光ネットワーク環境における MPI集団通信

滝澤 真一朗†
松岡 聡†,††
中田 秀基†††,†

†: 東京工業大学
††: 国立情報学研究所
†††: 産業技術総合研究所

背景

- 光ネットワーク技術の進歩
  - ファイバの多重化（WDM、DWDM）により、Tbpsオーダーの超広帯域を実現
  - 将来の基盤ネットワークとして期待
- グリッドの基盤ネットワークとして注目
  - OptIPuter Project [L. Smarr et al.]
    - 資源が光ネットワーク網に接続されたλグリッドを提唱
    - データインテンシブアプリケーション実行環境を提供
    - λグリッドに適した分散ストレージ、プロトコルの開発

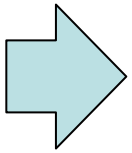
既存ネットワークと光ネットワークの比較

	既存ネットワーク	光ネットワーク
信号	電気	光
ハ		
リンク使用方法	共有	
通信前処理	なし	波長パスの確立（数msのコスト）
通信後処理	なし	波長パスの開放（数msのコスト）
接続数制限	なし	ネットワークが提供できる波長数は有限

通信を頻繁に行うアプリケーションに影響！

光ネットワーク上でのMPIアプリケーション

- Message Passing Interface（MPI）
  - メッセージパッシングライブラリとして広く利用
    - 将来、ネットワーク環境が置き換わっても利用され続けると想定
  - 集団通信機能をライブラリレベルで提供
- 光ネットワーク環境では通信前後で波長パス確立・解放が必要
  - 単純には、通信のたびに十数msの時間増加
  - 集団通信での影響大



アプリケーション実行時間増加を抑えるには、
波長パス確立・解放回数の削減、コストの隠蔽が必要

目的と成果

- 目的
  - 光ネットワーク上での波長パス確立・解放遅延を隠蔽・削減するMPI集団通信の提供
- 成果
  - ノードの持つポート数（最大同時接続先数）に応じた波長パス確立・解放回数削減手法を提案
  - 数値実験、およびシミュレーションにより提案手法の有効性を確認

MPIアプリケーション分析

NPB：MG[A, 16] on GbEther

MPI関数	平均実行時間	呼び出し回数
Allreduce	358 μs	88
Barrier	16.5ms	6
Bcast	16 μs	6
Irecv	4 μs	664
Reduce	299 μs	1
Send	363 μs	654
Wait	54 μs	664

総実行時間：0.79秒

光ネットワーク上では・・・

- 波長パス確立・解放コストを10msとする
- 通信のたびに波長パス確立・解放を行うと・・・
  - Send 654回
    - + 6.5秒 (654 x 0.01)
  - Irecv,Wait 664回
    - + 数秒 （最大6.6秒）
  - 集団通信 101回
    - + 101 x 0.01 x (p2p呼出回数) 秒

10～20秒ほどの増加

光ネットワーク上では、大幅な実行時間の増加が見込まれる！

提案手法

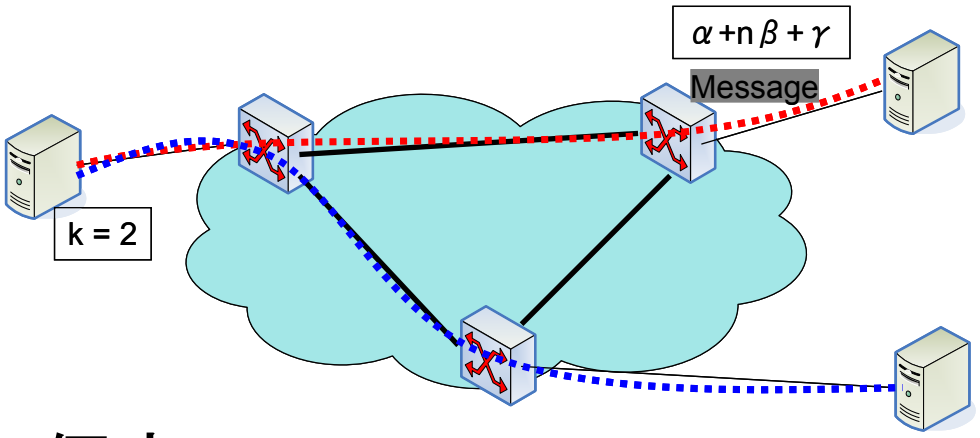
- 集団通信において、通信とは独立して波長パス確立・解放を行う手法
  - 複数宛先への波長パスを同時に確立・解放することによる回数削減
  - 同一宛先へ複数回メッセージ送信する場合には波長パスを長期利用することにより回数削減

集団通信アルゴリズム

- 提案手法を、トポロジを考慮した集団通信アルゴリズムに適応
  - Linear、 Ring、 Recursive Doubling、 Binomial Tree
- いずれも標準的なMPI実装で使用
  - MPI_Allgather (>= MPICH 1.2.6)
    - Ring (< 512KB) + Recursive Doubling (>= 512KB)
- そのほかのアルゴリズムはこれらの応用
  - 組み合わせ、 ステップが逆、 宛先プロセスが異なる

提案手法を適応させた、
各アルゴリズムの実行コスト式を作成・評価

光ネットワークモデル



仮定

- 1ノード1MPIプロセス
- ポート数は全ノードで等しい
- 1パスで全二重通信
- 複数ノードへの同時パス確立コストは、単一ノードへのパス確立コストと等しくγ

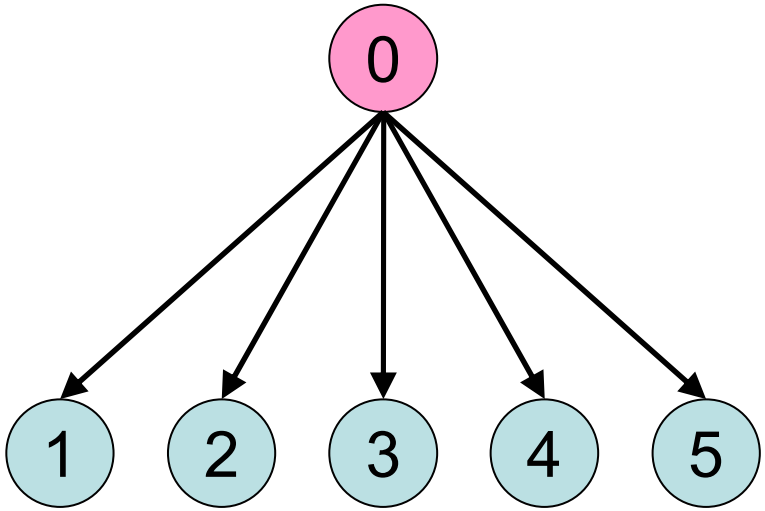
ネットワークプロパティ

p	プロセス数
k	ポート数 （ノードが同時接続できる宛先数）
n	メッセージサイズ
α	通信遅延
β	単位バイトあたりの通信時間
γ	波長パス確立・解放時間

通信コスト

- 波長パス未確立
  - $\alpha + n\beta + \gamma$
- 波長パス確立済み
  - $\alpha + n\beta$

Linear:
動作、既存ネットワークでのコスト



- Rootが持つnバイトのメッセージを他のプロセスに逐次に転送
- ステップ数
  $p-1$
- ステップあたりの通信コスト
  $\alpha + n\beta$

総コスト: $(p-1)(\alpha + n\beta)$

Linear：光ネットワーク環境

- 通信のたびにパスを確立・解放（Naiveな手法）

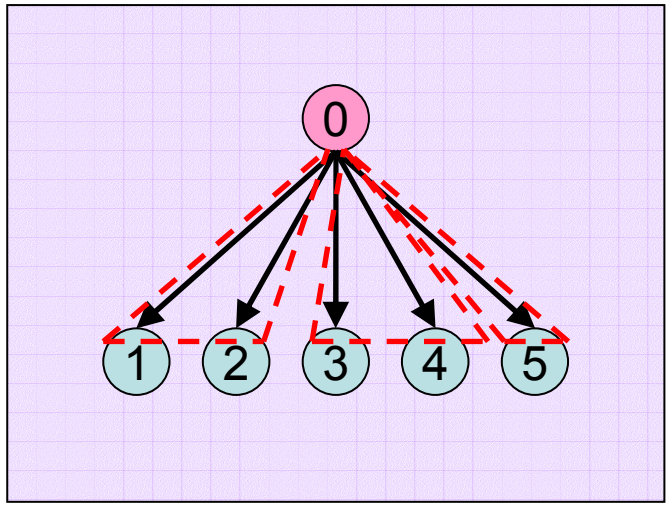
$$(p-1)(\alpha+n\beta+\gamma)$$

- 提案
  - 通信とは独立して、ポート数分ずつパスを確立・開放

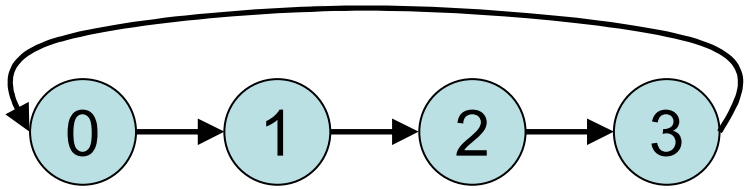
$$(p-1)(\alpha+n\beta)+\left\lceil\frac{p-1}{k}\right\rceil\gamma$$

- 遅延削減量

$$\left((p-1)-\left\lceil\frac{p-1}{k}\right\rceil\right)\gamma$$



Ring:
動作、既存ネットワークでのコスト



- 各プロセスが持つn/pバイトのメッセージをリング型の通信経路を用いて互いに交換
- ステップ数

$$p-1$$

- ステップあたりの通信コスト

$$\alpha+\frac{n}{p}\beta$$

総コスト: $\left(p-1\right)\left(\alpha+\frac{n}{p}\beta\right)$

Ring：光ネットワーク環境

- Naiveな手法

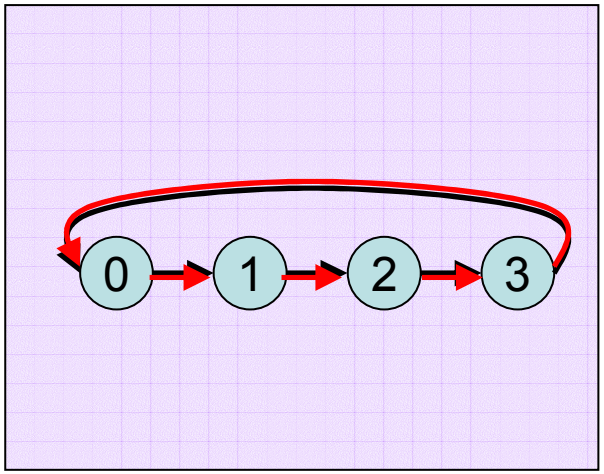
$$2(p-1)\left(\alpha+\frac{n}{p}\beta+\gamma\right)$$

- 提案
  - 複数ポートが存在するときには通信前にパスを張り、リングを構築し、通信後にパスを解放

$$(p-1)\left(\alpha+\frac{n}{p}\beta\right)+\gamma$$

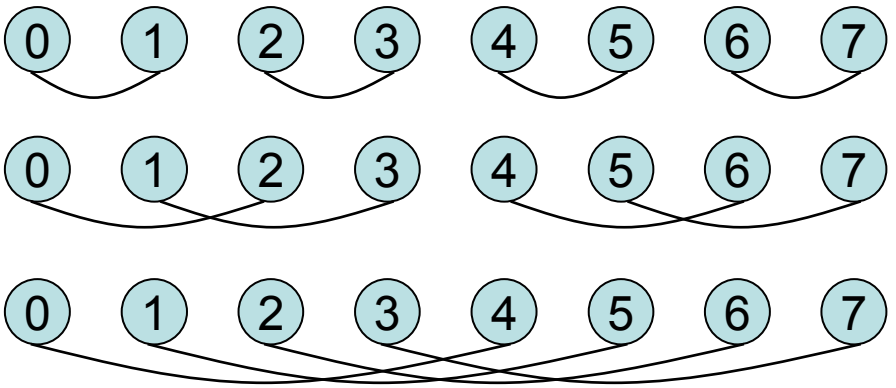
- 遅延削減量

$$(2p-3)\gamma$$



Recursive Doubling：動作、既存ネットワークでのコスト

- 各プロセスが持つn/pバイトのメッセージを2^{i-1}離れたプロセスと互いに交換



- ステップ数

$$\log p$$

- ステップあたりの通信コスト

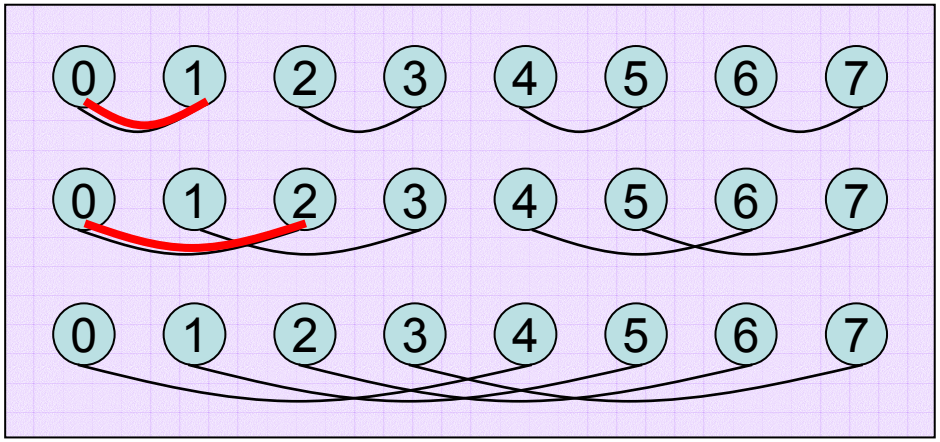
$$\alpha + 2^{i-1}\left(\frac{n}{p}\right)\beta$$

総コスト：$(\log p)\alpha + (p-1)\frac{n}{p}\beta$

Recursive Doubling：光ネットワーク環境

- Naiveな手法

$$(\log p)(\alpha+\gamma)+(p-1)\frac{n}{p}\beta$$



- 提案
  - 通信とは独立してポート数分ずつパスを確立・解放

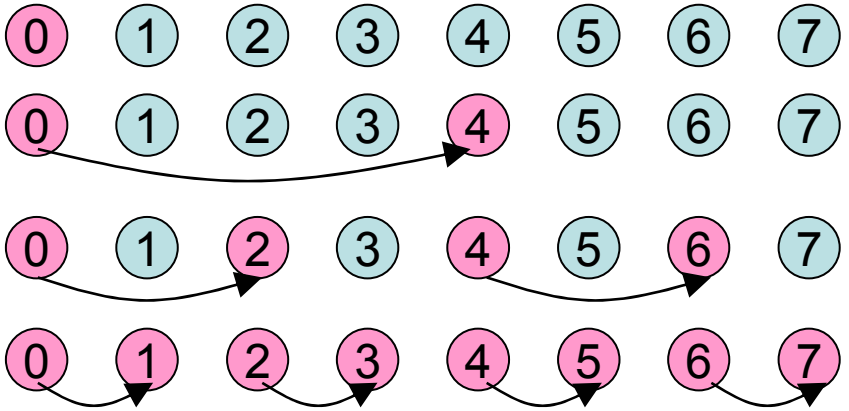
$$(\log p)(\alpha)+(p-1)\frac{n}{p}\beta+\left\lceil\frac{\log p}{k}\right\rceil\gamma$$

- 遅延削減量

$$\left(\log p-\left\lceil\frac{\log p}{k}\right\rceil\right)\gamma$$

Binomial Tree：
動作、既存ネットワークでのコスト

- 二項木を用いて、Rootが持つnバイトのメッセージを他のプロセスへ配布



- ステップ数

$$\log p$$

- ステップあたりの通信コスト

$$\alpha + n\beta$$

総コスト：$(\log p)(\alpha + n\beta)$

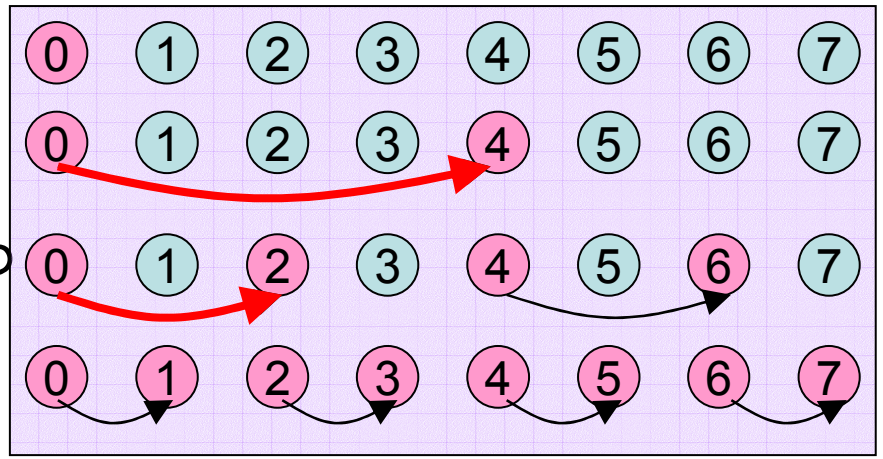
Binomial Tree：光ネットワーク環境

- Naiveな手法

$$(\log p)(\alpha + n\beta + \gamma)$$

- 提案
  - 通信とは独立してポート数分ずつパスを確立・解放

$$(\log p)(\alpha + n\beta) + \left\lceil \frac{\log p}{k} \right\rceil \gamma$$



- 遅延削減量

$$\left(\log p - \left\lceil \frac{\log p}{k} \right\rceil \right) \gamma$$

実環境での集団通信の考察

- 波長リソースに限りがあり、波長パス数は制限
  - パスが張れず、通信が行えない（**輻輳**）
- 集団通信が失敗する場合がある
  - Ringのように同時に利用するパス数が多いアルゴリズムで問題
    - パス確立の再試行機構が必要
- 最適な波長パス割り当てができない場合がある
  - Binomial Treeのようにプロセス間に主従関係があるアルゴリズムで問題
    - パス確立のスケジューリングが必要

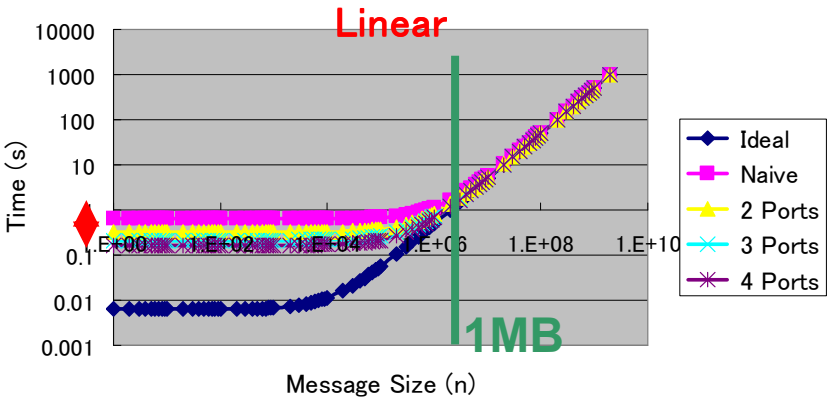
数値実験による評価

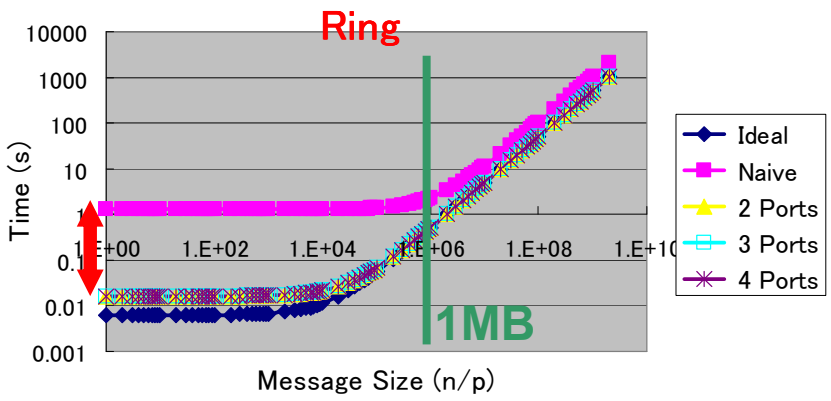
- 各アルゴリズムのコスト式をメッセージサイズを変化させて評価
  - Idealな場合（波長パス確立・解放コスト0）
  - Naiveな手法（通信のたびにパスを確立・解放）
  - ポート数を変化させた提案手法

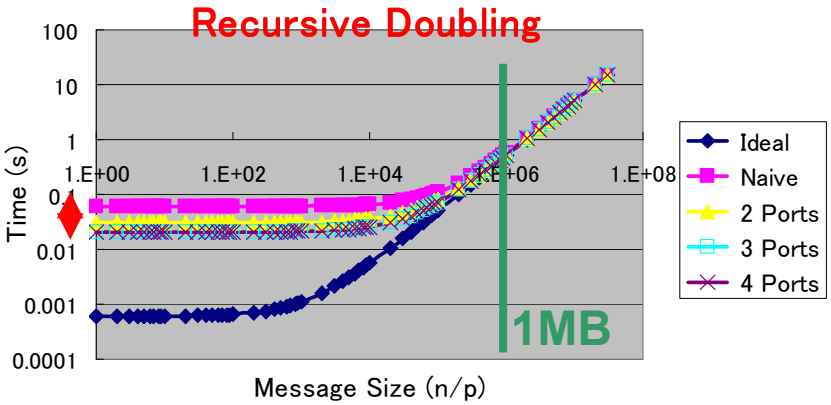
変数設定

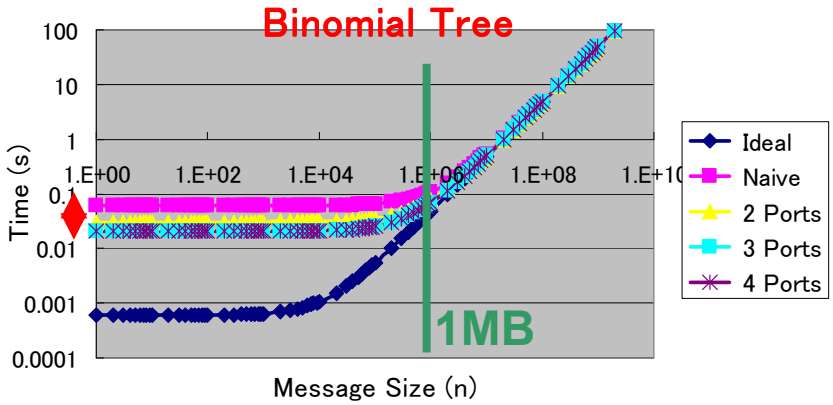
変数	設定値	備考
p	64	プロセス数
k	1,2,3,4	ポート数（1はNaiveと等しい）
α	0.0001	遅延（100μs相当）
β	10^9	バンド幅（1Gbps相当）
γ	0.01	波長パス確立・解放コスト（10ms相当）

数値実験結果：両対数グラフ









1MB以下のメッセージにおいて本手法の有効性が確認できる！

シミュレーションによるアプリケーション実行評価

- Nas Parallel BenchmarksのEPとFTをシミュレータ上で動作させ、実行時間を計算
  - EP : 乗算合成法による一様乱数、正規乱数を生成
  - FT : FFTを用いた3次元編微分方程式の解法
  - 共にクラスA、プロセス数16
  - Idealな場合、Naiveな手法、ポート数を変化させた提案手法を比較

ネットワーク設定

項目	値
バンド幅	1Gbps
通信遅延	100 μ s
光パスによる遅延	10ms

集団通信実装

MPI関数	アルゴリズム
MPI_Allreduce	Recursive Doubling
MPI_Alltoall	Recursive Doubling
MPI_Barrier	Recursive Doubling
MPI_Bcast	Binomial Tree
MPI_Reduce	Binomial Tree

シミュレータ

- MPIアプリケーションイベント列を実行し時間を計算
  - イベント列はCPU処理部、MPI関数実行部の繰り返し
    - MPIアプリケーション実行ログから生成
  - CPU処理部のコストはログデータ値そのもの
  - MPI関数部のコストは通信パターンをシミュレートし計算
- 光ネットワーク環境をシミュレート
  - 通信前に光パスを確立
    - +0.005 秒
  - 通信終了後に光パスを解放
    - +0.005 秒
  - 通信時間

$$\text{通信遅延} + \frac{\text{メッセージサイズ}}{\text{バンド幅}}$$

EP、FTの結果

EP　集団通信5回

手法	実行時間(s)	増加率(%)
Ideal	2.837	0.00
Naive	3.034	6.94
2 Ports	2.934	3.41
3 Ports	2.934	3.41
4 Ports	2.887	1.76

FT　集団通信17回

手法	実行時間(s)	増加率(%)
Ideal	2.001	0.00
Naive	2.666	33.23
2 Ports	2.328	16.53
3 Ports	2.328	16.35
4 Ports	2.158	7.83

- 集団通信回数増加による実行時間増加率の上昇
- 利用可能ポート数が多いほど実行時間が短縮

関連研究(1/2)

- 光インターフェース、電気インターフェースを持ったノード群からなる計算機システム[Barkerら'05]
  - 一対一通信において、電気インターフェースで転送されるメッセージ量が増加すると光インターフェースを用いた通信に変更
  - 集団通信は常に電気ネットワークのみが使用される
- 光ネットワーク上にMPICH-G2 [Karonis et al. '03]を移植し、MPIアプリケーションを実行[井本ら'05]
  - メッセージ交換は共有メモリインターフェースを通じた光ネットワーク通信
  - トポロジをリング型に固定

関連研究(2/2)

- 光ネットワーク上での集団通信[Afsahi et al '02]
  - ポート数に応じたアルゴリズムを提案し、コスト式を計算
  - 効率は良いが、実装が容易で無い
  - 輻輳発生時の考察がされていない
- MPIアプリケーションにおける通信発生予測アルゴリズム[Afsahi et al '99]
  - 通信発生を予測し、事前にコネクションを確立
  - 通信発生のためのさまざまなアルゴリズムを提案
  - CPU処理時間(μ秒オーダ)に対し、コネクション確立時間(ミリ秒オーダ)のため、隠蔽はできない

まとめと今後の課題

- まとめ
  - ポート数に応じて同時コネクション確立・解放を行うことにより、実行時間削減を行うMPI集団通信の設計
  - 数値実験、シミュレーションにより実行時間削減を確認
- 課題
  - 輻輳存在下での集団通信アルゴリズムの詳細検討
  - 実装し、実環境での評価
  - より実環境に近い条件でのアルゴリズムを検討
    - 実環境では他のアプリケーションにより、波長リソース、ポートが消費されうる