_2 を設定する（図 6(a) 参照）．

次に，波長 λ_1 における 1 ホップの遅延時間を T_{λ_1}，波長 λ_2 における 1 ホップの遅延時間を T_{λ_2} とする．波長 λ_1 は隣接するノード間で設定されているため，T_{λ_1} は $T_{TD} + T_{PD}$ となる．また，波長 λ_2 は 2^{n-i} のノード数をカットスルーするように設定されているため，T_{λ_2} は，$2^{n-i} \cdot T_{TD} + T_{PD}$ となる．したがって，波長 λ_1 によるホップ数を H_{λ_1}，波長 λ_2 によるホップ数を H_{λ_2} とすると，伝播遅延の係数 $F_{TD}(a_s, a_d)$ と処理遅延の係数 $F_{PD}(a_s, a_d)$ は $F_{TD}(a_s, a_d) = H_{\lambda_1} + H_{\lambda_2} \cdot 2^{n-i}$，$F_{PD}(a_s, a_d) = H_{\lambda_1} + H_{\lambda_2}$ と表すことができる．



(a) a_{w_k} の設定　　(b) セットへの分割

図 6: 2 波長の構成図

3.2.1 Point-to-Point による平均遅延時間

Point-to-Point で通信を行う場合，式 (1) を用いると，平均遅延時間は式 (2) で表すことができる．

$$T_{AveDelayPP} = \frac{1}{2^{n-i}} \sum_{s=0} \frac{(\sum_{d=0} T_{Delay}(a_s, a_d))}{2^n} \tag{2}$$

式 (1) における各係数の求め方は次の通りである．伝播遅延と処理遅延の係数 $F_{TD}(a_s, a_d)$, $F_{PD}(a_s, a_d)$ は使用する各波長のホップ数を計算することにより求めることができる．

2 波長を用いる $N = 16$, $I = 4$ のモデルを例に説明する．波長 λ_1 が隣接するノードを，波長 λ_2 が a_j $(j \equiv 0 \bmod 4)$ を経由するように設定されているとする（図 7）．波長 λ_1 は隣接するノード間で設定されているため，波長 λ_1 を使って a_s から a_d へ通信をした場合，8 ホップ移動することになり，遅延時間は $8 \cdot T_{TD} + 8 \cdot T_{PD}$ となる．つまり $F_{TD}(a_s, a_d) = 8$, $F_{PD}(a_s, a_d) = 8$ と表すことができる．また，波長 λ_2 を使って a_s から a_d へ通信をした場合，2 ホップ移動することになり，遅延時間は $2 \cdot 4 \cdot T_{TD} + 2 \cdot T_{PD}$ となる．つまり $F_{TD}(a_s, a_d) = 2 \cdot 4$, $F_{PD}(a_s, a_d) = 2$ と表すことができる．したがって，ホップ数を求めることにより各係数を決定することができる．また，波長変換に要する遅延の係数 $F_{CD}(a_s, a_d)$ は送信ノード a_s と目的ノード a_d がそれぞれどの波長を利用できるか，どのセットに存在するかによって一意に定めることができる．

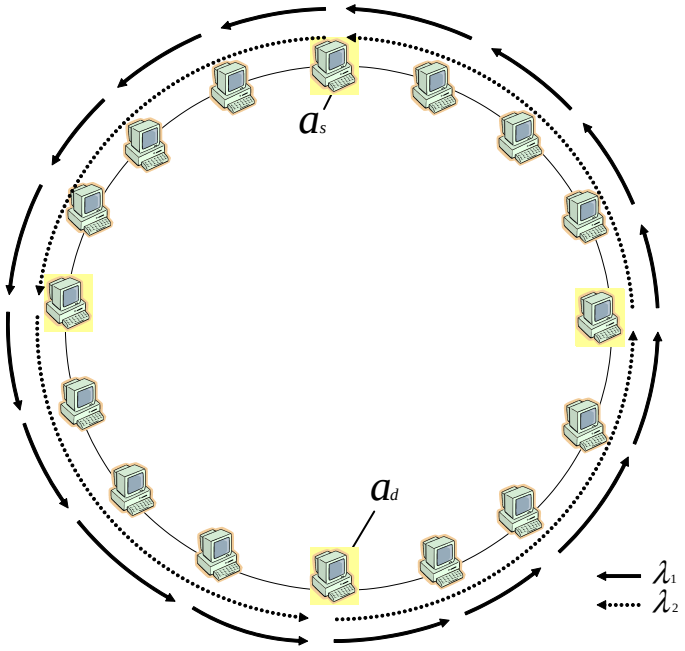


図 7: ノードの構成図

伝播遅延と処理遅延による遅延時間

平均遅延時間を求めるために，まず伝播遅延と処理遅延による遅延時間を求める．そのためには各波長のホップ数を求めればよい．a_s, a_d 間の各波長によるホップ数 H_{λ_1}, H_{λ_2} を以下の 3 つの場合に分けて計算する．また，ここでは送信ノードは $\mathcal{S}_0$ に属するものとする．

(i) 送信ノード a_s から波長 λ_2 を扱うことのできるノード a_{w_1} に到達するのに必要なホップ数

(ii) 目的ノード a_d が属するセット $\mathcal{S}_k$ の a_{w_k} に到達するのに必要なホップ数

(iii) $\mathcal{S}_k$ の a_{w_k} から目的ノード a_d に到達するのに必要なホップ数

ただし，送信ノードが a_{w_0} であった場合，(i)=0 とする．また，送信ノードと目的ノードが共に $\mathcal{S}_0$ に属し，かつ，送信ノードから目的ノードへ波長 λ_1 のみを使って $2^{n-i}-1$ ホップ以下で移動できる場合，(ii)=0 とする．(i) と (iii) は波長 λ_1 による通信，(ii) は波長 λ_2 による通信なので $H_{\lambda_1} = (\mathrm{i}) + (\mathrm{iii}), H_{\lambda_2} = (\mathrm{ii})$ となる．

a_s から a_d までの伝播遅延と処理遅延による遅延時間を $Hop(a_s, a_d)$ とし，$Hop(a_s, a_d) = H_{\lambda_1} \cdot T_{\lambda_1} + H_{\lambda_2} \cdot T_{\lambda_2}$ とする．a_s からあるセット $\mathcal{S}_k$ に属する全てのノードへの伝播遅延と処理遅延による遅延時間を足し合わせたものを $SumSet(a_s, \mathcal{S}_k)$ とする．a_s から全てのノードへの伝播遅延と処理遅延による遅延時間の合計を $SumHopByNode(a_s)$ とする．また，$\mathcal{S}_0$ に属する全てのノードの $SumHopByNode(a_s)$ の合計を $SumHop(n, i)$ とする（表 4）．

表 2: 伝播遅延と処理遅延による遅延時間の定義

$Hop(a_s, a_d)$	$H_{\lambda_1} \cdot T_{\lambda_1} + H_{\lambda_2} \cdot T_{\lambda_2}$
$SumSet(a_s, \mathcal{S}_k)$	$\sum_{d=0}^{2^{n-i}-1} Hop(a_s, a_d)$
$SumHopByNode(a_s)$	$\sum_{k=0}^{2^i-1} SumSet(a_s, \mathcal{S}_k)$
$SumHop(n, i)$	$\sum_{s=0}^{2^{n-i}-1} SumHopByNode(a_s)$

また，以下のように送信ノード・目的ノードの場合に分けて $Hop(a_s, a_d)$ を計算する．

- 送信ノードが a_{w_0} の場合
  - 目的ノードが $\mathcal{S}_0$ の場合
  - 目的ノードが $\mathcal{S}_0$ 以外の場合
- 送信ノードが a_{w_0} 以外の場合
  - 目的ノードが $\mathcal{S}_0$ の場合
  - 目的ノードが $\mathcal{S}_0$ 以外の場合

以上の場合分けに従って，送信ノード a_s から全てのセットに対する伝播遅延と処理遅延による遅延時間の和 $SumSet(a_s, \mathcal{S}_k)$ を求める．

送信ノードが a_{w_0} の場合

- 目的ノードが $\mathcal{S}_0$ の場合

$$\begin{aligned} SumSet(a_w, \mathcal{S}_0) &= (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_1}) \\ &\quad + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 1 \cdot T_{\lambda_1}) \\ &\quad + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 2 \cdot T_{\lambda_1}) + \cdots \\ &\quad + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + (2^{n-i} - 1) \cdot T_{\lambda_1}) \\ &= 0 \cdot 2^{n-i} \cdot T_{\lambda_1} + 0 \cdot 2^{n-i} \cdot T_{\lambda_2} + \sum_{j=0}^{2^{n-i}-1} j \cdot T_{\lambda_1} \end{aligned}$$

- 目的ノードが $\mathcal{S}_0$ 以外の場合

目的ノードが $\mathcal{S}_k$ に属するとすると次式で表せる．

$$\begin{aligned} SumSet(a_w, \mathcal{S}_k) &= (0 \cdot T_{\lambda_1} + k \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_1}) \\ &\quad +(0 \cdot T_{\lambda_1} + k \cdot T_{\lambda_2} + 1 \cdot T_{\lambda_1}) \\ &\quad +(0 \cdot T_{\lambda_1} + k \cdot T_{\lambda_2} + 2 \cdot T_{\lambda_1}) + \cdots \\ &\quad +(0 \cdot T_{\lambda_1} + k \cdot T_{\lambda_2} + (2^{n-i} - 1) \cdot T_{\lambda_1}) \\ &= 0 \cdot 2^{n-i} \cdot T_{\lambda_1} + k \cdot 2^{n-i} \cdot T_{\lambda_2} + \sum_{j=0}^{2^{n-i}-1} j \cdot T_{\lambda_1} \end{aligned} \tag{3}$$

送信ノードが a_{w_0} 以外の場合

- 目的ノードが $\mathcal{S}_0$ の場合

  送信ノードが $a_s = a_1$ のときは次式となる．

$$\begin{aligned} SumSet(a_s, \mathcal{S}_0) &= ((2^{n-i} - 1) \cdot T_{\lambda_1} + (2^i - 1) \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_1}) + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_1}) \\ &\quad +(0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 1 \cdot T_{\lambda_1}) + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 2 \cdot T_{\lambda_1}) \\ &\quad +\cdots + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + (2^{n-i} - 2) \cdot T_{\lambda_1}) \\ &= (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_1}) + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 1 \cdot T_{\lambda_1}) \\ &\quad +(0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 2 \cdot T_{\lambda_1}) \\ &\quad +\cdots + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + (2^{n-i} - 1) \cdot T_{\lambda_1}) + (2^i - 1) \cdot T_{\lambda_2} \\ &= SumSet(a_w, \mathcal{S}_0) + (2^i - 1) \cdot T_{\lambda_2} \end{aligned} \tag{4}$$

  同様に送信ノードが $a_s = a_j$ $(2 \leq j \leq 2^{n-i} - 1)$ のとき式 (5) と表せる．

$$SumSet(a_s, \mathcal{S}_0) = SumSet(a_w, \mathcal{S}_0) + (2^i - 1) \cdot s \cdot T_{\lambda_2} \tag{5}$$

- 目的ノードが $\mathcal{S}_0$ 以外の場合

  送信ノードが $a_s = a_1$ のときは次式で表せる．

$$\begin{aligned} SumSet(a_s, \mathcal{S}_k) &= ((2^{n-i} - 1) \cdot T_{\lambda_1} + (k - 1) \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_1}) \\ &\quad +((2^{n-i} - 1) \cdot T_{\lambda_1} + (k - 1) \cdot T_{\lambda_2} + 1 \cdot T_{\lambda_1}) \\ &\quad +((2^{n-i} - 1) \cdot T_{\lambda_1} + (k - 1) \cdot T_{\lambda_2} + 2 \cdot T_{\lambda_1}) + \cdots \\ &\quad +((2^{n-i} - 1) \cdot T_{\lambda_1} + (k - 1) \cdot T_{\lambda_2} + (2^{n-i} - 1) \cdot T_{\lambda_1}) \\ &= SumSet(a_w, \mathcal{S}_0) \\ &\quad +(2^{n-i} - 1) \cdot 2^{n-i} \cdot T_{\lambda_1} + (k - 1) \cdot 2^{n-i} \cdot T_{\lambda_2} \end{aligned} \tag{6}$$

同様に，送信ノードが $a_s = a_j$ $(2 \leq j \leq 2^{n-i} - 1)$ のときは式 (7) となる．

$$\begin{aligned} SumSet(a_s, \mathcal{S}_k) &= SumSet(a_w, \mathcal{S}_0) \\ &\quad + (2^{n-i} - s) \cdot 2^{n-i} \cdot T_{\lambda_1} + (k-1) \cdot 2^{n-i} \cdot T_{\lambda_2} \end{aligned} \tag{7}$$

波長変換に要する遅延時間

波長変換に要する遅延時間の係数は，波長変換の回数である．波長変換の回数 F_{CD} は送信ノード a_s と目的ノード a_d によって一意に決めることができる．よって，F_{CD} は以下のようになる．

- 送信ノードが a_{w_0} の場合
  - 目的ノードが $\mathcal{S}_0$ の場合

$$F_{CD} = 0$$

  - 目的ノードが $\mathcal{S}_0$ 以外の場合

$$F_{CD} = 2^{n-i} - 1$$

- 送信ノードが a_{w_0} 以外の場合
  - 目的ノードが $\mathcal{S}_0$ の場合

$$F_{CD} = 2s - 1$$

  - 目的ノードが $\mathcal{S}_0$ 以外の場合

$$F_{CD} = 2^{n-i+1} - 1$$

遅延時間

これらを用いて，H_{λ_1}，H_{λ_2} を計算すると $T_{Delay}(a_w, \mathcal{S}_0)$, $T_{Delay}(a_w, \mathcal{S}_k)$, $T_{Delay}(a_s, \mathcal{S}_0)$, $T_{Delay}(a_s, \mathcal{S}_k)$ は式 (8), (9), (10), (11) のようになる．

- 送信ノードが $a_w = a_j$ $(j \equiv 0 \bmod 2^{n-i})$ の場合
  - 目的ノードが $\mathcal{S}_0$ の場合

$$\begin{aligned} T_{Delay}(a_w, \mathcal{S}_0) &= ((2^{n-i} - 1)2^{n-i-1} + 0 \cdot 2^{n-i}) \cdot T_{TD} \\ &\quad + ((2^{n-i} - 1)2^{n-i-1} + 0) \cdot T_{PD} + 0 \cdot T_{CD} \end{aligned} \tag{8}$$

– 目的ノードが $\mathcal{S}_0$ 以外の場合

$$\begin{aligned} T_{Delay}(a_w, \mathcal{S}_k) &= ((2^{n-i}-1)2^{n-i-1}k \cdot 2^{n-i}) \cdot T_{TD} \\ &\quad +((2^{n-i}-1)2^{n-i-1}+k) \cdot T_{PD} \\ &\quad +(2^{n-i}-1) \cdot T_{CD} \end{aligned} \tag{9}$$

- 送信ノードが $a_w = a_j$ $(j \equiv 0 \bmod 2^{n-i})$ 以外の場合
  - 目的ノードが $\mathcal{S}_0$ の場合

$$\begin{aligned} T_{Delay}(a_s, \mathcal{S}_0) &= ((2^{n-i}-1)2^{n-i-1} + (2^{n-i}-1)s \cdot 2^{n-i}) \cdot T_{TD} \\ &\quad +((2^{n-i}-1)2^{n-i-1} + (2^{n-i}-1)s) \cdot T_{PD} \\ &\quad +(2s-1) \cdot T_{CD} \end{aligned} \tag{10}$$

  - 目的ノードが $\mathcal{S}_0$ 以外の場合

$$\begin{aligned} T_{Delay}(a_s, \mathcal{S}_k) &= ((2^{n-i}-1)2^{n-i-1} + (2^{n-i}-s)2^{n-i} + 2^{n-i}(k-1) \cdot 2^{n-i}) \cdot T_{TD} \\ &\quad +((2^{n-i}-1)2^{n-i-1} + (2^{n-i}-s)2^{n-i} + 2^{n-i}(k-1)) \cdot T_{PD} \\ &\quad +(2^{n-i+1}-1) \cdot T_{CD} \end{aligned} \tag{11}$$

よって,

$$T_{Delay} = \frac{1}{2^n}\Big(T_{Delay}(a_w, \mathcal{S}_0) + \sum_{k=1}^{2^i-1} T_{Delay}(a_w, \mathcal{S}_k) + \sum_{s=1}^{2^{n-i}-1} (T_{Delay}(a_s, \mathcal{S}_0) + \sum_{k=1}^{2^i-1} T_{Delay}(a_s, \mathcal{S}_k))\Big)$$

したがって，平均遅延時間 $T_{AveDelayPP}$ は

$$T_{AveDelayPP} = \frac{T_{Delay}}{2^{n-i}}$$

となる.

3.2.2 ブロードキャストによる平均遅延時間

ブロードキャストを行う場合，送信ノードは各 $\mathcal{S}_k$ 宛てにメッセージを複製し，送信する必要がある．また，全てのノードに送信したメッセージが届いたことを確認する必要がある．そのため，送信した全てのメッセージが再び自ノードに帰ってくるのを待たなければならない．よって，ブロードキャストに要する時間は最後の送信メッセージが戻ってくるまでの時間となる．送信ノード a_s が目的セット $\mathcal{S}_k$ にブロードキャストした場合の遅延時間を

$$T_{BC}(a_s, \mathcal{S}_k) = F_{TD}(a_s, a_d) \cdot T_{TD} + F_{PD}(a_s, a_d) \cdot T_{PD} + F_{CD}(a_s, a_d) \cdot T_{CD}$$

目的セット番号	送信セット番号	ノード番号

図 8: ブロードキャストアドレス

とする．

ブロードキャストを行う際にブロードキャストアドレスが必要となる．これは図 8 のように決めるものとする．先に述べたように，ブロードキャストのメッセージは各セット $\mathcal{S}_k$ に対して送信される．よって，送信先のセット番号を目的セット番号，送信する側のセット番号を送信セット番号，送信する側のノード番号をノード番号とする．例えば，$\mathcal{S}_0$ に属する a_1 が $\mathcal{S}_2$ に送信したい場合のブロードキャストアドレスは，10 00 01 となる．複数の波長を扱うノードのアルゴリズムは扱う波長数によって異なるため，各波長数のところで述べる．

遅延時間

ノード数 $N = 2^n$，分割数 $I = 2^i$ の場合においてブロードキャストに要する遅延時間を求める．ブロードキャストに要する遅延時間は，送信ノードが a_{w_0} の場合

$$T(\mathcal{S}_0) \quad = \quad (2^{n-i}) \cdot T_{\lambda_1} + (2^i - 1) \cdot T_{\lambda_2} + 2 \cdot T_{CD}$$

送信ノードが a_{w_0} 以外の場合

$$T(\mathcal{S}_0) \quad = \quad (2 \cdot 2^{n-i}) \cdot T_{\lambda_1} + (2^i - 2) \cdot T_{\lambda_2} + 4 \cdot T_{CD}$$

となる．

2 波長の場合のブロードキャストの制御信号

2 波長を扱うノード a_{w_k} が次のノードに送る波長を選択するアルゴリズムを以下のように定める．また，ノード a_{w_k} が属しているセット k のことを自セットと呼ぶこととする．

- 受信したデータのアドレス目的セットを自セットと一致するか確認し，一致した場合は波長 λ_1 に送信する
- 送信セットを自セットと一致するか確認し，一致した場合
  - ノード番号が一致するか確認し，一致した場合はメッセージを回収する
  - ノード番号が一致するか確認し，一致しなかった場合は波長 λ_1 に送信する

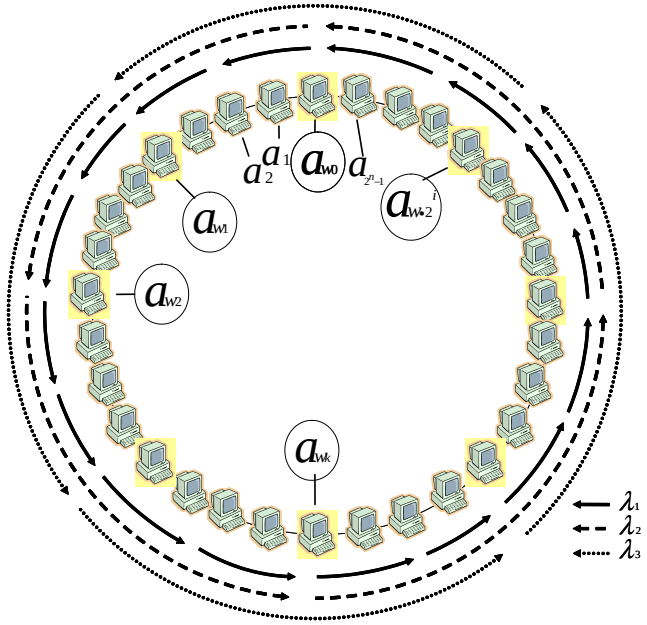


図 9: 3 波長の構成図

- それ以外の場合は波長 λ_2 に送信する．

a_{w_k} 以外の各ノードはノード番号のみを自ノード番号と比較し，一致すればメッセージを回収し，一致しなければ波長 λ_1 に送信する．

また，このときに必要となるブロードキャストアドレスの桁数は目的セットを表すのに $\log 2^i$，送信セットを表すのに $\log 2^i$，ノード番号を表すのに $\log 2^{n-i}$ 桁必要となる．つまり $i + i + (n - i) = n + i$ 桁必要となる．

3.3　3 波長を用いた光リングネットワーク構成

まず，波長 λ_2 を 2 波長を用いた構成手法と同様に設定する．次に波長 λ_3 を設定する．波長 λ_3 のホップ数は波長 λ_2 と同じとし，波長 λ_3 を扱うことのできるノードは波長 λ_2 を扱うことのできる複数波長ノードの中間のノード a_j $(j \equiv 2^{n-i-1} \bmod 2^{n-i})$ とする（図 9）・これにより各セットのノード数は 2^{n-i-1} となる．2 波長の時と同様に，複数波長ノードを順に a_{w_k} とする．

波長 λ_1 における 1 ホップの遅延時間を T_{λ_1}，波長 λ_2 における 1 ホップの遅延時間を T_{λ_2}，波長 λ_3 における 1 ホップの遅延時間を T_{λ_3} とする．波長 λ_1 は隣接するノード間で設定されているため，$T_{\lambda_1} = T_{TD} + T_{PD}$ となる．また，波長 λ_2，λ_3 は 2^{n-i} ノードをカットス

表 3: 3 波長における伝播遅延と処理遅延の係数の値

$F_{TD}(a_s, a_d)$	$H_{\lambda_1} + (H_{\lambda_2} + H_{\lambda_3}) \cdot 2^{n-i}$
$F_{PD}(a_s, a_d)$	$H_{\lambda_1} + H_{\lambda_2} + H_{\lambda_3}$

ルーするように設定されているため，T_{λ_2}，T_{λ_3} はそれぞれ $2^{n-i} \cdot T_{TD} + T_{PD}$ となる．したがって，波長 λ_1 によるホップ数を H_{λ_1}，波長 λ_2 によるホップ数を H_{λ_2}，波長 λ_3 によるホップ数を H_{λ_3} とすると，伝播遅延の係数 $F_{TD}(a_s, a_d)$ と処理遅延の係数 $F_{PD}(a_s, a_d)$ は $F_{TD}(a_s, a_d) = H_{\lambda_1} + (H_{\lambda_2} + H_{\lambda_3}) \cdot 2^{n-i}$，$F_{PD}(a_s, a_d) = H_{\lambda_1} + H_{\lambda_2} + H_{\lambda_3}$ と表すことができる（表 3）．

3.3.1 Point-to-point による平均遅延時間

伝播遅延と処理遅延による遅延時間

平均遅延時間を求めるために，まず伝播遅延と処理遅延による遅延時間を求める．そのためには各波長のホップ数を求めればよい．各セット $\mathcal{S}_k$ は対称なので送信ノードは $\mathcal{S}_0$ に属するとする．a_s, a_d 間の各波長によるホップ数 H_{λ_1}, H_{λ_2}, H_{λ_3} を以下の 3 つの場合に分けて計算する．

(i) 送信ノード a_s から複数波長ノード a_{w_1} に到達するのに必要なホップ数

(ii) 目的ノード a_d が属するセット $\mathcal{S}_k$，あるいは，その前のセット $\mathcal{S}_{k-1}$ の $a_{w_{k-1}}$ に到達するのに必要なホップ数

(iii) $\mathcal{S}_k$ の a_{w_k} から目的ノード a_d に到達するのに必要なホップ数

ただし，送信ノードが a_{w_0} であった場合，(i)=0 とする．また，目的ノードが $\mathcal{S}_0$ に属し，かつ，送信ノードから目的ノードへ波長 λ_1 のみを使って $2^{n-i-1} - 1$ ホップ以下で移動できる場合，(ii)=0 とする．(i) と (iii) は波長 λ_1 による通信，(ii) は波長 λ_2，もしくは，波長 λ_3 による通信となる．

a_s から a_d までの伝播遅延と処理遅延による遅延時間を $Hop(a_s, a_d)$ とし，$Hop(a_s, a_d) = H_{\lambda_1} \cdot T_{\lambda_1} + H_{\lambda_2} \cdot T_{\lambda_2} + H_{\lambda_3} \cdot T_{\lambda_3}$ とする．a_s からあるセット $\mathcal{S}_k$ に属する全てのノードへの伝播遅延と処理遅延による遅延時間を足し合わせたものを $SumSet(a_s, \mathcal{S}_k)$ とする．a_s から全てのノードへの伝播遅延と処理遅延による遅延時間の合計を $SumHopByNode(a_s)$ とする．また，$\mathcal{S}_0$ に属する全てのノードの $SumHopByNode(a_s)$ の合計を $SumHop(n, i)$ とする（表 4）．

表 4: 伝播遅延と処理遅延による遅延時間の定義

$Hop(a_s, a_d)$	$H_{\lambda_1} \cdot T_{\lambda_1} + H_{\lambda_2} \cdot T_{\lambda_2} + H_{\lambda_3} \cdot T_{\lambda_3}$
$SumSet(a_s, \mathcal{S}_k)$	$\sum_{d=0}^{2^{n-i-1}-1} Hop(a_s, a_d)$
$SumHopByNode(a_s)$	$\sum_{k=0}^{2^{i+1}-1} SumSet(a_s, \mathcal{S}_k)$
$SumHop(n, i)$	$\sum_{s=0}^{2^{n-i-1}-1} SumHopByNode(a_s)$

また，以下のように送信ノード・目的ノードの場合に分けて $Hop(a_s, a_d)$ を計算する．

- 送信ノードが a_{w_0} の場合
  - 目的ノードが $\mathcal{S}_0$ に属する場合
  - 目的ノードが $\mathcal{S}_k$ $(k \neq 0)$ 以外に属する場合
    - $\mathcal{S}_k$ $(k \equiv 0 \bmod 2)$ の場合
    - $\mathcal{S}_k$ $(k \equiv 1 \bmod 2)$ の場合
- 送信ノードが a_{w_0} 以外の場合
  - 目的ノードが $\mathcal{S}_0$ に属する場合
  - 目的ノードが $\mathcal{S}_k$ $(k \neq 0)$ 以外に属する場合
    - $\mathcal{S}_k$ $(k \equiv 0 \bmod 2)$ の場合
    - $\mathcal{S}_k$ $(k \equiv 1 \bmod 2)$ の場合

以上の場合分けに従って，送信ノード a_s から全てのセットに対する伝播遅延と処理遅延による遅延時間の和 $SumSet(a_s, \mathcal{S}_k)$ を求める．

送信ノードが a_{w_0} の場合

- 目的ノードが $\mathcal{S}_0$ に属する場合

$$
\begin{aligned}
SumSet(a_w, \mathcal{S}_0) &= (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 0 \cdot T_{\lambda_1}) + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 1 \cdot T_{\lambda_1}) \\
&\quad + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 2 \cdot T_{\lambda_1}) + \cdots \\
&\quad + (0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + (2^{n-i-1} - 1) \cdot T_{\lambda_1}) \\
&= 0 \cdot 2^{n-i-1} \cdot T_{\lambda_1} + 0 \cdot 2^{n-i-1} \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + \sum_{j=0}^{2^{n-i-1}-1} j \cdot T_{\lambda_1}
\end{aligned}
$$

- 目的ノードが $\mathcal{S}_k (k \neq 0)$ 以外に属する場合
  - $\mathcal{S}_k$ $(k \equiv 0 \bmod 2)$ の場合
    目的ノードが $\mathcal{S}_k$ に属するとすると式 (12) となる.

$$\begin{aligned} SumSet(a_w, \mathcal{S}_k) &= (0 \cdot T_{\lambda_1} + \frac{k}{2} \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 0 \cdot T_{\lambda_1}) \\ &\quad +(0 \cdot T_{\lambda_1} + \frac{k}{2} \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 1 \cdot T_{\lambda_1}) \\ &\quad +(0 \cdot T_{\lambda_1} + \frac{k}{2} \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 2 \cdot T_{\lambda_1}) + \cdots \\ &\quad +(0 \cdot T_{\lambda_1} + \frac{k}{2} \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + (2^{n-i-1} - 1) \cdot T_{\lambda_1}) \\ &= \frac{k}{2} \cdot 2^{n-i-1} \cdot T_{\lambda_2} + \sum_{j=0}^{2^{n-i-1}-1} j \cdot T_{\lambda_1} \end{aligned} \tag{12}$$

  - $\mathcal{S}_k$ $(k \equiv 1 \bmod 2)$ の場合
    目的ノードが $\mathcal{S}_k$ に属するとすると式 (13) となる.

$$\begin{aligned} SumSet(a_w, \mathcal{S}_k) &= (2^{n-i-1} \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + \frac{k-1}{2} \cdot T_{\lambda_3} + 0 \cdot T_{\lambda_1}) \\ &\quad +(2^{n-i-1} \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + \frac{k-1}{2} \cdot T_{\lambda_3} + 1 \cdot T_{\lambda_1}) \\ &\quad +(2^{n-i-1} \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + \frac{k-1}{2} \cdot T_{\lambda_3} + 2 \cdot T_{\lambda_1}) + \cdots \\ &\quad +(2^{n-i-1} \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + \frac{k-1}{2} \cdot T_{\lambda_3} + (2^{n-i-1} - 1) \cdot T_{\lambda_1}) \\ &= 2^{2n-2i-2} \cdot T_{\lambda_1} + + \frac{k-1}{2} \cdot 2^{n-i-1} \cdot T_{\lambda_3} + \sum_{j=0}^{2^{n-i}-1} j \cdot T_{\lambda_1} \end{aligned} \tag{13}$$

送信ノードが a_{w_0} 以外の場合

- 目的ノードが $\mathcal{S}_0$ に属する場合
  送信ノードが $a_s = a_j$ $(j \equiv 1 \bmod 2^{n-i-1})$ とすると,

$$\begin{aligned} SumSet(a_s, \mathcal{S}_0) &= ((2^{n-i-1} + 2^{n-i-1} - 1) \cdot T_{\lambda_1} + (2^i - 1) \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 0 \cdot T_{\lambda_1}) \\ &\quad +(0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 0 \cdot T_{\lambda_1}) \\ &\quad +(0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 1 \cdot T_{\lambda_1}) + \cdots \\ &\quad +(0 \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + (2^{n-i-1} - 2) \cdot T_{\lambda_1}) \\ &= 0 \cdot 2^{n-i-1} \cdot T_{\lambda_1} + 0 \cdot 2^{n-i-1} \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} \\ &\quad + \sum_{j=0}^{2^{n-i-1}-1} j \cdot T_{\lambda_1} + 1 \cdot (2^{n-i-1} \cdot T_{\lambda_1} + (2^i - 1) \cdot T_{\lambda_2}) \end{aligned}$$

となる．よって，送信ノードが $a_s = a_j$ $(j \equiv s \bmod 2^{n-i-1}, s \neq 0)$ とすると，

$$\begin{aligned} SumSet(a_s, \mathcal{S}_0) &= 0 \cdot 2^{n-i-1} \cdot T_{\lambda_1} + 0 \cdot 2^{n-i-1} \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} \\ &\quad + \sum_{j=0}^{2^{n-i-1}-1} j \cdot T_{\lambda_1} + s \cdot (2^{n-i-1} \cdot T_{\lambda_1} + (2^i - 1) \cdot T_{\lambda_2}) \end{aligned}$$

- 目的ノードが $\mathcal{S}_k$ $(k \neq 0)$ 以外に属する場合
  - $\mathcal{S}_k (k \equiv 0 \bmod 2)$ の場合
    目的ノードが $\mathcal{S}_k$ に属するとすると式 (14) となる．

$$\begin{aligned} SumSet(a_s, \mathcal{S}_k) &= ((2^{n-i} - s) \cdot T_{\lambda_1} + (\frac{k}{2} - 1) \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 0 \cdot T_{\lambda_1}) \\ &\quad + ((2^{n-i} - s) \cdot T_{\lambda_1} + (\frac{k}{2} - 1) \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 1 \cdot T_{\lambda_1}) \\ &\quad + ((2^{n-i} - s) \cdot T_{\lambda_1} + (\frac{k}{2} - 1) \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + 2 \cdot T_{\lambda_1}) + \cdots \\ &\quad + ((2^{n-i} - s) \cdot T_{\lambda_1} + (\frac{k}{2} - 1) \cdot T_{\lambda_2} + 0 \cdot T_{\lambda_3} + (2^{n-i-1} - 1) \cdot T_{\lambda_1}) \\ &= (2^{n-i} - s) \cdot 2^{n-i-1} \cdot T_{\lambda_1} + (\frac{k}{2} - 1) \cdot 2^{n-i-1} \cdot T_{\lambda_2} + 0 \cdot 2^{n-i-1} \cdot T_{\lambda_3} \\ &\quad + \sum_{j=0}^{2^{n-i-1}-1} j \cdot T_{\lambda_1} \end{aligned} \tag{14}$$

  - $\mathcal{S}_k$ $(k \equiv 1 \bmod 2)$ の場合
    目的ノードが $\mathcal{S}_k$ に属するとすると式 (15) となる．

$$\begin{aligned} SumSet(a_s, \mathcal{S}_k) &= ((2^{n-i-1} - s) \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + \frac{k-1}{2} \cdot T_{\lambda_3} + 0 \cdot T_{\lambda_1}) \\ &\quad + ((2^{n-i-1} - s) \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + \frac{k-1}{2} \cdot T_{\lambda_3} + 1 \cdot T_{\lambda_1}) \\ &\quad + ((2^{n-i-1} - s) \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + \frac{k-1}{2} \cdot T_{\lambda_3} + 2 \cdot T_{\lambda_1}) + \cdots \\ &\quad + ((2^{n-i-1} - s) \cdot T_{\lambda_1} + 0 \cdot T_{\lambda_2} + \frac{k-1}{2} \cdot T_{\lambda_3} + (2^{n-i-1} - 1) \cdot T_{\lambda_1}) \\ &= ((2^{n-i-1} - s) \cdot 2^{n-i-1} \cdot T_{\lambda_1} + 0 \cdot 2^{n-i-1} \cdot T_{\lambda_2} + \frac{k-1}{2} \cdot 2^{n-i-1} \cdot T_{\lambda_3} \\ &\quad + \sum_{j=0}^{2^{n-i}-1} j \cdot T_{\lambda_1} \end{aligned} \tag{15}$$

波長変換に要する遅延時間

波長変換に要する遅延時間は，波長変換の回数を求めればよい．波長変換の回数 F_{CD} は送信ノード a_s と目的ノード a_d によって一意に決めることができる．よって，F_{CD} は以下のようになる．

- 送信ノードが a_{w_0} の場合

$$F_{CD} \quad = \quad 2^n - (2^{i+1} - 2) - 2^{n-i}$$

- 送信ノードが a_{w_0} 以外の場合
  送信ノードを a_s とすると

$$F_{CD} \quad = \quad 2^{n+1} - (2^{i+1} - 2) - 2^{n-i+1} + (s - 1)$$

遅延時間

これらを用いると遅延時間は

$$\begin{aligned} T_{Delay} \quad = \quad & \frac{1}{2^n}\Big(T_{Delay}(a_w, \mathcal{S}_0) + \sum_{k=1}^{2^{i+1}-1} T_{Delay}(a_w, \mathcal{S}_k) + \sum_{s=1}^{2^{n-i-1}-1} (T_{Delay}(a_s, \mathcal{S}_0) + \sum_{k=1}^{2^i-1} T_{Delay}(a_s, \mathcal{S}_k)) \\ & + ((2^n - (2^{i+1} - 2) - 2^{n-i}) + \sum_{s=1}^{2^{n-i-1}-1} (2^{n+1} - (2^{i+1} - 2) - 2^{n-i+1} + (s - 1))) \cdot T_{CD}\Big) \end{aligned}$$

したがって，平均遅延時間 $T_{AveDelayPP}$ は

$$T_{AveDelayPP} \quad = \quad \frac{T_{Delay}}{2^{n-i-1}}$$

となる．

3.3.2 ブロードキャストによる平均遅延時間

遅延時間

ノード数 $N = 2^n$，分割数 $I = 2^{i+1}$ の場合においてブロードキャストに要する遅延時間を求める．ブロードキャストに要する遅延時間は，送信ノードが a_{w_0} の場合

$$T(\mathcal{S}_0) \quad = \quad (2^{n-i}) \cdot T_{\lambda_1} + (2^i - 1) \cdot T_{\lambda_2} + 3 \cdot T_{CD} \tag{16}$$

送信ノードが a_{w_0} 以外の場合

$$T(\mathcal{S}_0) \quad = \quad (2 \cdot 2^{n-i}) \cdot T_{\lambda_1} + (2^i - 2) \cdot T_{\lambda_2} + 4 \cdot T_{CD} \tag{17}$$

となる．

したがって，伝播遅延，処理遅延による遅延時間は 2 波長のとき同じになり，波長変換に要する遅延時間が増加し，全体として遅延時間が少し長くなる．

ブロードキャストアドレスについて

複数波長ノード a_{w_k} のアルゴリズムを以下のように定める．

- 目的セットを自セット，および自セットの次のセットと一致するか確認し，一致した場合は波長 λ_1 に送信する
- 送信セットを自セット，および自セットの次のセットと一致するか確認し，一致した場合
  - 送信セットが自セット，かつノード番号が一致した場合はメッセージを回収する
  - 送信セットもしくはノード番号が異なる場合は波長 λ_1 に送信する
- それ以外の場合は波長 λ_2 もしくは波長 λ_3 に送信する．

a_{w_k} 以外の各ノードはノード番号のみを自ノード番号と比較し，一致すればメッセージを回収し，一致しなければ波長 λ_1 に送信する．

また，このときに必要となるブロードキャストアドレスの桁数は目的セットを表すのに $\log 2^{i+1}$，送信セットを表すのに $\log 2^{i+1}$，ノード番号を表すのに $\log 2^{n-i-1}$ 桁必要となる．つまり $(i+1)+(i+1)+(n-i-1) = n+i+1$ 桁必要となる．

3.4 分割数 i の決定方法

分割数 i は point-to-point，および，ブロードキャストの遅延時間である $T_{AveDelayPP}$，$T_{DelayBC}$ の値が最も小さくなる，$[\frac{n}{2}]$ となる．

4　共有メモリアーキテクチャのモデル化と評価

本章では，2.3 節で検討した複数波長を有するリングネットワークを利用してデータ共有を行う共有メモリアーキテクチャをモデル化し，解析を行う．モデル化と解析にはセミ・マルコフ過程を用いる．

4.1　セミ・マルコフ過程

本報告では，セミ・マルコフ過程 [9] によって共有メモリアーキテクチャをモデル化する．セミ・マルコフ過程では，それぞれの状態に任意の滞在時間を設定することができる．したがって，セミ・マルコフ過程を用いることにより，キャッシュ一貫性制御などの複雑な要求が起こる共有メモリアーキテクチャのモデル化が容易になる．

確率過程を $\{X(t), t \geq 0\}$ とし，有限状態を持っているとする．$\{X(t)\}$ では，状態変化の瞬間を $t_0 < t_1 < t_2 \cdots$ とし，$X_n = X(t_n)$ とする．この場合，セミ・マルコフ過程の確率過程 $\{X(t), t \geq 0\}$ はマルコフ過程の確率過程 $\{X_n | n = 0, 1, 2, \cdots\}$ と同じになる．離散時間型マルコフ連鎖の定常状態確率を求めるのと同じ方法を用いることにより，セミ・マルコフ過程により定常状態確率を得ることができる．セミ・マルコフ過程による状態の変化の瞬間は，離散時間型マルコフ連鎖と同じ振る舞いをする．定常状態確率をセミ・マルコフ過程より得る具体的なアルゴリズムは以下の通りである [9, 10].

1. 離散時間型マルコフ連鎖のための定常状態確率を π とし，状態遷移行列 $p = (p_{i,j})$ を用いてこの定常状態確率を計算する

2. セミ・マルコフ過程の全ての状態 $\{i\}$ について，平均滞在時間 η_i を計算する．

3. 以下のようにして，表 6 の滞在時間を用いることにより，セミ・マルコフ過程における定常状態確率を計算する．

$$P_i = \frac{\pi_i \eta_i}{\sum_j \pi_j \eta_j} \tag{18}$$

また，脱出確率 λ_i は以下のようになる．

$$\lambda_i = \frac{P_i}{\eta_i} \tag{19}$$

4.2　モデルで用いる変数の定義

シミュレーションに用いるパラメータを表 5 に示す.

表 5: モデルで用いるパラメータ

キャッシュヒット率	h
メインメモリアクセスにおける読み込みの割合	r
メインメモリアクセスにおける書き込みの割合	w
メインメモリアクセスにおける共有メモリへのアクセスの割合	s

あるノードが D 状態のキャッシュラインを持っている確率 P_D は以下のように表せる．

$$P_D = hws \tag{20}$$

ひとつのキャッシュラインに対して，自分以外のノードのうちひとつのノードだけが D 状態であり，それ以外のノードがそのキャッシュラインで D 状態を持たない確率 P_d は以下のようになる．

$$P_d = (N-1)hws(1-hws)^{N-1} \tag{21}$$

あるノードが C 状態のキャッシュラインを持っている確率 P_C は以下のようになる．

$$P_C = hrs \tag{22}$$

したがって，自分以外のノードのうち少なくともひとつのノードが C 状態のキャッシュラインを持っている確率 P_c は以下のようになる．

$$P_c = 1 - (1-hrs)^{N-1} \tag{23}$$

新しいキャッシュラインへのキャッシュメモリの空きが全くない確率 P_x は以下のようになる．キャッシュメモリが新しいキャッシュラインに空きを全く持っていないと，キャッシュラインの空きが自分以外のノードの無効化メッセージによって作られる．

$$P_x = (1-P_{inv})^{N-1} \tag{24}$$

ここで確率 P_{inv} は，あるキャッシュラインが自分以外のノードによって無効化される確率である．無効化メッセージは，自分以外のあるノードが自分の持つキャッシュラインに書き込みを行う場合に送信される．自分以外のあるノードが自分の持つキャッシュラインに書き込みを行う場合は 2 通りある．1 つ目は，自分以外のキャッシュラインを更新したいノードと自分も，該当するキャッシュラインを持っており，そのキャッシュラインが C 状態の場合である．2 つ目は，自分以外のキャッシュラインを更新したいノードが，該当するキャッシュ

ラインを持っておらず，かつ，自分は該当するキャッシュラインを持っており，そのキャッシュラインが D 状態の場合である．よって，確率 P_{inv} は以下のように表すことができる．

$$P_{inv} = hP_c w + (1-h)(1-P_d)w \tag{25}$$

4.3 共有メモリのモデル化

λ コンピューティング環境における共有メモリアーキテクチャをセミ・マルコフ過程を用いてモデル化する．各アーキテクチャの振る舞いにしたがって，各ノードの CPU の観点から状態遷移図を作成する．

4.3.1 リング UMA アーキテクチャ

リング UMA アーキテクチャの状態遷移図を図 10 に表す．状態 1 は計算状態である．この状態では，CPU はメモリアクセスを必要としない計算処理を行う．そして，読み込みか書き込み処理が発生した場合，CPU の状態は，CPU がローカルメモリにアクセスする場合は状態 {32} に，共有メモリにアクセスする場合は状態 {2} に遷移する．CPU がローカルメモリにアクセスする状態 {32} に遷移した場合，ローカルメモリアクセス後に CPU は状態 {1} に戻る．CPU が共有メモリにアクセスする状態 {2} に遷移した場合，状態 $\{3, 4, 8, 22, 23, 31\}$ のうちのひとつに遷移する．書き込みアクセスの場合は状態 $\{3, 4, 8\}$ のいずれかに，また，読み込みアクセスの場合は状態 $\{22, 23, 31\}$ のいずれかに遷移する．

自分以外のノードがあるキャッシュラインにアクセスしている間，CPU による同じ共有メモリに対するアクセスはブロックされる．この確率を P_B とする．それぞれの CPU がアクセスするキャッシュラインをランダムに選択するとすると，各キャッシュラインがアクセスされる確率は $\frac{l}{C}$ となる．さらに，CPU が共有キャッシュにアクセスしているのは状態 $\{10, 19, 20, 21, 25, 34\}$ であり，これらの状態を S_a とする．少なくともひとつ以上のノードが同じキャッシュラインにアクセスしたときに，アクセスがブロックされる．これは．CPU が状態 S_a にいるときに確率 s で起こる．したがって，各キャッシュラインへアクセスされる確率を α_1 とすると，これは以下のように表すことができる．

$$\alpha_1 = \frac{l}{C} \sum_{s \in S_a} P_s \tag{26}$$

したがって，確率 P_B は以下のようになる．

$$P_B = 1 - (1-\alpha_1)^{N-1} \tag{27}$$

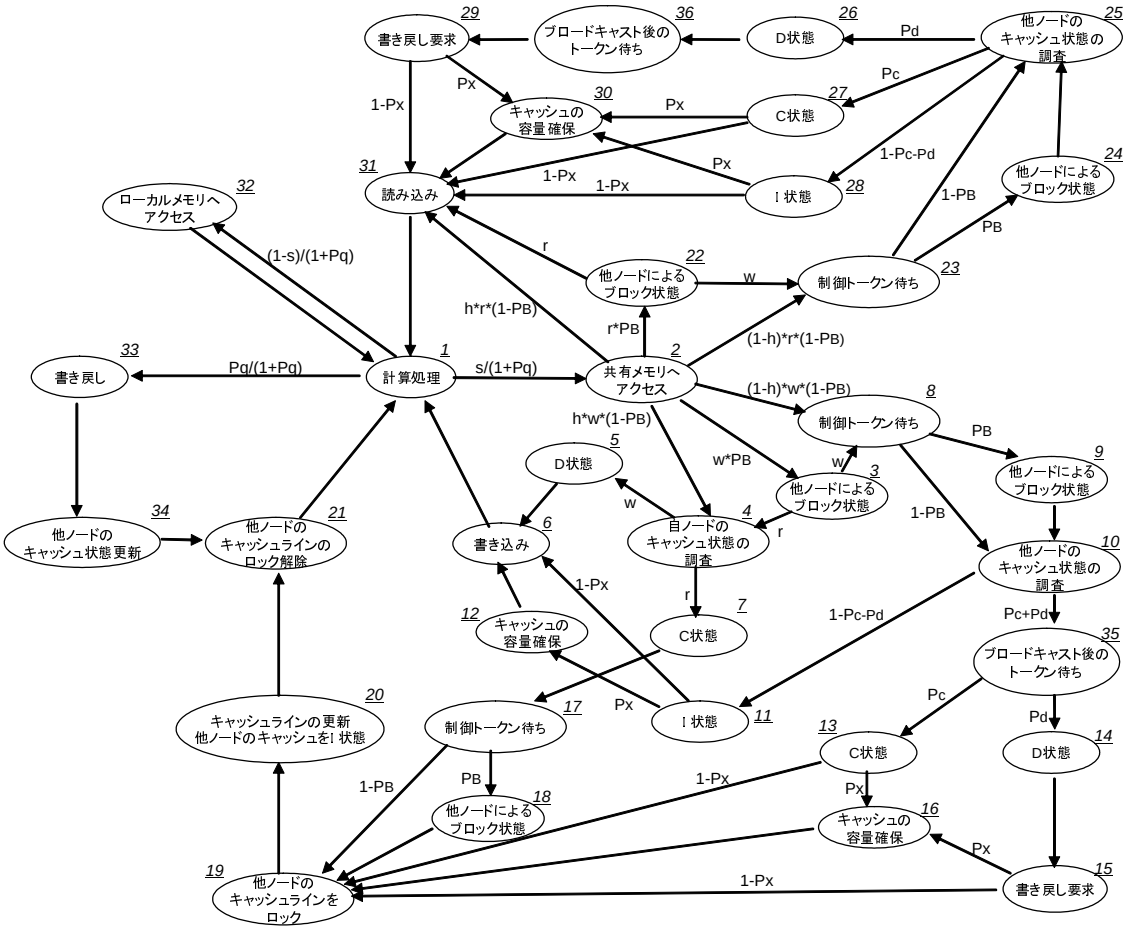


図 10: リング UMA アーキテクチャの状態遷移図

自分以外のノードから書き戻し要求メッセージが送られた場合，CPU は共有メモリに該当キャッシュを書き戻さなければならない．書き戻し要求の起こる確率を P_q とする．書き戻し要求メッセージが送られる状態は状態 $\{14, 36\}$ である．したがって書き戻し要求は少なくともひとつのノードが状態 $\{35, 36\}$ であり，状態 $\{35\}$ の場合は確率 P_d で，状態 $\{36\}$ の場合は確率 1 で起こる．これより確率 P_q は以下のようになる．

$$P_q = P_d \cdot \lambda_{35} + \lambda_{36} \tag{28}$$

また，リング UMA キテクチャにおける状態の滞在時間を表 6 に示す．

4.3.2 リング NUMA アーキテクチャ

リング NUMA アーキテクチャの状態遷移図を図 11 に表す．ほとんどの状態と遷移確率がリング UMA アーキテクチャと同じであるため，詳細な説明は省略する．リング NUMA アーキテクチャの状態遷移図で重要となるのは，状態 $\{15, 17, 32\}$ の滞在時間である．キャッシュミスが生じた場合，これらの状態に遷移する．そして，この状態ではキャッシュコピー要求メッセージがホームノードに送られる．キャッシュコピー要求メッセージを受け取ったホームノードは，自分の共有メモリに自分しかアクセスできないように制限しなければならない．この確率は $\frac{M}{M \times N} = \frac{1}{N}$ となる．ホームノードにキャッシュコピー要求メッセージを送ったノードは，ホームノードからキャッシュコピーを受け取るのを待たなければならない．この確率は $1 - \frac{1}{N}$ である．したがって，状態 $\{15, 17, 32\}$ の滞在時間は以下のようになる．

$$\eta_{15} = \eta_{17} = \eta_{32} = t_2 \frac{1}{N} + \tau(1 - \frac{1}{N}) \tag{29}$$

さらに，確率 P_B，P_q はリング UMA アーキテクチャの場合と同様に以下のようになる．α_2 は書くキャッシュラインにアクセスされる確率である．

$$\alpha_2 = \frac{l}{C} \sum_{s \in S_b} P_s \tag{30}$$

また，CPU が共有キャッシュにアクセスする状態は $S_b = \{10, 15, 17, 21, 22, 23, 27, 32, 37\}$ となる（図 11）．したがって，確率 P_B は以下のようになる．

$$P_B = 1 - (1 - \alpha_2)^{N-1} \tag{31}$$

そして，確率 P_q は以下のようになる．

$$P_q = P_d \cdot \lambda_{10} + \lambda_{38} \tag{32}$$

また，リング NUMA アーキテクチャにおける状態の滞在時間を表 6 に示す．

表 6: 各状態の滞在時間 [μs].

State	Ring-UMA	Ring-NUMA
1	0.002	0.002
2	0	0
3	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$
4	t_1	t_1
5	0	0

表 6: 各状態の滞在時間 [μs].

State	Ring-UMA	Ring-NUMA
6	t_1	0
7	0	t_1
8	0.5τ	0.5τ
9	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$
10	$T_{DelayBC}$	τ
11	0	0
12	t_2	0
13	0	0
14	0	2τ
15	2τ	$\frac{1}{N} \times t_2 + (1 - \frac{1}{N}) \times T_{AveDelayPP}$
16	t_2	t_2
17	0.5τ	$\frac{1}{N} \times t_2 + (1 - \frac{1}{N}) \times T_{AveDelayPP}$
18	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$	t_2
19	τ	0.5τ
20	τ	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$
21	$T_{DelayBC}$	τ
22	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$	τ
23	0.5τ	$T_{DelayBC}$
24	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$
25	$T_{DelayBC}$	0.5τ
26	0	$\frac{3}{2}\tau + \frac{1}{2}T_{DelayBC}$
27	0	$T_{DelayBC}$
28	0	0
29	2τ	0
30	t_2	0
31	t_1	2τ
32	$ht_1 + (1-h)(t_1 + t_2)$	$\frac{1}{N} \times t_2 + (1 - \frac{1}{N}) \times T_{AveDelayPP}$
33	$t_1 + t_2$	t_2
34	2τ	t_1
35	$\tau - T_{DelayBC}$	$ht_1 + (1-h)(t_1 + t_2)$

表 6: 各状態の滞在時間 [μs].

State	Ring-UMA	Ring-NUMA
36	$\tau - T_{DelayBC}$	$t_1 + t_2$
37		2τ
38		$\tau - T_{DelayBC}$

4.4 セミ・マルコフ過程による解析

本節では，セミ・マルコフ過程による定常状態確率の求め方を述べ，数値例により解析結果を示す.

4.4.1 解析方法

セミ・マルコフ過程を次の手法で解析する.

1. 状態確率 $P = (P_{i,j})$ を初期化する
2. 離散時間型マルコフ連鎖で方程式 $\pi = \pi P$ を解くことによって，定常分布ベクトル $\pi = \{\pi_i\}$ を得る
3. 式 (18) により定常状態確率を求める
4. $\{P_i\}$ を用いて状態確率を更新する
5. 直前の $\{P_i\}$ の値と現在の $\{P_i\}$ の値の差が，設定した十分小さな閾値より大きければ 2 に戻る

4.4.2 数値例

本報告では，λ コンピューティング環境において物理的にリングトポロジを採る場合の共有メモリアーキテクチャを対象としている．したがって，光リングネットワークのリング長を L とし，ネットワークのノード数とは独立したパラメータとして用いる．また，共有メモリアーキテクチャの解析のために，表 7 のようにそれぞれのパラメータを設定する．次に，CPU の使用割合として，読み込みが 15%，書き込みが 5%，その他の演算処理が 80%とする．したがって，メモリアクセスの読み込み割合は $r = 0.75$，書き込み割合は $w = 0.25$ となる．また，ノード数 N，ネットワークのリング長 L，共有メモリへのアクセス率 s をパラメータとする.

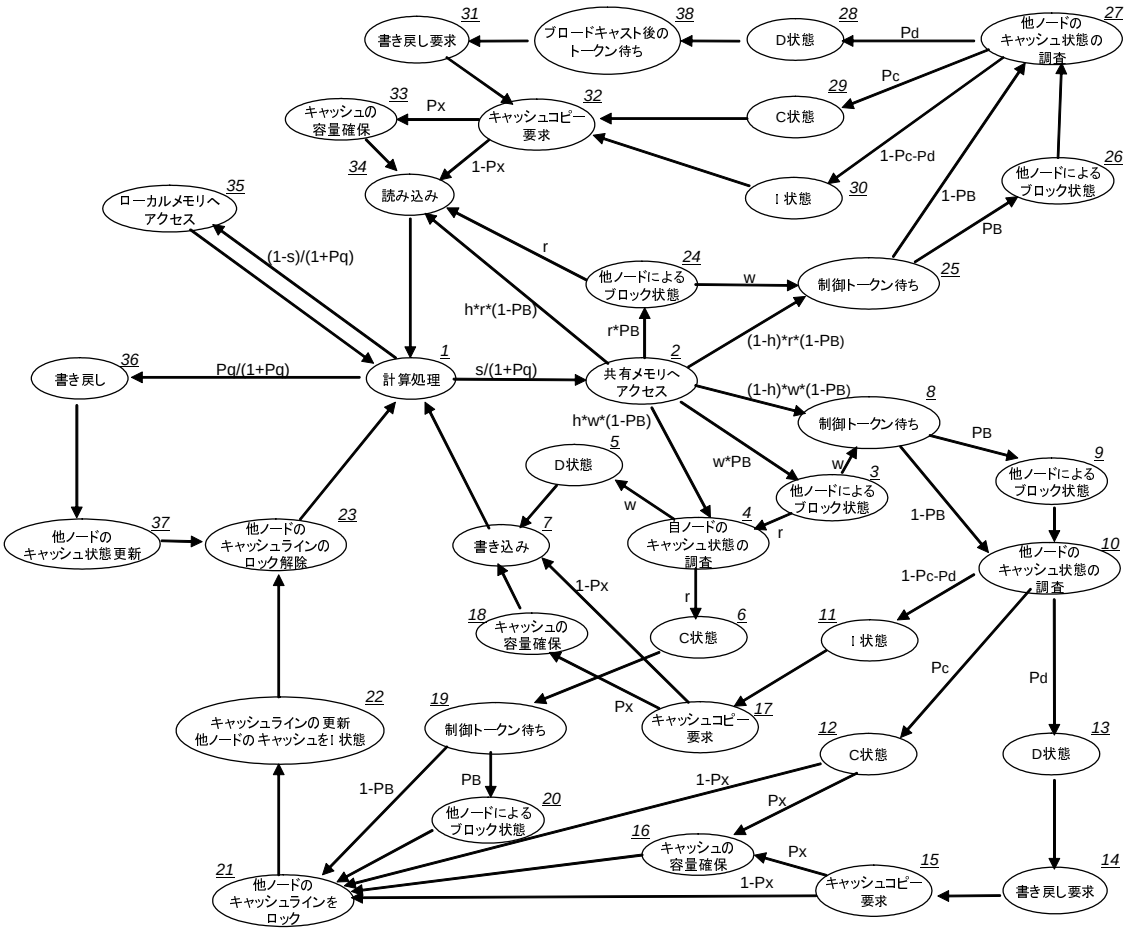


図 11: リング NUMA アーキテクチャの状態遷移図

定常状態確率

図 12，図 13，図 14 にリング UMA アーキテクチャの状態確率を，図 15，図 16，図 17 にリング NUMA アーキテクチャの状態確率を示す．リング UMA アーキテクチャでは，CPU が一貫性制御を行っている状態確率は P_{19}, P_{20}, P_{21} であり，計算処理を行っている状態確率は P_{32} である．また，リング NUMA アーキテクチャでは，CPU が一貫性制御を行っている状態確率は P_{21}, P_{22}, P_{23} であり，計算処理を行っている状態確率は P_{35} である

一貫性制御の状態確率は，共有メモリへのアクセス頻度やデータ共有に要する遅延時間に影響される．遅延時間が増加したり共有メモリアクセスの頻度が増加した場合，この状態確率は大きくなる．遅延時間としては，ノードの処理遅延やデータの伝播遅延，波長変換に要する遅延などが挙げられる．しかし波長変換に要する遅延は，ノードの処理遅延やデータの伝播遅延に比べると小さく，全体の遅延時間にそれほど影響を与えない．一方，全体の遅延

表 7: 対象モデルにおけるパラメータの値

CPU クロック	2	[GHz]
CPU–L2 キャッシュ間のアクセス時間 t_1	0.01	[μs]
L2 キャッシュ–メインメモリ間のアクセス時間 t_2	1	[μs]
ネットワークインターフェイスのフレーム処理時間 t_3	3	[μs]
各ノードの共有メモリ容量 M	1024	[MB]
L2 キャッシュ容量 C	1024	[KB]
キャッシュラインのサイズ l	4	[KB]
ネットワーク容量 B	10	[Gbps]
キャッシュヒット率 h	0.95	
波長変換に要する時間 T_{CD}	0	[μs]

時間の多くを占めるノードの処理遅延やデータの伝播遅延は，リング長が長くなったりノード数が増加したりすると大幅に増加する．そのため，リング長が短くノード数が少ない場合は，データ共有における遅延時間が小さく，共有メモリへのアクセス頻度によらず一貫性制御の状態確率が小さく，計算処理の状態確率が大きくなる（図 12，図 15）．またリング長が長くなったりノード数が増加した場合でも共有メモリへのアクセス頻度が低い場合は同様の状態確率となる（図 13(b)，図 16(b)）．一方，共有メモリへのアクセス頻度が高い場合には，一貫性制御の状態確率が大きく，計算処理の状態確率が小さくなる（図 13(a)，図 16(a)）．これらの状態では 1 波長の場合と複数波長を用いた場合で状態確率に差が生じていない．複数波長を用いる利点は，データ共有のノードの処理遅延を低く抑えることができる点である．しかし，共有メモリへのアクセス頻度が高くてもノード数が少ない場合，使用する波長数によるノードの処理遅延の差がほとんどないため，各波長数における状態確率の差はない（図 12(a)，図 13(a)，図 15(a)，図 16(a)）．また共有メモリへのアクセス頻度が低い場合は，一貫性制御を行う回数が少なくデータ共有の遅延時間に差があっても各波長数における状態確率はほとんど変わらない（図 12(b)，図 13(b)，図 15(b)，図 16(b)）．一方，アクセス頻度が高い場合，ノード数が多くなると波長数による状態確率に差が生じる（図 14，図 17）．複数波長を用いた状態確率では一貫性制御の状態確率が減少し，計算処理の状態確率が増加している．リング長が短いほど演算処理の状態確率に差が生じ，効率よく計算処理が行われているのがわかる（図 14(a)，図 17(a)）．リング長が長くなると演算処理の状態確率の差が小さくなるのは，データ共有に要する遅延時間に占めるノードの処理遅延の割合が減少しデータの伝播遅延の割合が増加するためである．

リング UMA アーキテクチャでは 2 波長と 3 波長における差が生じていない．リング UMA アーキテクチャではデータ共有方法としてブロードキャストを用いている．ブロードキャストによる遅延時間は全ノードにメッセージが伝わる時間である．波長 $\lambda 2$ と波長 $\lambda 3$ を用いて各ノードへの平均遅延時間を短くすることはできる．しかし，最大遅延時間を短くすることはほとんどできていない．そのためブロードキャストの遅延時間が変わらず，状態確率に差が生じていない．また，リング NUMA アーキテクチャでは，リング UMA アーキテクチャでは見られなかった 2 波長と 3 波長における差を確認することができるが，状態確率に大きな違いは生じていない（図 17(b)）．これはトポロジとしてリングトポロジを用いているために，リングの伝播遅延時間を短くできないことが大きな原因となっている．

4.5 共有メモリアーキテクチャの性能評価

4.4 節では，パラメータに値を設定することで状態確率分布を求め共有メモリアーキテクチャの傾向，振る舞いを知ることができた．しかし，共有メモリアーキテクチャの性能を評価することはできていない．そこで，本節では，4.4 節で行った数値解析の結果を用いて，ネットワーク利用率，平均メモリアクセス時間，計算スループットを求め，これらを指標として共有メモリアーキテクチャの性能評価を行う．

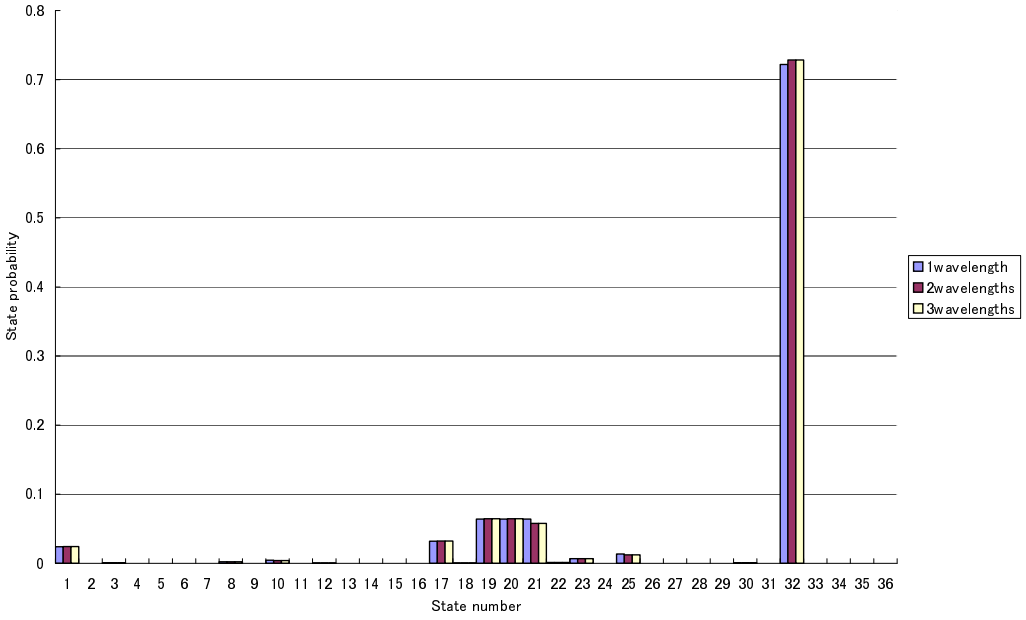
4.5.1 ネットワーク利用率

ネットワーク利用率を以下のように定義する．

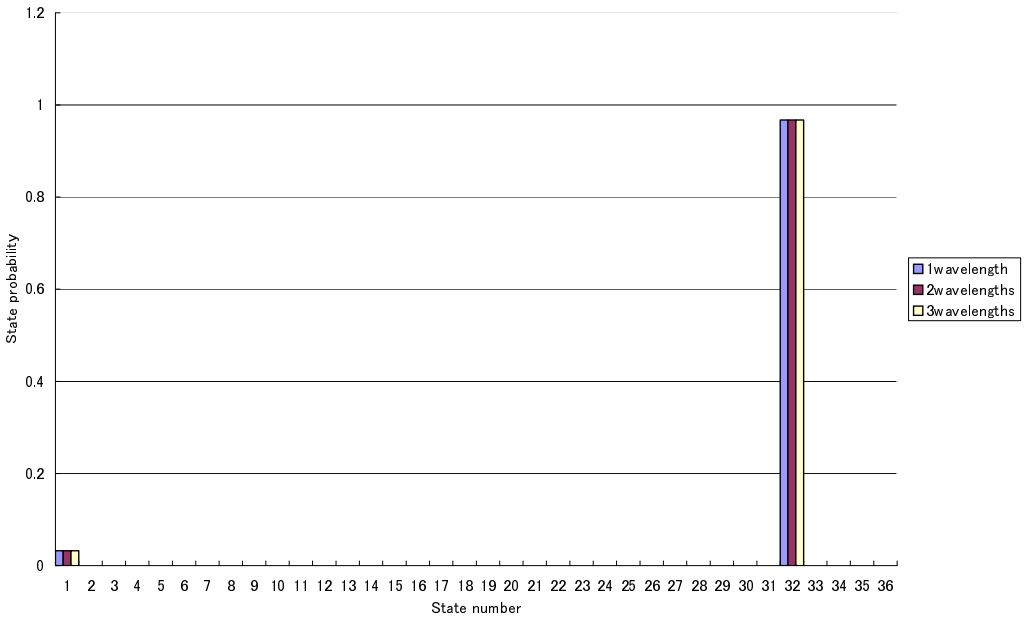
$$\text{Network utilization} = N \times \sum_{v \in V} \left(\frac{D_v}{B \times \eta_v} P_v \right) \tag{33}$$

ここで，V をデータを送信する状態の集合，D_v を状態 v で送信するデータのサイズとする．状態 v の滞在時間はネットワーク利用率に影響を及ぼす．すなわち，これらの状態の滞在時間が長い場合，ネットワーク利用率は低くなる．ここでは送るデータのサイズとして，制御メッセージを 32byte，キャッシュラインを 4Kbyte とする．

- リング UMA アーキテクチャ
  $V = \{10, 15, 19, 20, 21, 25, 29, 34\}$
  $D_{10} = D_{21} = D_{25} = 32 \times N$ bit
  $D_{15} = D_{19} = D_{20} = D_{29} = 32$ bit
  $D_{34} = 32 + 4 \times 8 \times 10^3$ bit

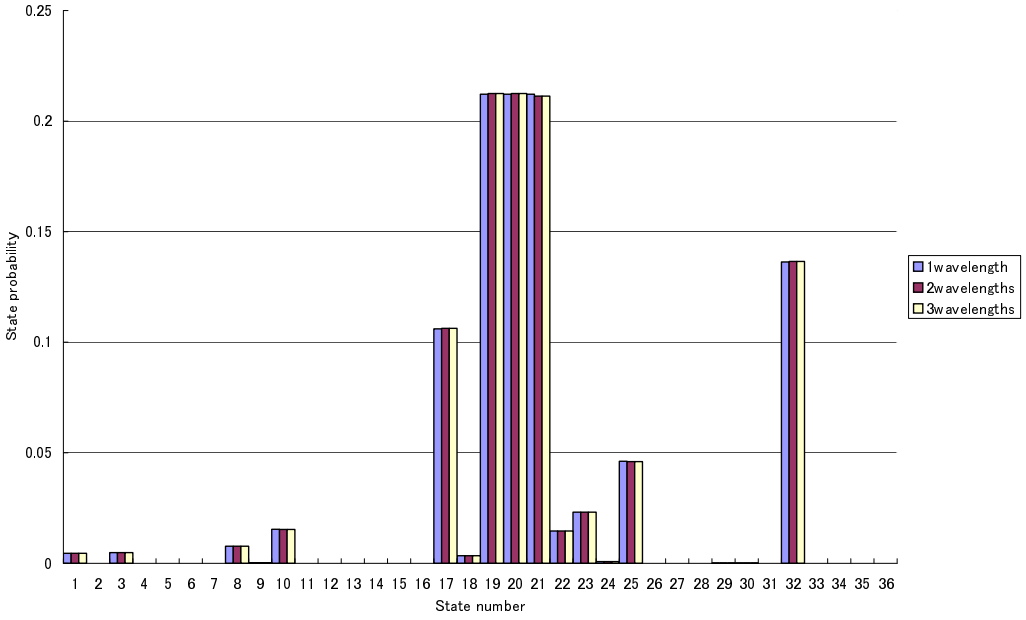


(a) $s = 10^{-3}$

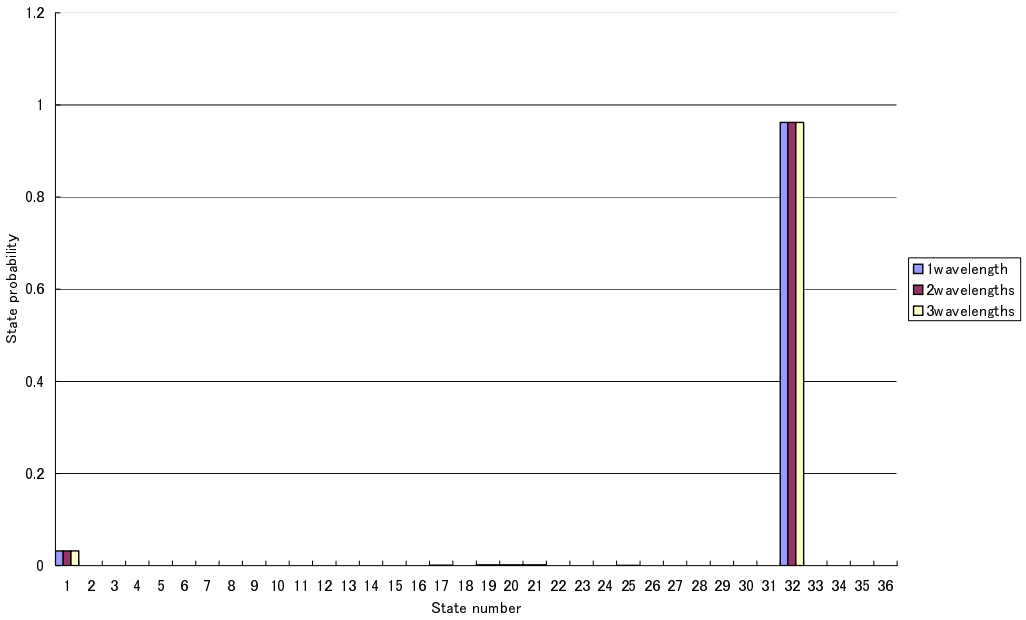


(b) $s = 10^{-6}$

図 12: リング UMA アーキテクチャの定常状態確率 ($L = 1\text{km}$, $N = 4$)

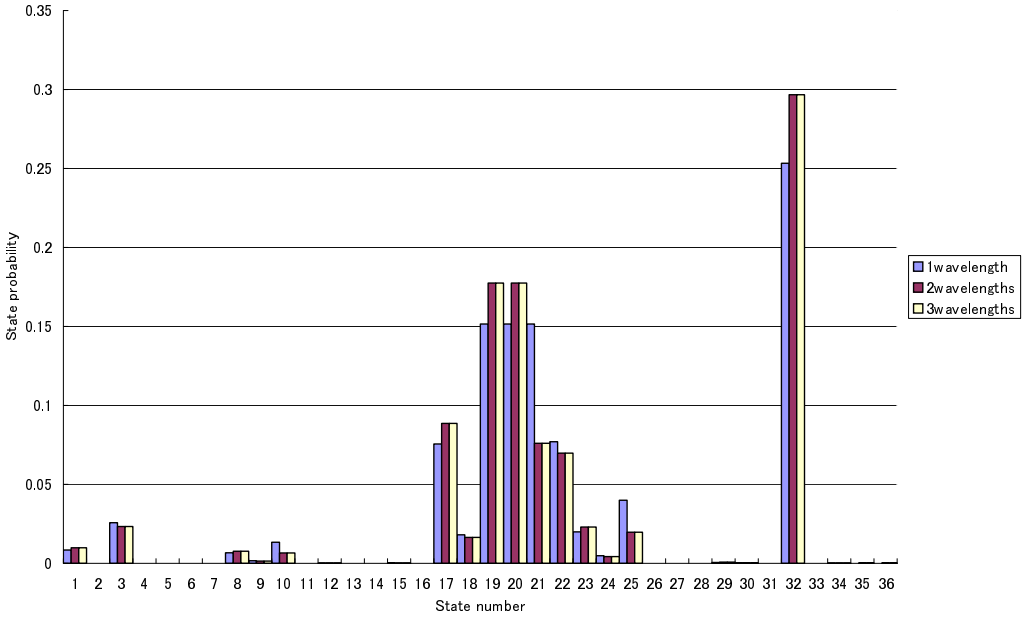


(a) $s = 10^{-3}$

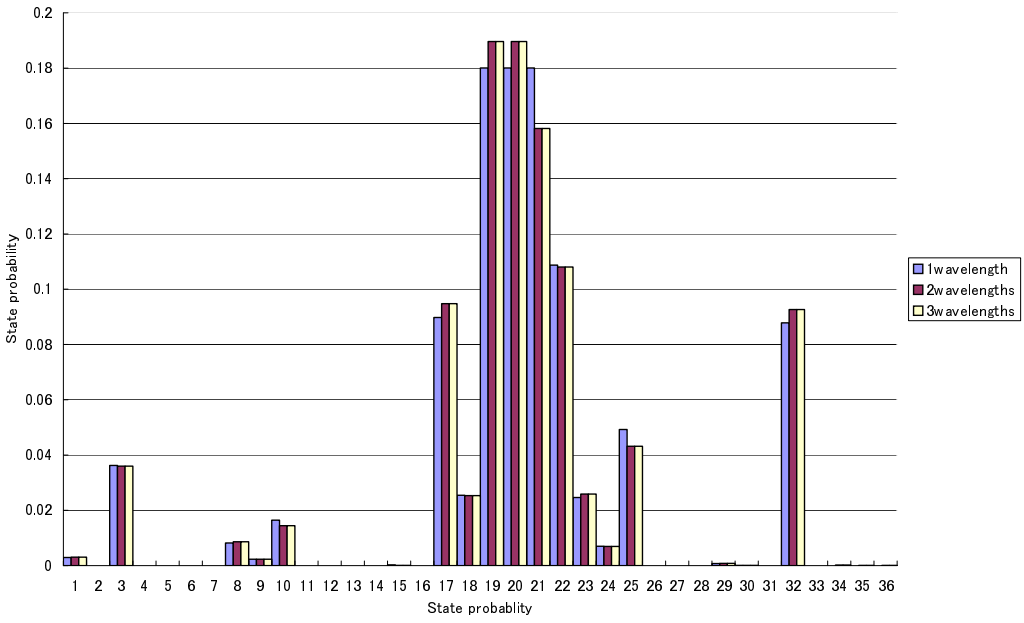


(b) $s = 10^{-6}$

図 13: リング UMA アーキテクチャの定常状態確率 ($L = 100$km, $N = 4$)

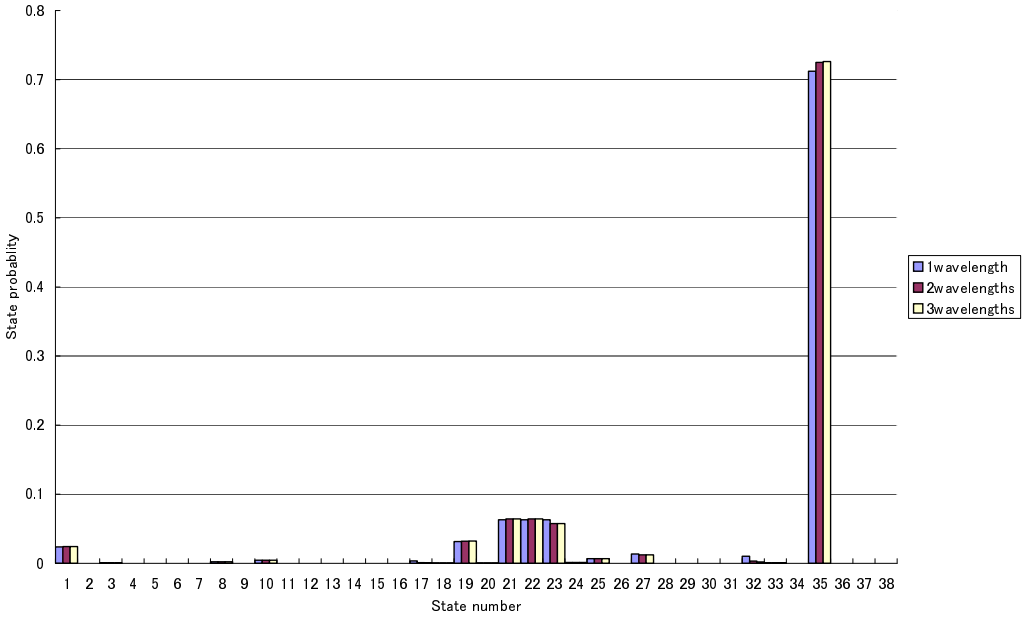


(a) $L = 1\text{km}$

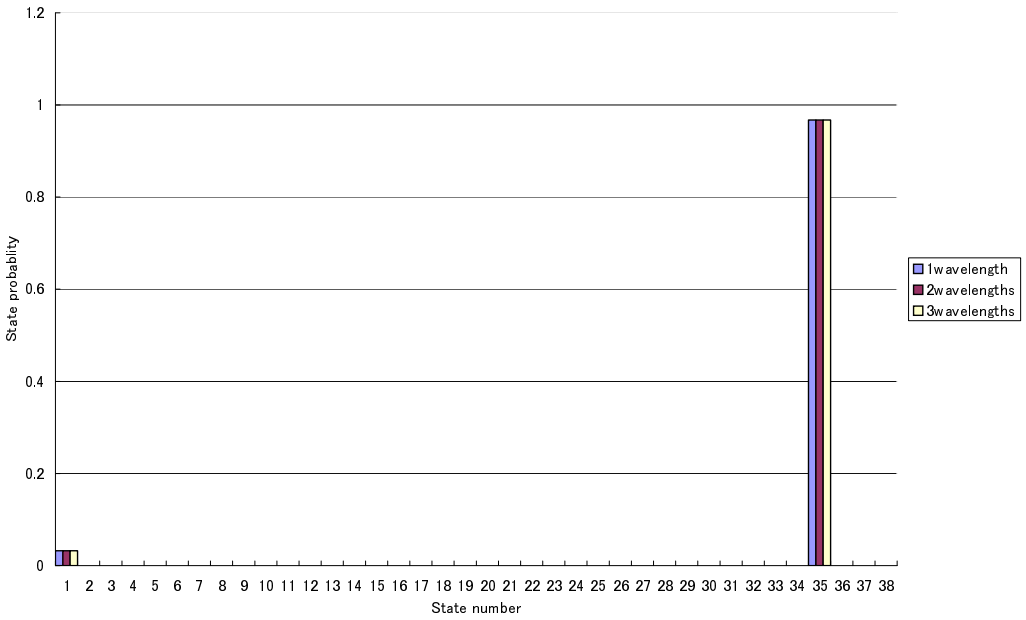


(b) $L = 100\text{km}$

図 14: リング UMA アーキテクチャの定常状態確率 ($N = 32$, $s = 10^{-3}$)

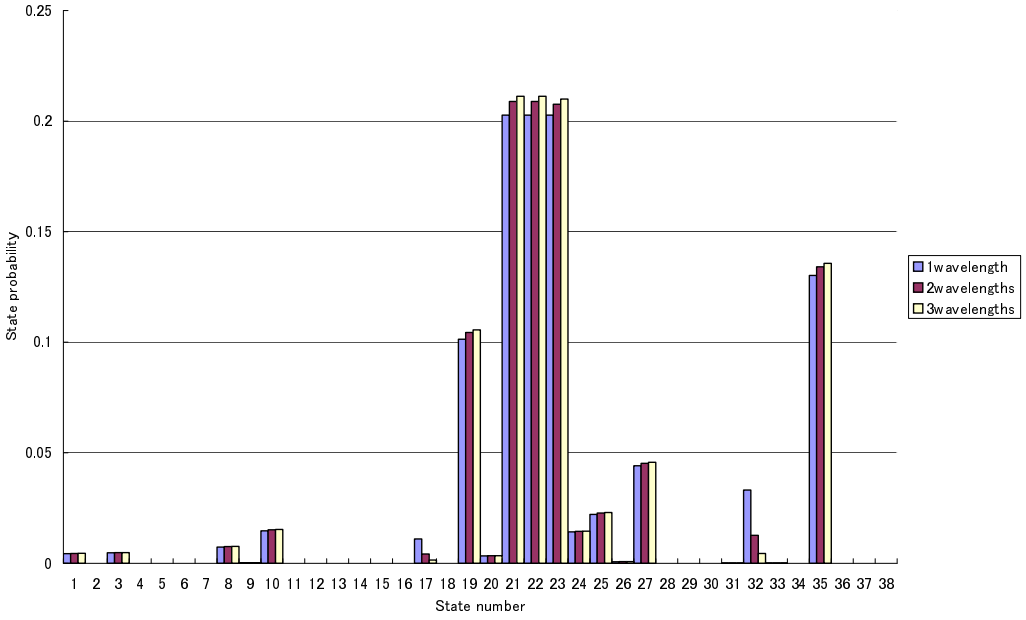


(a) $s = 10^{-3}$

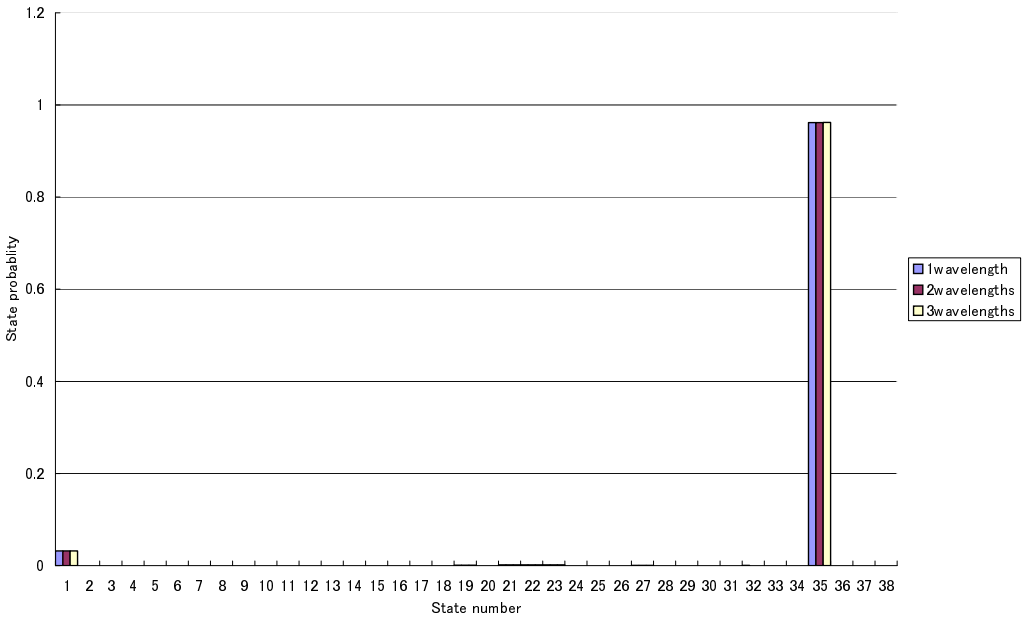


(b) $s = 10^{-6}$

図 15: リング NUMA アーキテクチャの定常状態確率 ($L = 1\text{km}$, $N = 4$)

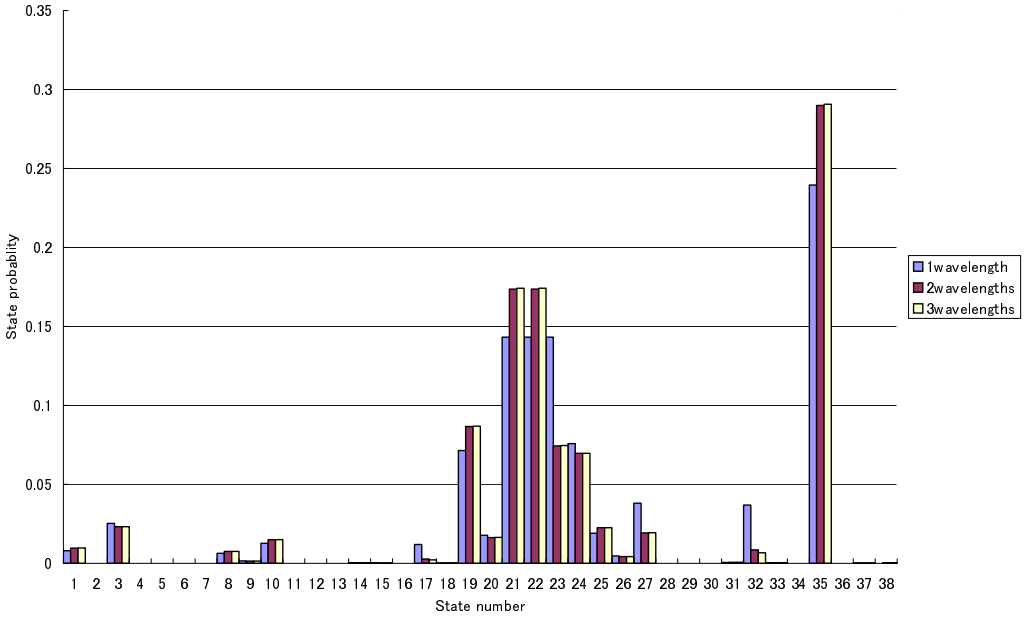


(a) $s = 10^{-3}$

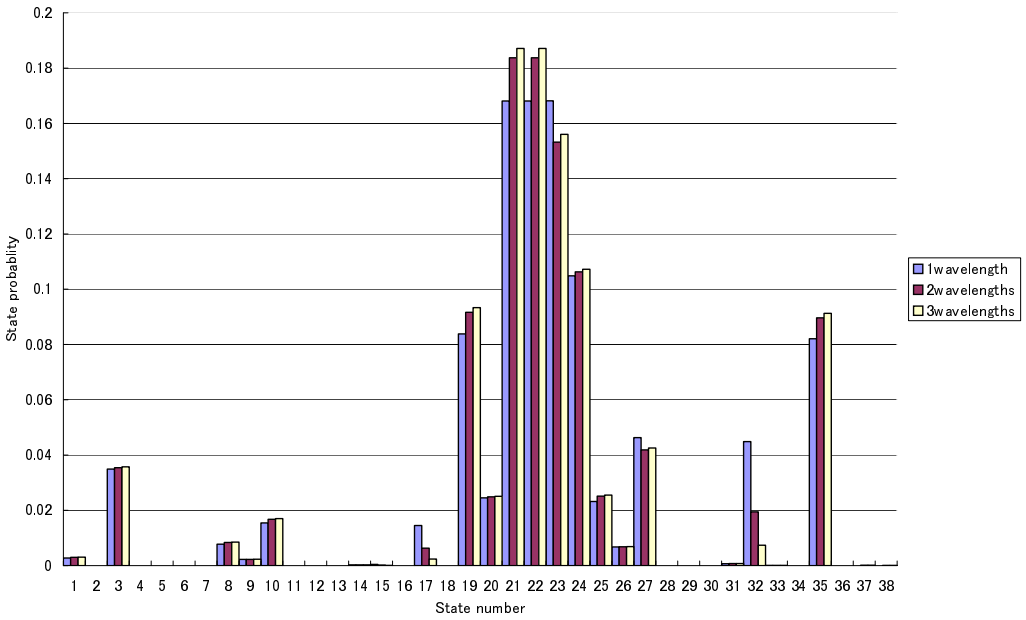


(b) $s = 10^{-6}$

図 16: リング NUMA アーキテクチャの定常状態確率 ($L = 100$km, $N = 4$)



(a) $L = 1\text{km}$



(b) $L = 100\text{km}$

図 17: リング NUMA アーキテクチャの定常状態確率 ($N = 32$, $s = 10^{-3}$)

- リング NUMA アーキテクチャ
  $V = \{10, 14, 15, 17, 21, 22, 23, 27, 31, 32, 37\}$
  $D_{10} = D_{23} = D_{27} = 32 \times N$ bit
  $D_{14} = D_{21} = D_{22} = D_{31} = 32$ bit
  $D_{15} = D_{17} = D_{32} = D_{37} = 32 + 4 \times 8 \times 10^3$ bit

図 18，19 に平均メモリアクセス時間を示す．この数値例におけるパラメータ領域では，リング UMA，リング NUMA アーキテクチャのネットワーク利用率は非常に低く，10^{-8} 以下になっている．ノード計算機は，キャッシュ一貫性制御の処理を行うときにネットワークを利用する．しかし，キャッシュヒット率が高く，共有メモリへのアクセスは大部分が読み込みアクセスであるため，キャッシュ一貫性プロトコルの処理はバースト的に発生するだけである．さらにリング長が長くなると，伝播遅延が大きくなり，ノード計算機のデータ待ちの時間が長くなるため，ネットワーク利用率は低くなる．また，ネットワーク利用率が非常に低いため，ネットワーク帯域には十分な空きがあると考えられる．

リング UMA アーキテクチャのネットワーク利用率とリング NUMA アーキテクチャのネットワーク利用率を比較すると，リング NUMA アーキテクチャの方がネットワーク利用率が低くなっている．それはリング NUMA で用いるキャッシュコピー要求メッセージが 4byte と小さいためである．

また，波長数が増えると，ネットワーク利用率は増加する．これは複数の波長を用いてブロードキャストを行っているためと考えられる．しかし，複数の波長を用いた場合においてもネットワーク利用率は非常に低く抑えられている．

4.5.2 平均メモリアクセス時間

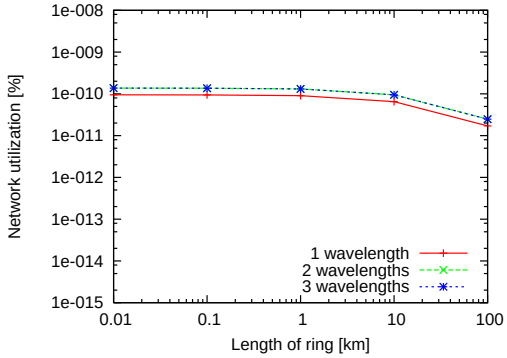
平均メモリアクセス時間 t_{share} を以下のように定義する．

$$t_{share} = r \times t_{sr} + w \times t_{sw} \tag{34}$$

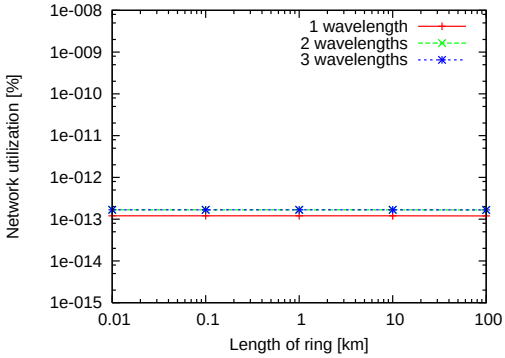
$$t_{sr} = h \times \sum_{u \in R_h} \delta_u \eta_u + (1 - h) \times \sum_{v \in R_m} \delta_v \eta_v \tag{35}$$

$$t_{sw} = h \times \sum_{u \in W_h} \delta_u \eta_u + (1 - h) \times \sum_{v \in W_m} \delta_v \eta_v \tag{36}$$

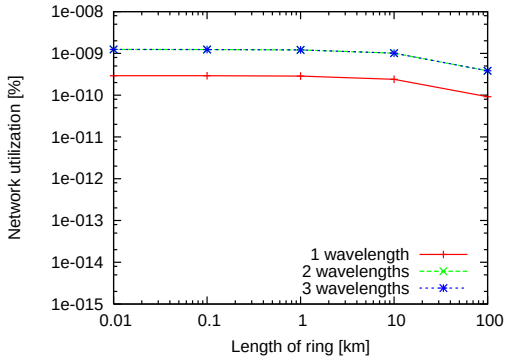
ここで，t_{sr} は平均読み込みアクセス時間，t_{sw} は平均書き込みアクセス時間とする．また，読み込みアクセスがキャッシュヒットしたときに遷移することができる状態の集合を R_h とし，読み込みアクセスがキャッシュミスしたときに遷移することのできる状態の集合を R_m とする．同様に，書き込みアクセスがキャッシュヒットしたときに遷移することができる状



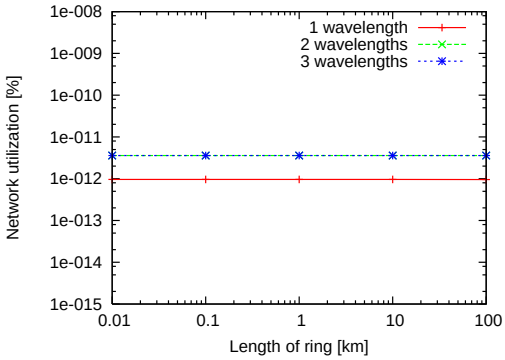
(a) $N = 4$, $s = 10^{-3}$



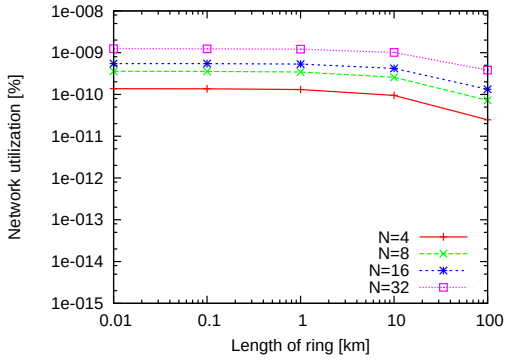
(b) $N = 4$, $s = 10^{-6}$



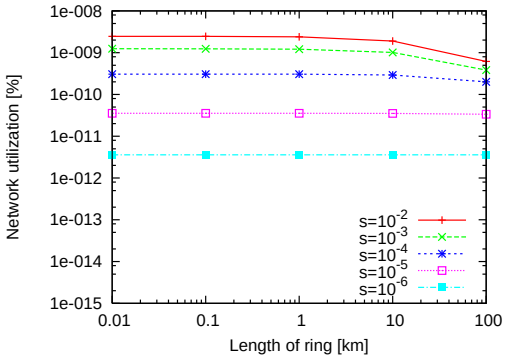
(c) $N = 32$, $s = 10^{-3}$



(d) $N = 32$, $s = 10^{-6}$

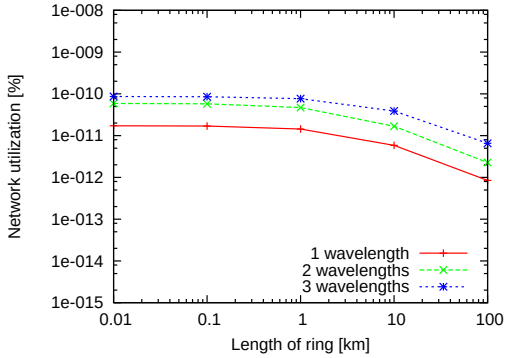


(e) 3 波長, $s = 10^{-3}$

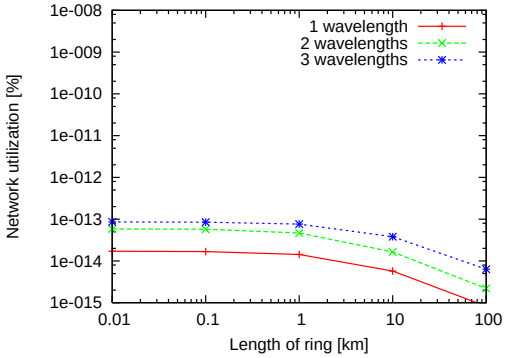


(f) 3 波長, $N = 32$

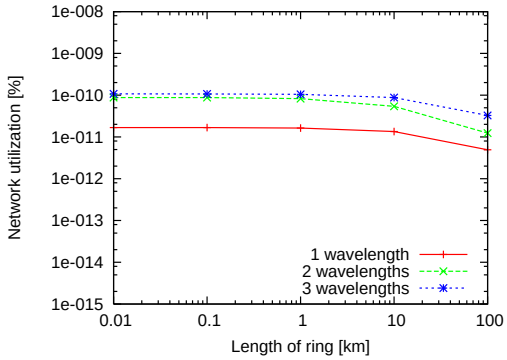
図 18: リング UMA アーキテクチャのネットワーク利用率



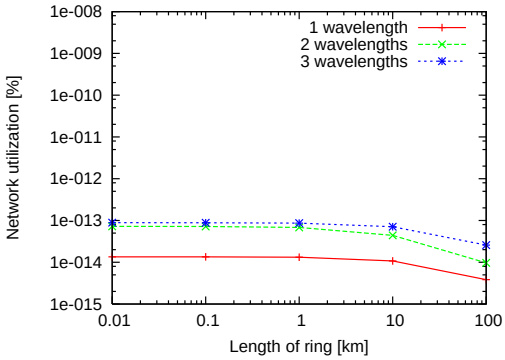
(a) $N = 4$, $s = 10^{-3}$



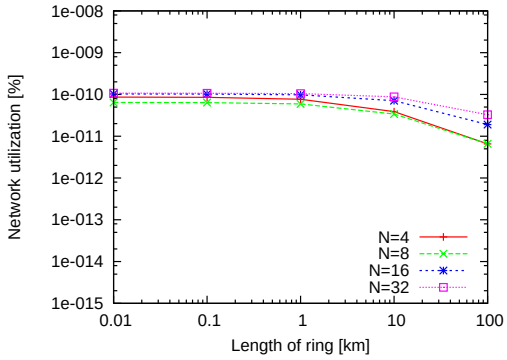
(b) $N = 4$, $s = 10^{-6}$



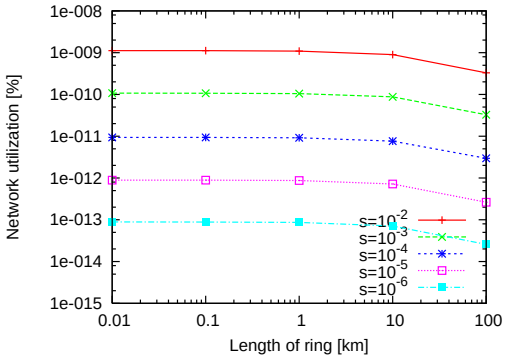
(c) $N = 32$, $s = 10^{-3}$



(d) $N = 32$, $s = 10^{-6}$



(e) 3 波長, $s = 10^{-3}$



(f) 3 波長, $N = 32$

図 19: リング NUMA アーキテクチャのネットワーク利用率

態の集合を W_h とし，書き込みアクセスがキャッシュミスしたときに遷移することのできる状態の集合を W_m とする．δ_u は，状態 u が通過される確率とする．t_{sr} と t_{sw} を下に示す．

- リング UMA アーキテクチャ

$$
\begin{aligned}
t_{sr} &= \eta_2 + P_B\eta_{22} + h\eta_{31} \\
&\quad +(1-h)(\eta_{23} + P_B\eta_{24} + \eta_{25} + P_d(\eta_{26} + \eta_{29} + \eta_{36}) + P_c\eta_{27} \\
&\qquad + (1 - P_c - P_d)\eta_{28} + P_x\eta_{30} + \eta_{31}) \\
t_{sw} &= \eta_2 + P_B\eta_3 + h(\eta_4 + w(\eta_5 + \eta_6) + r(\eta_7 + \eta_{17} + B\eta_{18} + \eta_{19} + \eta_{20} + \eta_{21})) \\
&\quad +(1-h)(\eta_8 + P_B\eta_9 + \eta_{10} + (1 - c - d)(\eta_{11} + P_x\eta_{12} + \eta_6 + \eta_{35}) \\
&\qquad + c(\eta_{13} + P_x\eta_{16} + \eta_{19} + \eta_{20} + \eta_{21}) \\
&\qquad + d(\eta_{14} + \eta_{15} + P_x\eta_{16} + \eta_{19} + \eta_{20} + \eta_{21}))
\end{aligned}
$$

- リング NUMA アーキテクチャ

$$
\begin{aligned}
t_{sr} &= \eta_2 + P_B\eta_{24} + h\eta_{34} \\
&\quad +(1-h)(\eta_{25} + P_B\eta_{26} + \eta_{27} + P_d(\eta_{28} + \eta_{31} + \eta_{38}) + P_c\eta_{29} \\
&\qquad + (1 - P_c - P_d)\eta_{30} + \eta_{32+}P_x\eta_{33} + \eta_{34}) \\
t_{sw} &= \eta_2 + P_B\eta_3 + h(\eta_4 + w(\eta_5 + \eta_7) + r(\eta_6 + \eta_{19} + P_B\eta_{20} + \eta_{21} + \eta_{22} + \eta_{23})) \\
&\quad +(1-h)(\eta_8 + P_B\eta_9 + \eta_{10} + (1 - c - d)(\eta_{11} + \eta_{17} + P_x\eta_{18} + \eta_5 + \eta_7) \\
&\qquad + c(\eta_{12} + \eta_{15} + P_x\eta_{16} + \eta_{21} + \eta_{22} + \eta_{23}) \\
&\qquad + d(\eta_{13} + \eta_{14} + \eta_{15} + P_x\eta_{16} + \eta_{21} + \eta_{22} + \eta_{23}))
\end{aligned}
$$

図 20，21 に平均メモリアクセス時間を示す．リング UMA，リング NUMA アーキテクチャの平均メモリアクセス時間はほぼ同様の結果となっている．リング長が長くなると，ノード数，共有メモリへのメモリアクセス頻度によらず平均メモリアクセス時間は大きくなっている．一方，リング長が 1km 未満の場合，平均メモリアクセス時間が小さく，高速なメモリアクセスが可能となっている．

次に波長数が異なる場合の結果について述べる．ノード数が小さい場合は，波長数による平均メモリアクセス時間にほとんど差は出ていない．これはデータ共有の遅延時間において大部分を伝播遅延が占めるからである．一方，ノード数が多い場合，波長数による差が大きく出ている．特にリング長が短い場合，複数波長を用いることによりノードの処理遅延を低く抑えることができ，結果として平均メモリアクセス時間が 20%ほど減少している．複数波長を用いた場合でもリング長が長くなると，平均メモリアクセス時間は大きくなる．これ

は，トポロジとして，リングトポロジを採るため複数波長を用いても伝播遅延を短くすることができないからである．これはトポロジとしてリングトポロジを用いているからである．

4.5.3 計算スループット

計算スループットを以下のように定義する．また，計算スループットの単位はMIPS（Million Instructions Per Second）とする．

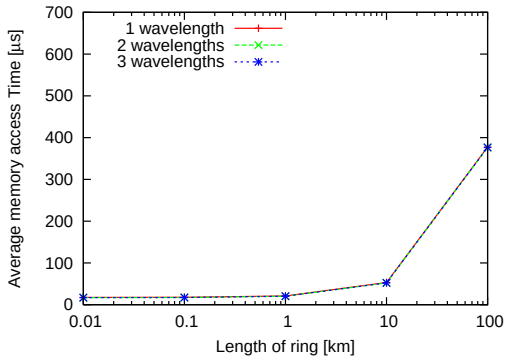
$$\text{Throughput} = \frac{N}{0.8 \times \eta_1 + 0.2 \times s \times t_{share} + 0.2 \times (1-s) \times t_{private}} \tag{37}$$

ここで $t_{private}$ はローカルメモリにアクセスする平均時間である．$t_{private}$ は以下のようになる．

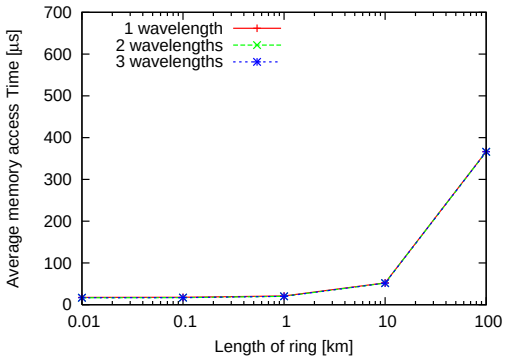
$$t_{private} = h \times t_1 + (1-h) \times (t_1 + t_2) \tag{38}$$

図 22，23 に計算スループットを示す．リング UMA，リング NUMA アーキテクチャの計算スループットはほぼ同様の結果となっている．

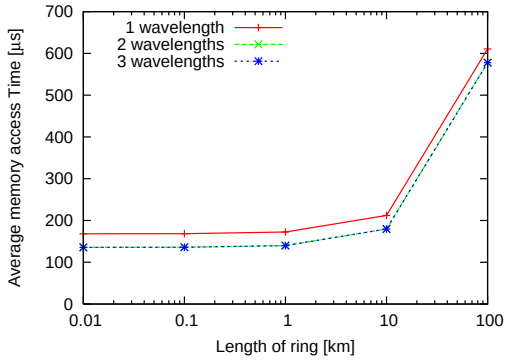
リング長が短い場合，複数波長を用いるとノード数が増えるにしたがって計算スループットは向上しており，リング長 1km，ノード数 32 の場合には約 15%向上している．しかし，計算スループットの数値としては 1000MIPS にも達しておらず，共有メモリへのアクセス頻度が高い並列計算では十分な性能は得られない．また，リング長が長い場合や共有メモリアクセス頻度が低い場合は文献 [5] で設計された 1 波長を用いるリング UMA，リング NUMA アーキテクチャと同じ性能となっている．これはノードの処理遅延に比べてデータの伝播遅延の割合が大きく，リングトポロジでは波長数を増やしても長いリング長で性能のよい計算ループットを得ることが難しい．



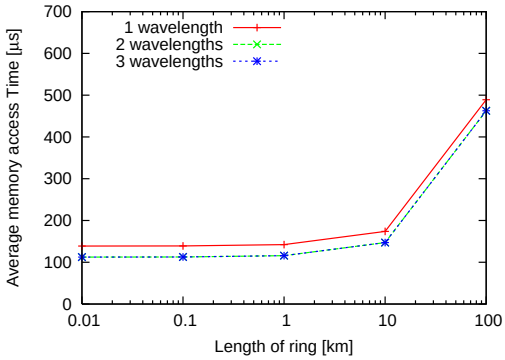
(a) $N = 4$, $s = 10^{-3}$



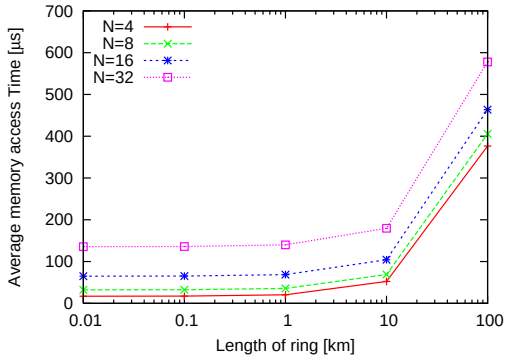
(b) $N = 4$, $s = 10^{-6}$



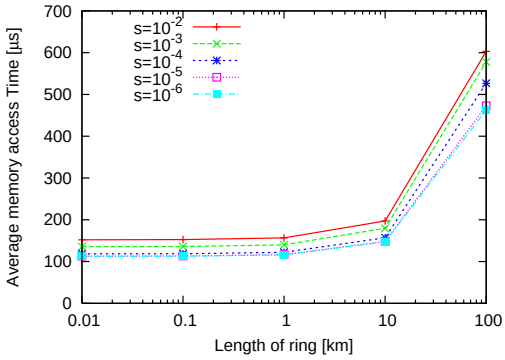
(c) $N = 32$, $s = 10^{-3}$



(d) $N = 32$, $s = 10^{-6}$

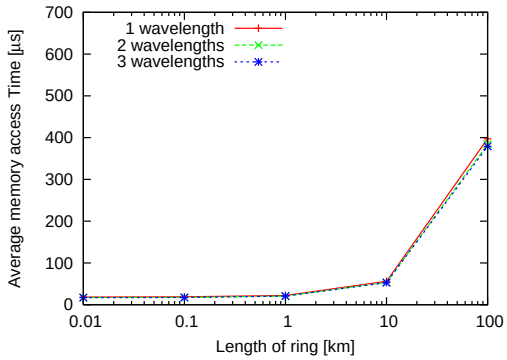


(e) 3 波長, $s = 10^{-3}$

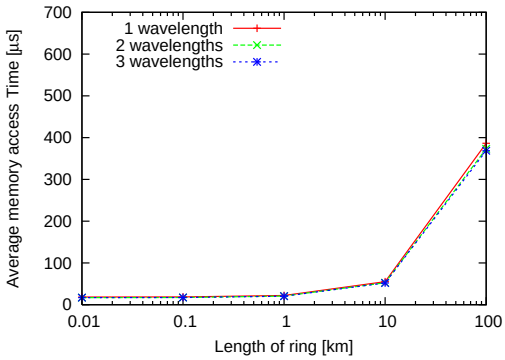


(f) 3 波長, $N = 32$

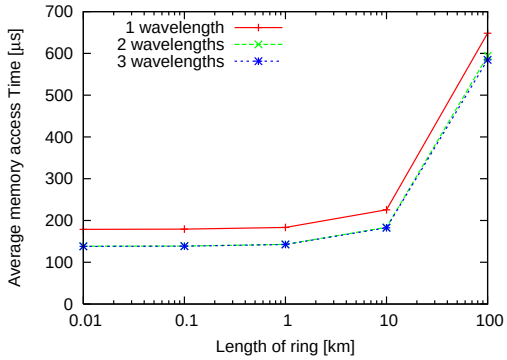
図 20: リング UMA アーキテクチャの平均メモリアクセス時間



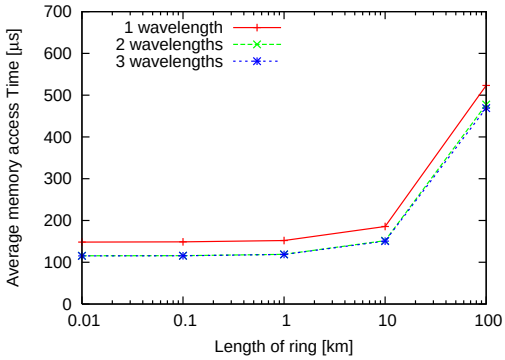
(a) $N = 4$, $s = 10^{-3}$



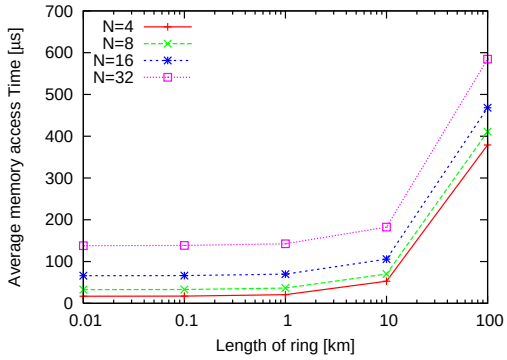
(b) $N = 4$, $s = 10^{-6}$



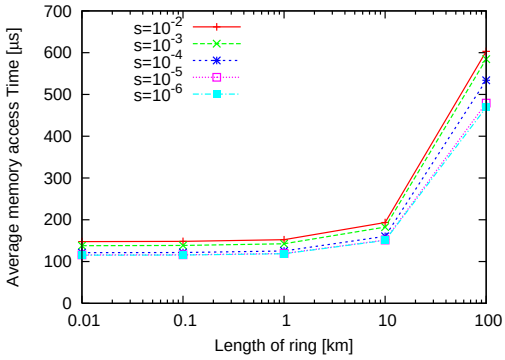
(c) $N = 32$, $s = 10^{-3}$



(d) $N = 32$, $s = 10^{-6}$

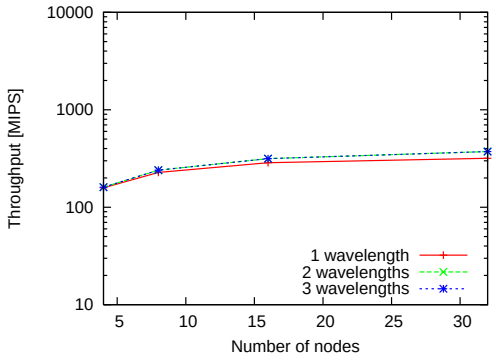


(e) 3 波長, $s = 10^{-3}$

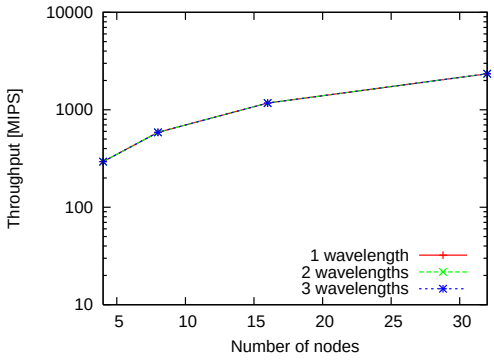


(f) 3 波長, $N = 32$

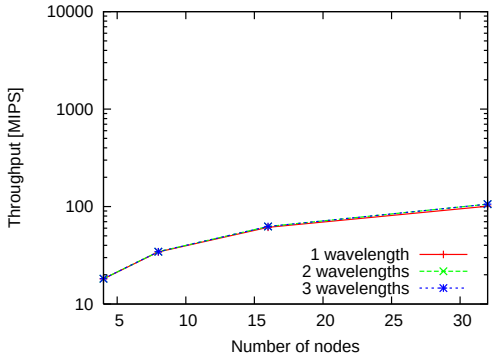
図 21: リング NUMA アーキテクチャの平均メモリアクセス時間



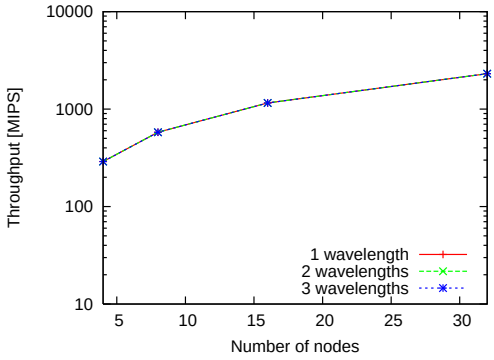
(a) $L = 1\text{km}$, $s = 10^{-3}$



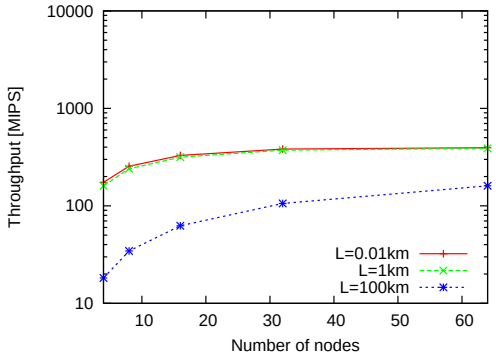
(b) $L = 1\text{km}$, $s = 10^{-6}$



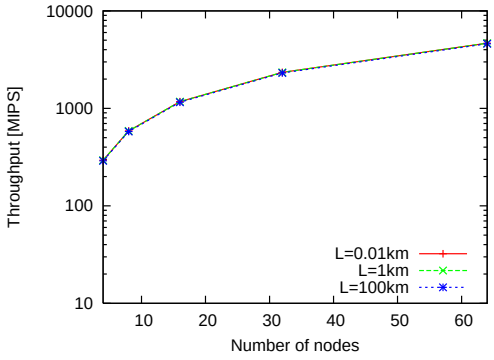
(c) $L = 100\text{km}$, $s = 10^{-3}$



(d) $L = 100\text{km}$, $s = 10^{-6}$

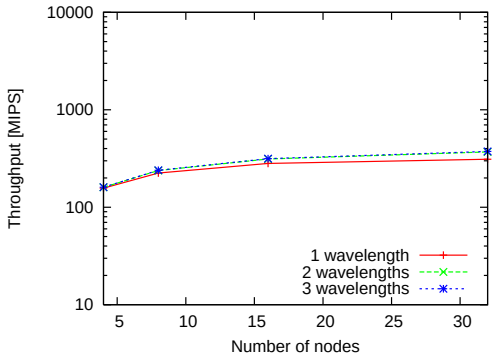


(e) 3 波長, $s = 10^{-3}$

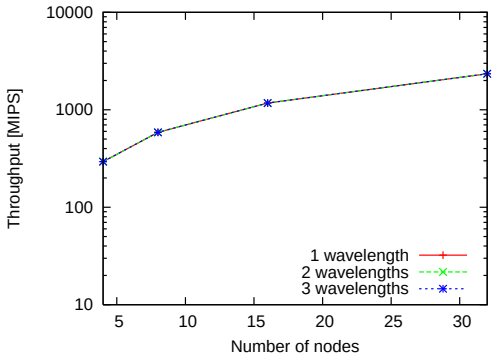


(f) 3 波長, $s = 10^{-6}$

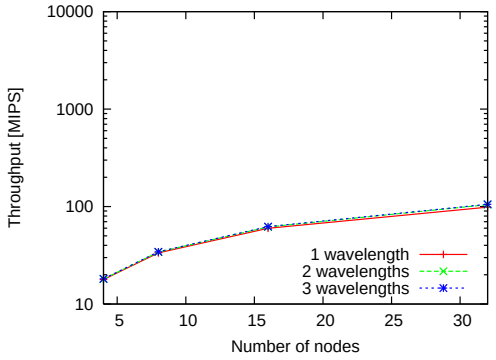
図 22: リング UMA アーキテクチャの計算スループット



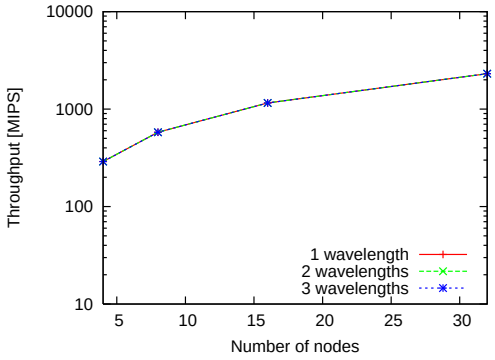
(a) $L = 1\text{km}$, $s = 10^{-3}$



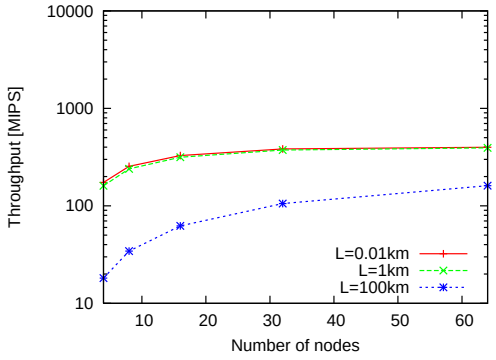
(b) $L = 1\text{km}$, $s = 10^{-6}$



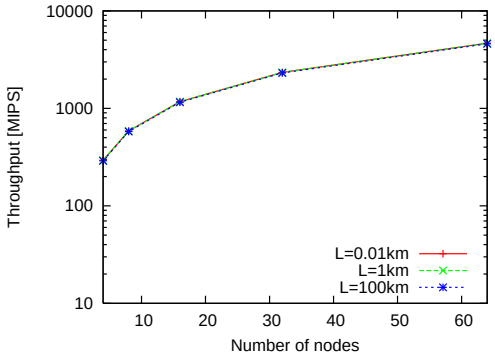
(c) $L = 100\text{km}$, $s = 10^{-3}$



(d) $L = 100\text{km}$, $s = 10^{-6}$



(e) 3 波長, $s = 10^{-3}$



(f) 3 波長, $s = 10^{-6}$

図 23: リング NUMA アーキテクチャの計算スループット

5　まとめ

本報告では，光リングネットワークを用いたλコンピューティング環境に適した共有メモリアーキテクチャの設計と評価を行った．文献 [5] で設計された単一波長リングトポロジを用いる共有メモリアーキテクチャに対し，使用する波長数を増やした場合の共有メモリアーキテクチャについて，その性能の評価を行った．リングトポロジにおいてはリングを一周回るのに要する伝播遅延を短くすることはできないため，ノードによる処理遅延を減らすことに着目した．すなわち，単一波長リングの場合，送信したメッセージは全てのノード計算機を経由するため，データ共有に要する遅延時間は大きくなる．そこで複数の波長を設定し，経由するノード数を減らすことで，ノードによる処理遅延を低くした．

設計したリングトポロジを用いる共有メモリアーキテクチャをモデル化し，セミ・マルコフ過程を用いて解析を行った結果，リング長が短く，共有メモリへのアクセス頻度が高い場合に性能が向上することが明らかになった．しかしながら，共有メモリへのアクセス頻度が高い並列計算では，リングトポロジを用いる共有メモリアーキテクチャに複数波長を用いてもデータの伝播遅延がボトルネックとなり，十分な計算性能を得ることが難しいと考えられる．一方，共有メモリへのアクセス頻度が低い並列計算においては，ノード数が多い場合には 10000MIPS 以上の性能を得ることができる．しかし，波長数の違いによる性能の差はほとんどないため，複数波長を用いるメリットがあまりないことがわかった．

今後の課題として，リングトポロジを用いる共有メモリアーキテクチャではノード計算機を複数のリングに分割する手法を用いて伝播遅延を低く抑える必要がある．また，波長数をある程度に抑えながら伝播遅延が小さくなるメッシュトポロジを構成した場合の共有メモリアーキテクチャの性能の評価を行う必要がある．

謝辞

本報告を終えるにあたり，御指導，御教授をいただきました大阪大学大学院情報科学研究科の村田正幸教授に深く感謝いたします．また，本報告において直接御指導いただきまし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，様々な相談に乗っていただき，多くの助言をいただきましたことを，心よりお礼申し上げます．本報告において，多大な御協力をいただいた大阪大学大学院情報科学研究科の藤本典幸助教授に心からお礼申し上げます．

また，平素から適切なご助言をいただいた大阪大学大学院情報科学研究科の若宮直紀助教授，荒川伸一助手，大阪大学サイバーメディアセンターの長谷川剛助教授に深く感謝いたします．最後に，本報告の作成にあたり，多くの相談に乗っていただき，支えていただいた井本舞氏，合田圭吾氏を初めとする村田研究室，中野研究室の皆様方に心からお礼申し上げます．

参考文献

[1] 井本 舞, 合田 圭吾, 馬場 健一, 村田 正幸, "λ コンピューティング環境における OpenMP ライブラリのためのデータ共有機構の設計," **電子情報通信学会技術報告** *(PN2006-28)*, vol. 106, pp. 19–24, Oct. 2006.

[2] M. Imoto, E. Taniguchi, K. Baba, and M. Murata, "Implementation and Evaluation of MPI Library with Globus Toolkit for Establishing λ Computing Environment," in *Proceedings of 6th Asia-Pacific Symposium on Information and Telecommunication Technologies*, pp. 421–426, November 2005.

[3] 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 29th European Conference on Optical Communication*, vol. 4, pp. 978–979, Sept. 2003.

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

[5] E. Taniguchi, "Design and evaluation of shared memory architecture for WDM-based λ computing environmnet," Master's thesis, Graduate School of Informantion Science and Technology, Osaka University, Feb. 2006.

[6] A. K. Somani, *Survivability and Traffic Grooming in WDM Optical Networks*. CAMBRIDGE, 2005.

[7] O. Gerstel and S. Kutten, "Dynamic wavelength allocation in all-optical ring networks," in *Proceedings of ICC '97*, pp. 432–436, 1997.

[8] O. Gerstel, P. Lin, and G. Sasaki, "Wavelength assignment in a WDM ring to minimize cost of embedded SONET rings," in *Proceedings of IEEE INFOCOM '98*, pp. 94–101, Apr. 1998.

[9] Oudewijn R. Haverkort, *PERFORMANCE OF COMPUTER COMMUNICATION SYSTEMS*. WILEY, 1998.

[10] Kazuki Joe and Jun Naito, “An Analytic Model for the Performance of the ASURA Cluster using a Semi–Markov Processing,” *Technical Report of IPSJ* (ARC–1992–097), pp. 65–72, 1992. (in Japanese).

[11] 合田 圭吾, 井本 舞, 藤本 典幸, 馬場 健一, 村田 正幸, “λ コンピューティング環境における OpenMP ライブラリの設計と実装,” **電子情報通信学会技術報告** *(PN2006-40)*, vol. 106, pp. 5–8, Dec. 2006.