

# 仮想計算機を用いた グリッド上でのMPI実行環境

立菌 真樹†

中田 秀基††,†

松岡 聡†,†††

†: 東京工業大学

††: 産業技術総合研究所

†††: 国立情報学研究所



# 目次

1. はじめに
2. 提案と設計
3. 実装
4. 評価
5. 考察
6. おわりに

# 背景

- 遊休計算機利用とは
  - オフィスやキャンパスの遊休計算資源を有効活用
  - 計算機所有者の使用していない時間帯にジョブを実行
    - 所有者の復帰時には、その使用を妨げてはならない
    - 実行していたジョブは、中止もしくは他の資源上へマイグレーション
  - 高いスループットを容易に実現可能
- 遊休計算機利用の一般化
  - スーパーコンピュータやクラスタコンピュータに比べて低コストで実現可能
  - e.g. SETI@HOME

# 背景

- MPIを遊休計算機上で実行できれば、大幅な計算効率の向上が実現可能
  - MPIにおける学術計算は、非常に多くの計算資源を必要とする場合が多い
- Condor[Livny *et al.* '88 ]
  - 遊休資源の利用が可能なジョブスケジューラ
    - チェックポイントによるマイグレーションの実現
  - MPIの実行時にはマイグレーションに非対応

# MPIを遊休計算機上で実行する際の問題点

- MPIのマイグレーションは難しい
  - 通信タイミングとの整合性
  - ロールバックにより他のプロセスにも影響
- 多数の遊休資源を扱う場合、複数のネットワークが混在
  - 一般的なMPI実装では、実行マシンのIPアドレスの変更を許さない
  - WANを経由する場合、高遅延低バンド幅なリンクを通過

# 目的と成果



## ● 目的

- 容易なMPIマイグレーションを実現し、それを利用した遊休計算機利用の実現

## ● 成果

- 仮想計算機XenとVPNを用い、グリッド環境でMPIのマイグレーションを実現
- 遊休資源利用システムのプロトタイプ実装
- 仮想計算機でのMPI実行性能の評価と考察



# 目次

1. はじめに
2. 提案と設計
3. 実装
4. 評価
5. 考察
6. おわりに

# 提案 - 遊休計算機上でMPI実行を行える環境

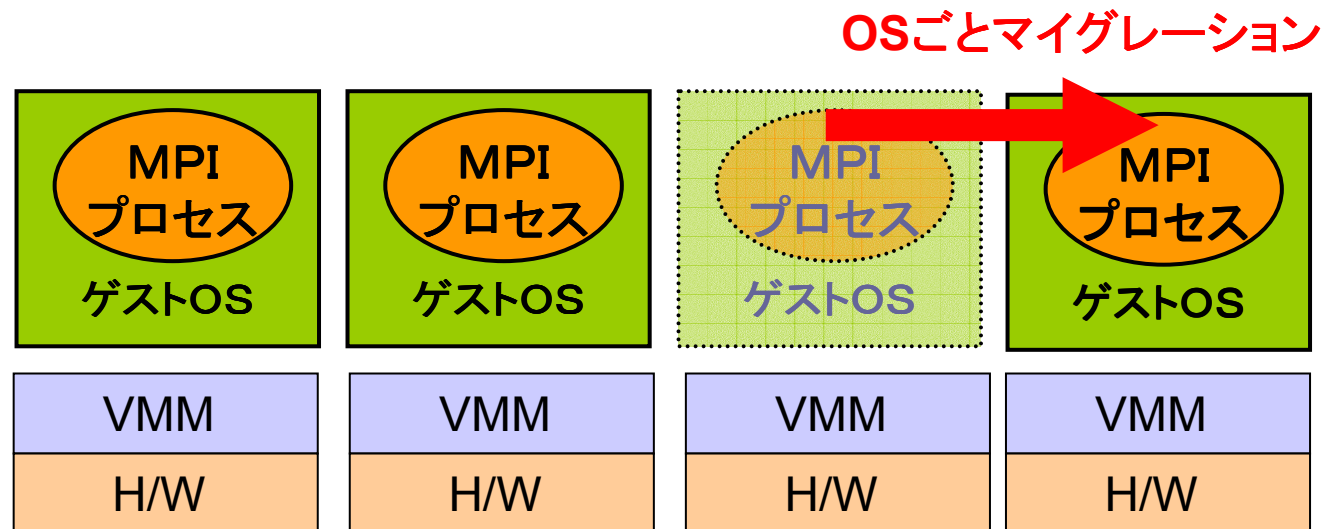
- 仮想計算機を用いた低コストなMPIマイグレーション
  - 仮想計算機により、MPIの動作するゲストOSごとマイグレーション
- 計算資源の監視を行い、遊休計算機利用におけるジョブ実行を実現するシステム
  - マイグレーションはユーザーに対して透過に行われる
- グリッドなどの広域に分散する資源を効率的に使用
  - ネットワークの違いをVPNの使用により吸収
  - ネットワークトポロジーを意識したマイグレーションの実行

→グリッド上の遊休資源上で既存のMPI実装  
利用したMPIの実行が可能に



# 仮想計算機を用いたMPIのマイグレーション

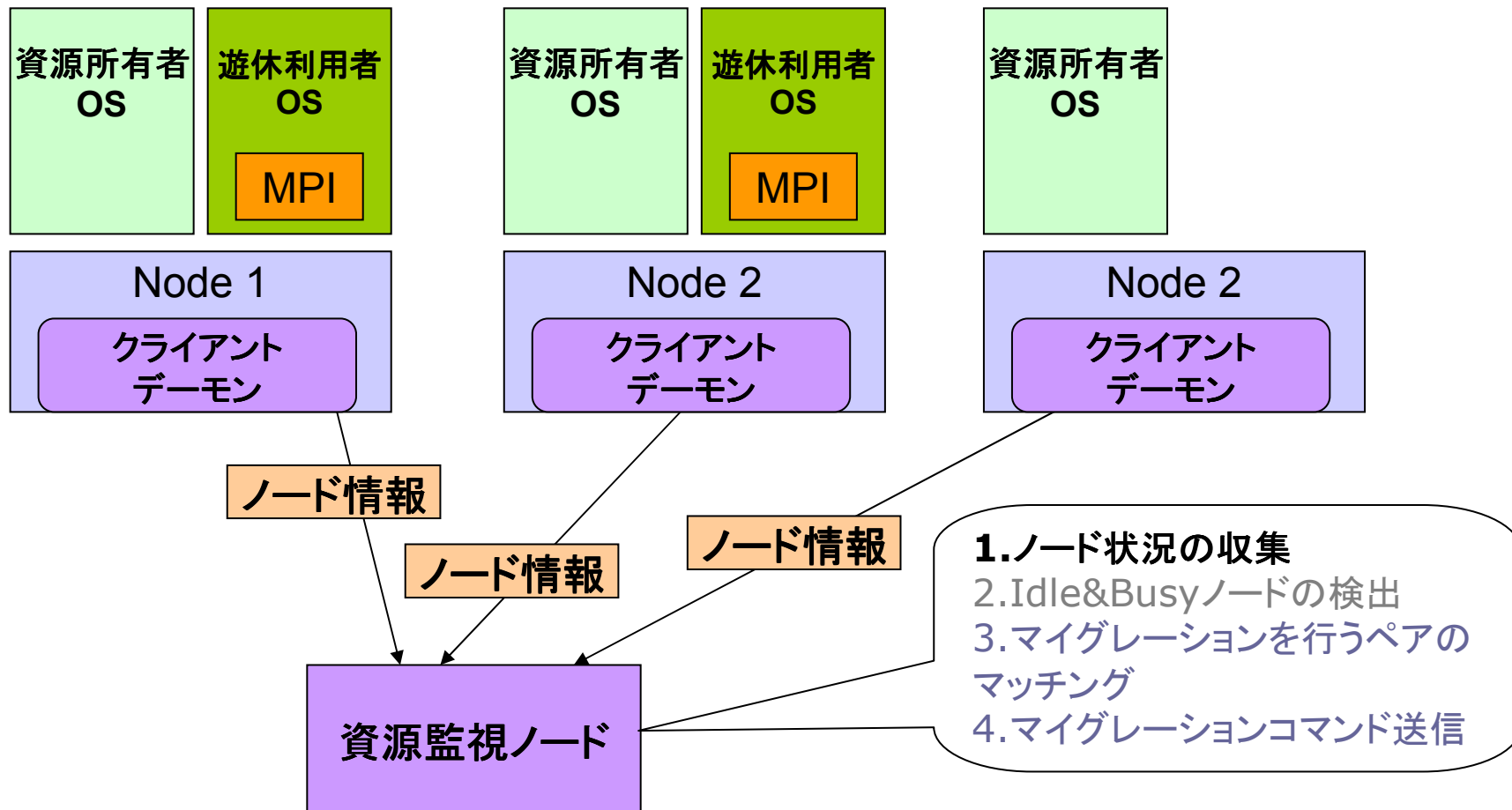
- MPIは仮想計算機のゲストOS上で実行
- 必要に応じてゲストOSごとマイグレーション
  - 仮想計算機のマイグレーション機能を使用



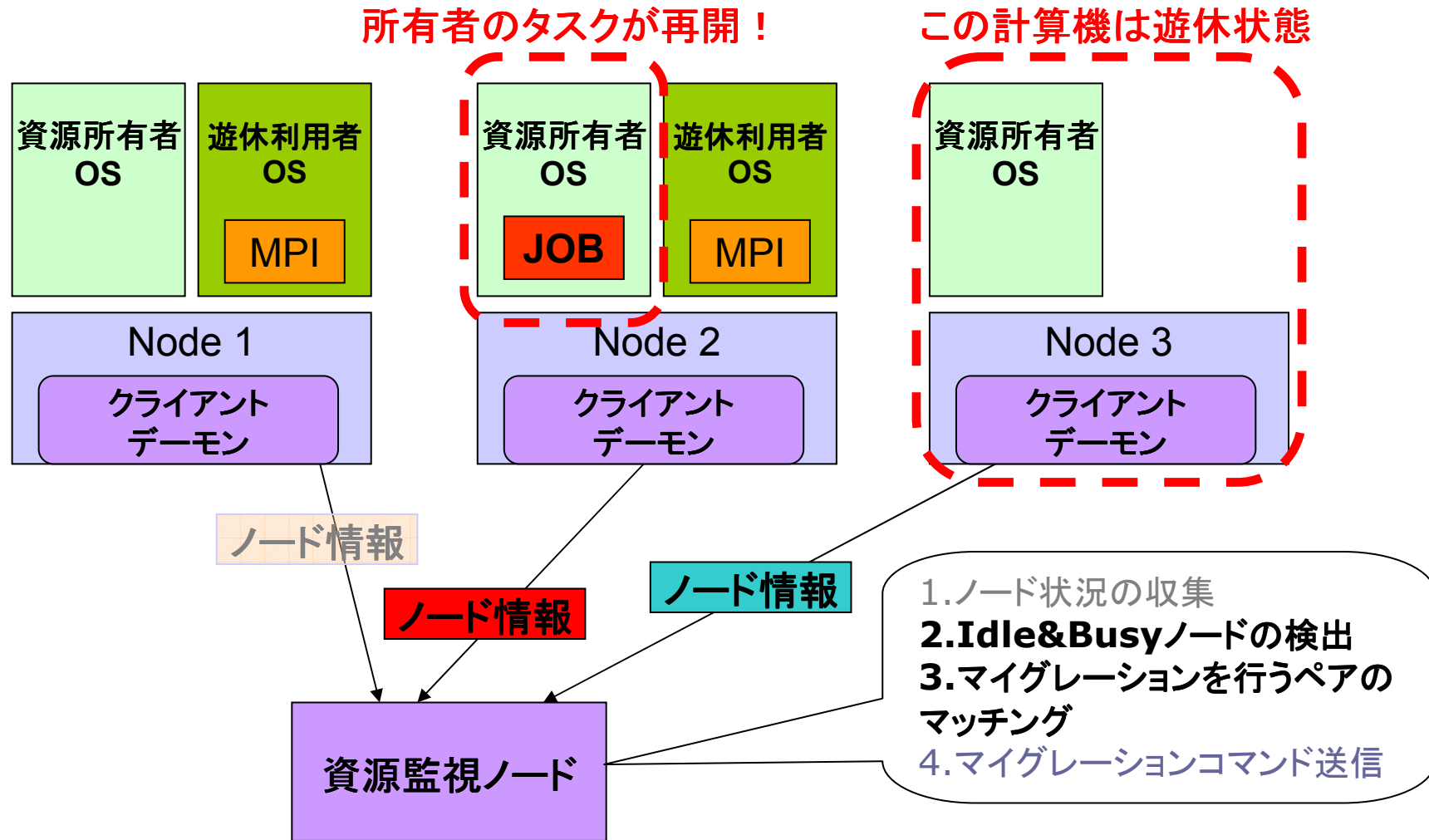
# プロセス・マイグレーションとの比較

- ネットワーク接続
  - OSマイグレーション
    - ネットワークに関する変更が無い
    - コネクションを維持したまま実行を継続
  - プロセス・マイグレーション
    - ネットワーク接続の張りなおしが必要
  - OSマイグレーションは既存のMPI実装で実現可能
- 転送するイメージサイズ
  - プロセス・マイグレーション
    - メモリのフットプリントサイズ
  - OSマイグレーション
    - ゲストOSのメモリサイズ
  - $\text{フットプリント} \leq \text{ゲストOSのメモリ}$
- その他
  - OSマイグレーションにはVM依存のオーバーヘッドが存在

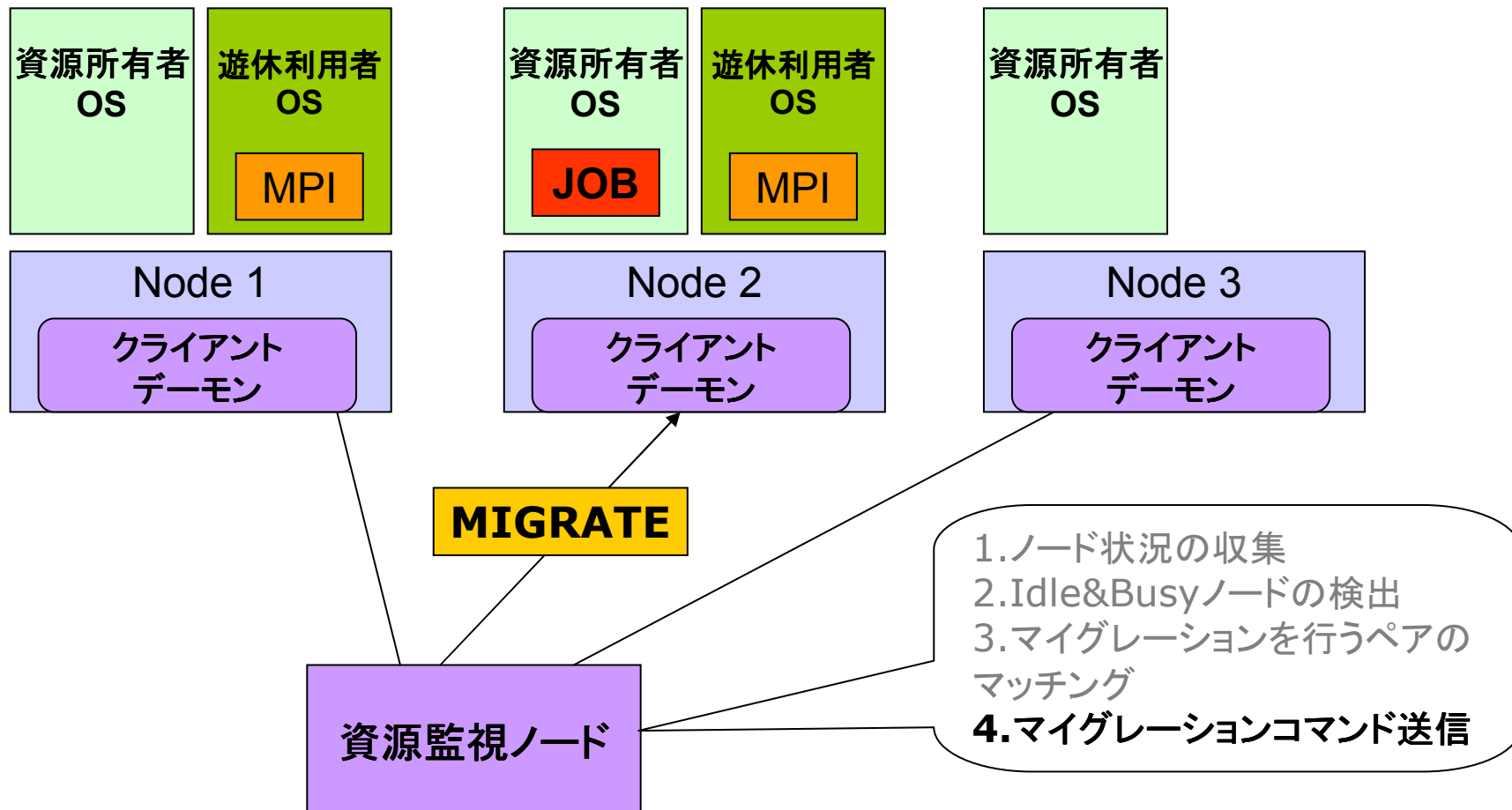
# 仮想計算機を用いた遊休計算機利用システム



# 仮想計算機を用いた遊休計算機利用システム

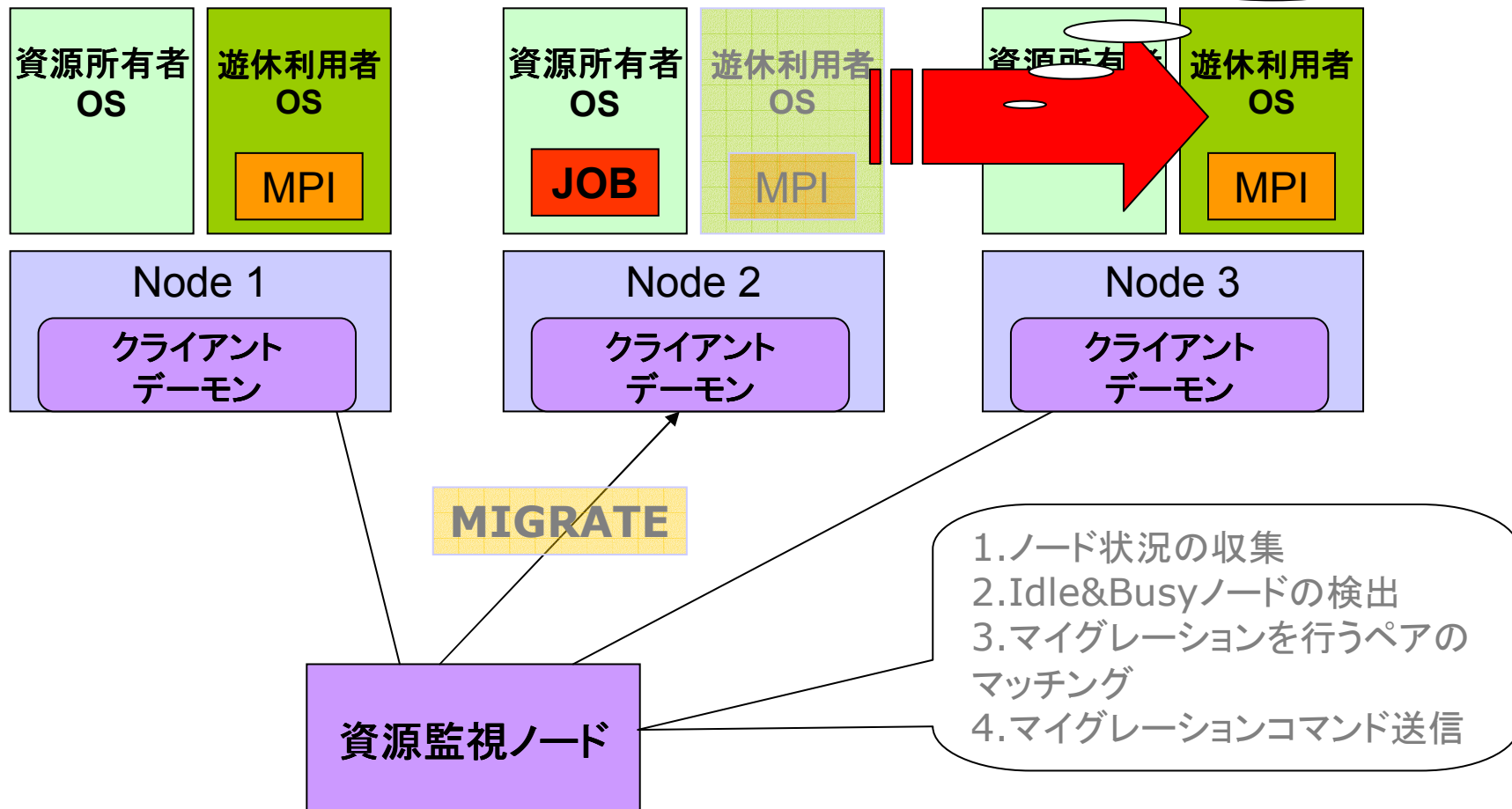


# 仮想計算機を用いた遊休計算機利用システム



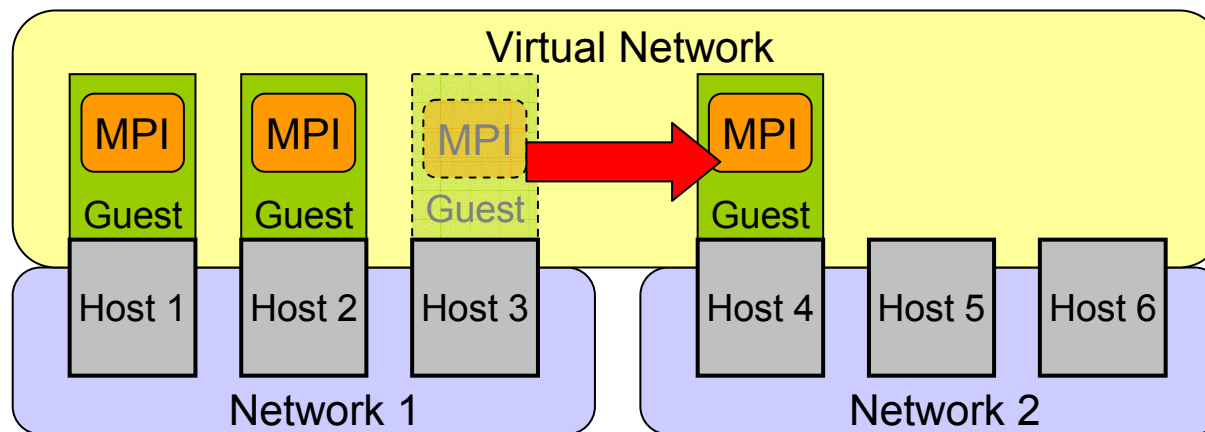
# 仮想計算機を用いた遊休計算機利用システム

マイグレーションの実行



# VPNを用いた複数サイト間でのMPI実行環境

- MPIの実行には固定されたIPアドレスが必要
  - 他のネットワーク上にマイグレーション不可
- VPNを用いてホストOS上に仮想ネットワークを構築
- そのネットワーク上でMPIを実行
  - どこへマイグレーションしてもホスト名(IPアドレス)は不変
  - ゲストOSは仮想ネットワークにブリッジ接続





# 目次

1. はじめに
2. 提案と設計
3. 実装
4. 評価
5. 考察
6. おわりに

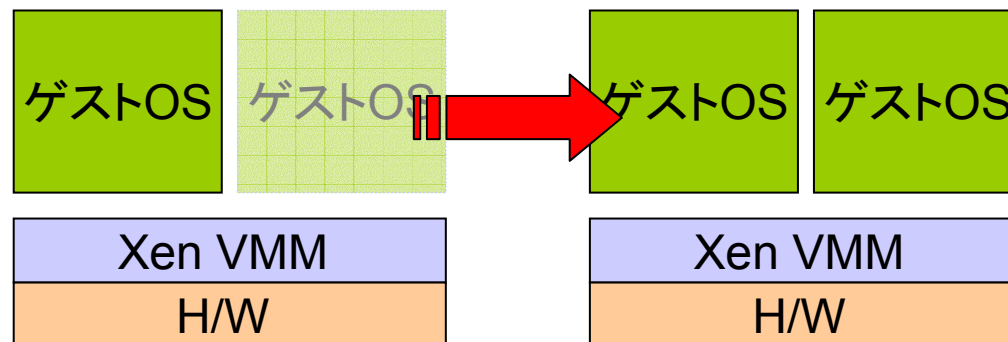


# プロトタイプ実装の概要

- 提案環境の実現に向けて、その要件について次のように実装
  - MPIのマイグレーション
    - 仮想計算機Xen[Pratt *et al.* '03]の機能により実現
  - 遊休計算機利用におけるジョブ実行システム
    - 資源モニタリング、マイグレーション実行決定をおこなう資源管理システムをJavaで実装
  - 複数サイトへのマイグレーション
    - ネットワークが異なるサイト
      - OpenVPNを用いたネットワークの仮想化により実現
  - 広域での実行への適応
    - 今後の課題

# XenとOSマイグレーション機能

- Xen:複数のOSを同時に動作可能なVMM
- 動作中のゲストOSを停止することなく、ネットワーク経由で他のXenホストへマイグレーション
  - ホストOSからコマンドによって実行
  - 転送する内容はメモリ内容、レジスタ等の内部状態
  - ディスクイメージは転送しない
    - ネットワークファイルシステム等に対応





# 目次

1. はじめに
2. 提案と設計
3. 実装
4. 評価
5. 考察
6. おわりに

# 評価環境



- 松岡研究室 PrestoIII クラスタ
  - CPU : Opteron242 \* 2
  - Memory : 2GB
  - Network : GigabitEther (Broadcom NetXtremeBCM5702X )
  - ノード数 : 256台

# 評価環境

- 仮想計算機環境

- Xen3.0.0

- Kernel 2.6.12

- Memory : 256MB

- rootイメージはNFS共有

- OpenVPN2.0

- MPI : mpich-1.2.7

- ベンチマーク:

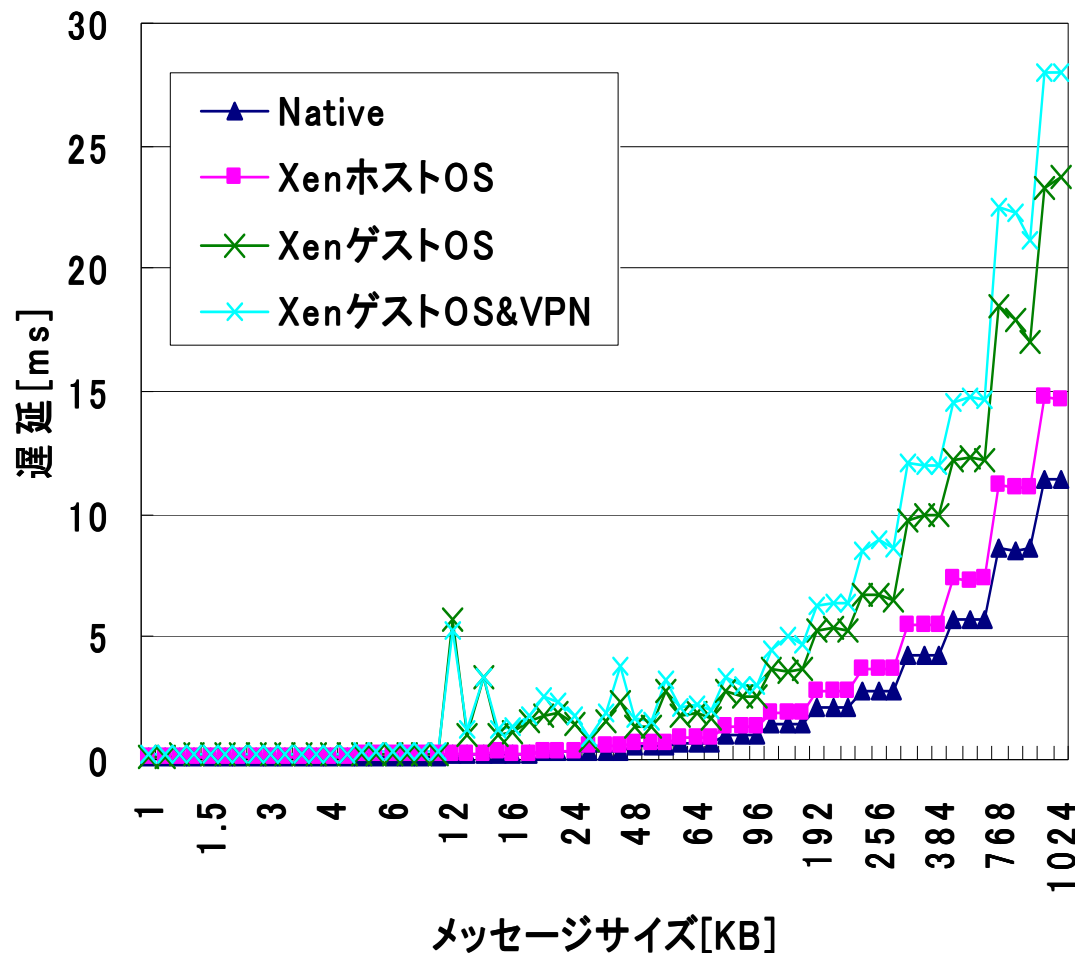
- Nas Parallel Benchmark3.1

# 評価項目



- Xenのネットワーク性能のマイクロベンチ
  - NetPIPEによるメッセージサイズと遅延の関係
- マイグレーションによるアプリケーション実行性能への影響
- MPIアプリケーション性能
  - NPB3.1およびqn24b(N-Queen)
- 実装システム上での遊休計算機利用によるMPI実行

# NetPIPE [Turner *et al.* '02 ]による NativeなカーネルとXenカーネルの遅延の比較



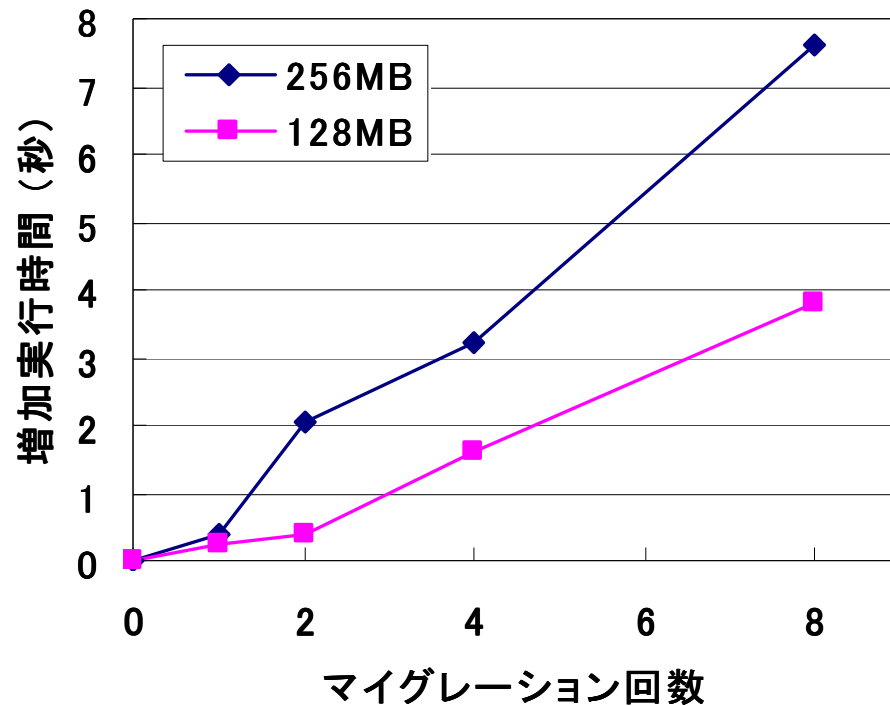
MPIによるメッセージピンポン  
での遅延測定

NativeとXenゲストOSでは約  
2倍の差  
Xenのネットワーク構造に起因  
する

# メモリサイズとマイグレーションコストの関係

NPB2.4 EP CLASS=B 8CPUで実行

実行時にマイグレーションを行い、実行時間の増加を計測



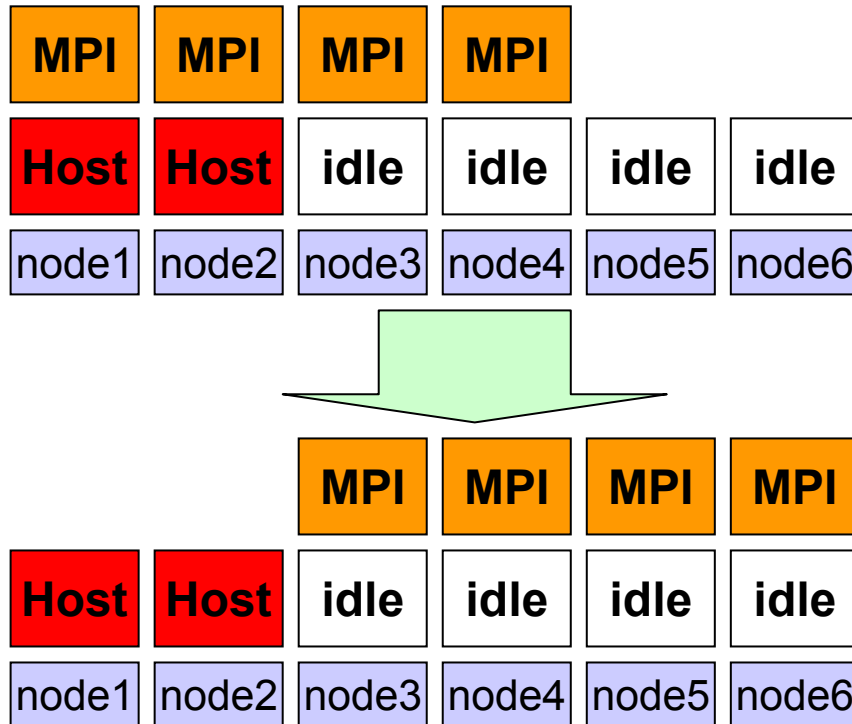
8回行った場合の1回あたりのコスト

- 256MB時 約1秒
- 128MB時 約0.5秒

いずれも長時間の実行を前提とした場合、軽微なオーバーヘッド



# 遊休計算機利用による計算効率向上



node1~4でMPIを実行  
 途中、node1,2で所有者のジョブ再開  
 →監視システムが検知

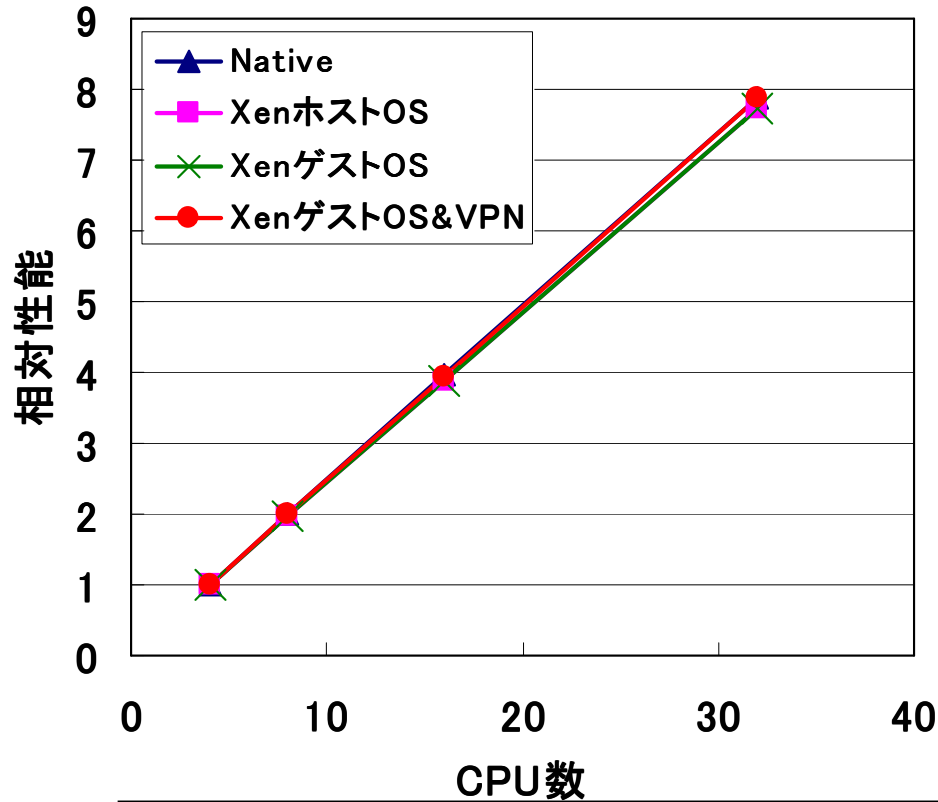
node1,2のMPIをnode5,6へ  
 マイグレーション

(単位:秒)

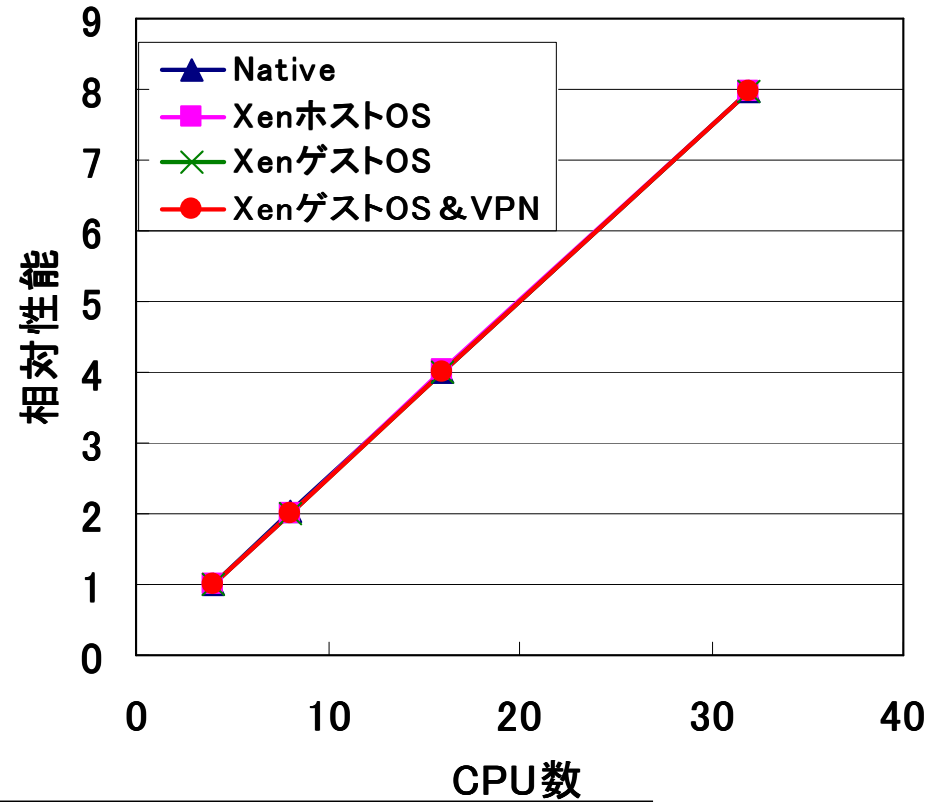
ジョブ名	Xen + プロトタイプ実装	Native + マイグレーション無し
Host(LU,A,4)	171.34	230.88
MPI (LU,A,2)	339.89	380.38

# NPB3.1&qn24bでのスケーラビリティの測定 Nativeカーネル4CPU時に対する相対性能

EP CLASS=B



qn24b

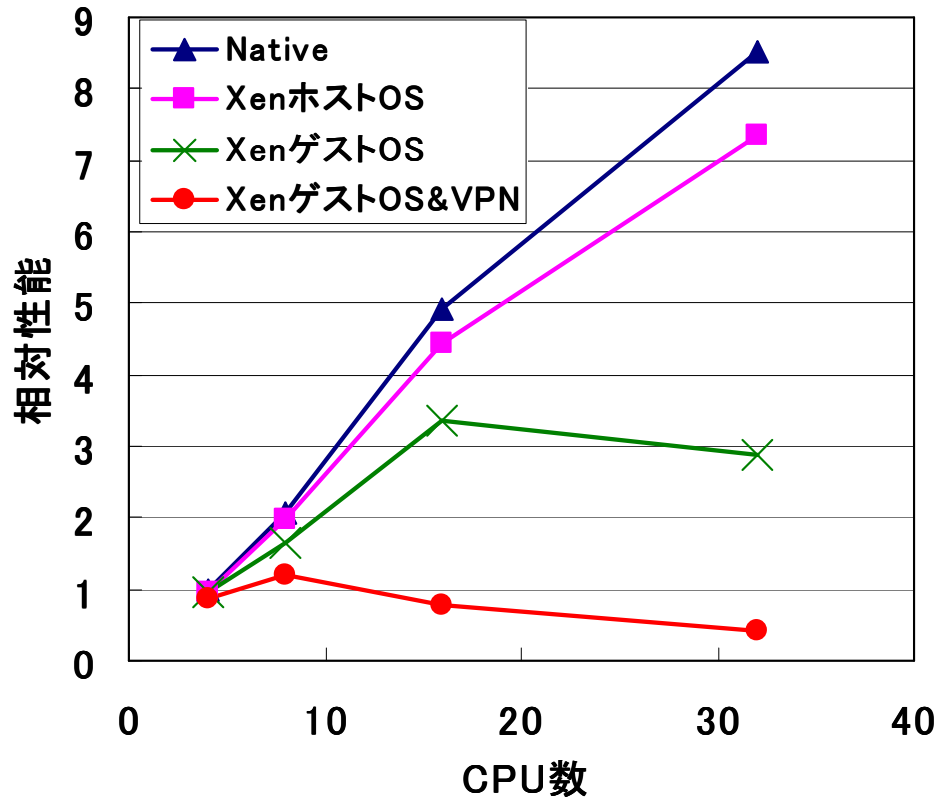


各実行環境とも高い台数効果が得られている

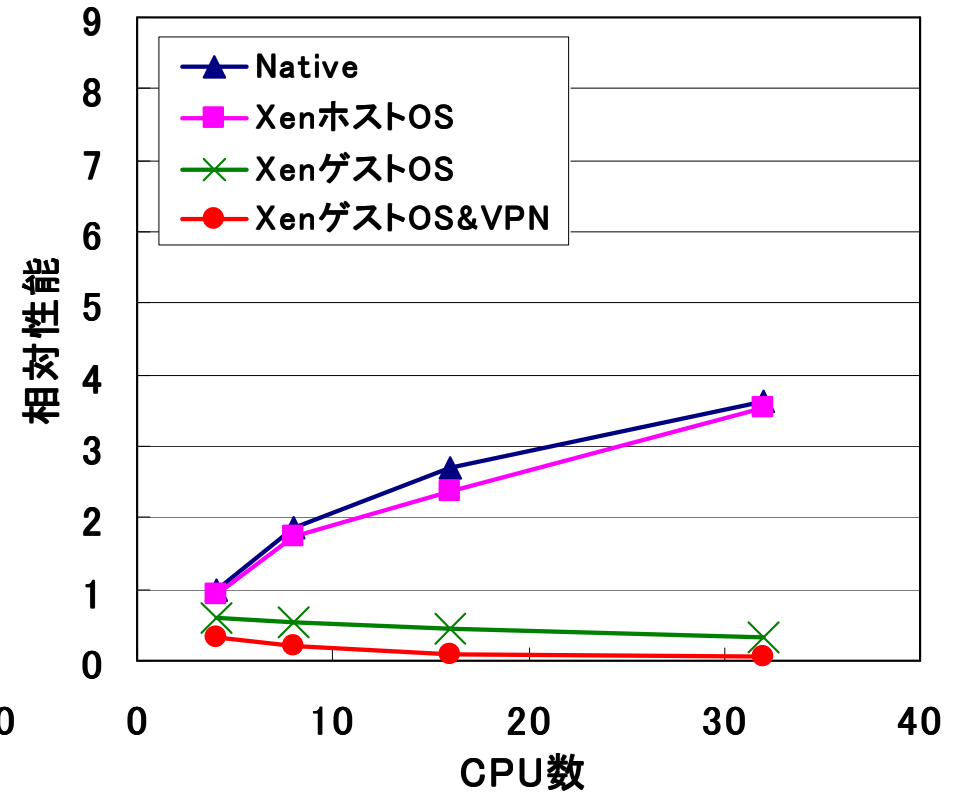
# NPB3.1でのスケーラビリティの測定

## Nativeカーネル4CPU時に対する相対性能

LU CLASS=B



CG CLASS=B



**ゲストOS上の実行で台数効果が得られていない！**



# 目次

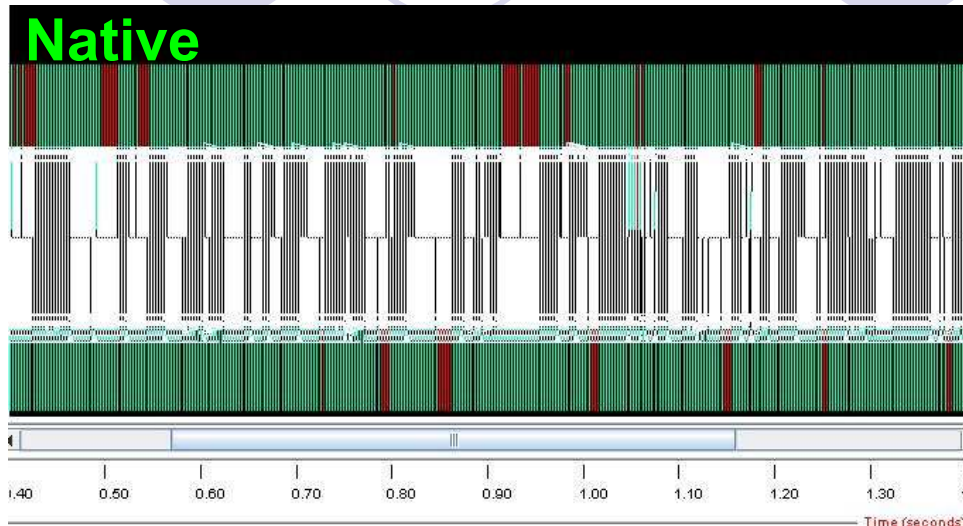
1. はじめに
2. 提案と設計
3. 実装
4. 評価
5. 考察
6. おわりに

# 考察 – NPBの性能低下の原因

- 評価の結果、XenのゲストOS上でCG,LUの性能が低下
- 推測される原因としてネットワークの性能低下
  - CG,LUともに頻繁な通信を伴い、EP,qn24bは通信が少ない
  - Xenのネットワーク遅延はNative環境の約2倍

→これだけではCGの性能低下を説明できない

# MPI実行関数 & 通信のログ

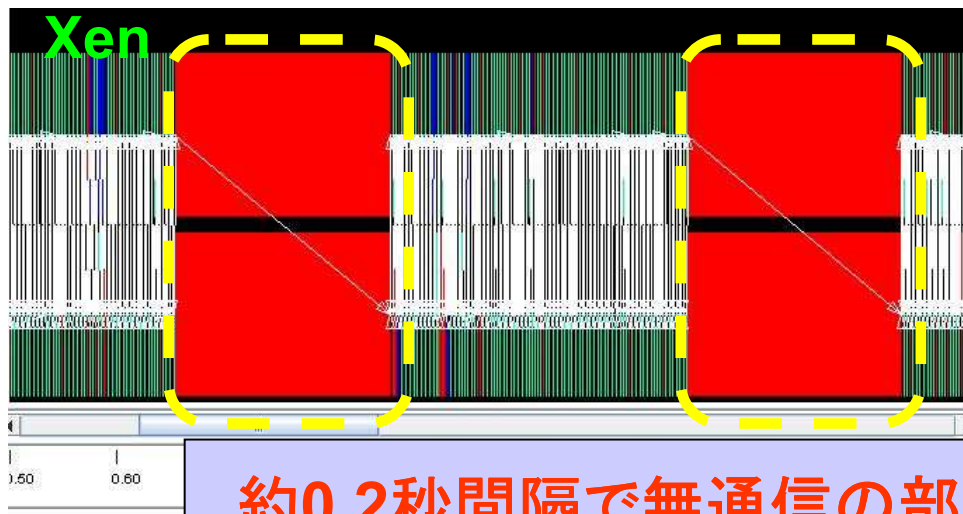


NPB3.1 CG クラスWを  
2ノードで実行

mpichで実行関数ログを取得

本来通信が断続的に行われ  
るはず

→Xenでの実行では無通信  
の部分が存在



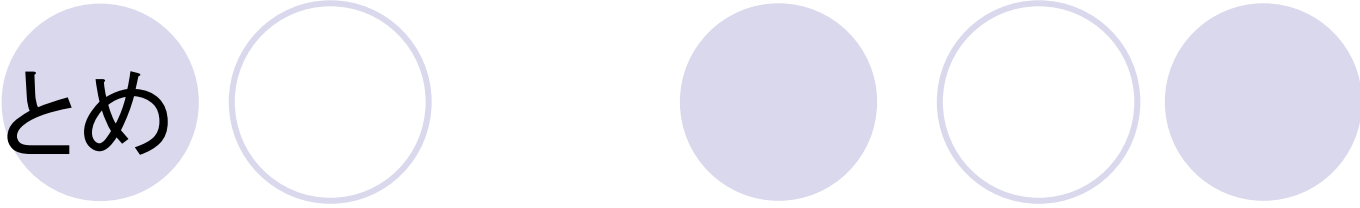
約0.2秒間隔で無通信の部分が存在



# 目次

1. はじめに
2. 提案と設計
3. 実装
4. 評価
5. 考察
6. おわりに

# まとめ



- 仮想計算機Xenを用いたMPIのプロセスマイグレーションの有用性の実証
- VPNとの組み合わせにより、既存のMPI実装を用いて広域へのマイグレーションの実現
- 実現した低コストなマイグレーションを用いた遊休計算機利用のプロトタイプを構築
- Xen上でのMPI性能の評価と考察



# 今後の課題

- 性能低下原因の調査と対応
  - ログの解析を進め、無通信部分の原因を解明
- グリッドへの対応強化
  - WAN、LANの違い、ヘテロな環境を考慮したマイグレーションポリシー
  - 耐故障性の実現
  - プロトタイプ実装ではゲストOSのイメージ共有にNFSを使用
    - グリッド対応のファイルシステムとしてGfarm[Tatebe et al, '02]等の使用
- 実環境下での評価
  - 資源提供者から視点
  - 大規模なグリッド環境



ご静聴ありがとうございました

- 本研究の一部は科学技術振興事業団・戦略的創造研究「低電力化とモデリング技術によるメガスケールコンピューティング」による

## スイッチングハブへの対応

- マイグレーションを行うとアドレステーブルのポート情報と異なるポートにゲストが移動
- 通信がwait状態にあると、テーブルが更新されない
  - マイグレーションを行ったノードへのパケットの送信が不可能

→ プロトタイプ実装では、pingによるICMPパケット送出によりポートテーブルを強制的に更新

## XenのOSマイグレーション時のダウンタイム短縮

- メモリイメージの送信はゲストOSの実行と並行に行われる
  - ホストAからホストBへマイグレーションする場合
    1. AでゲストOSを動作させたまま、メモリ内容をBへ転送
    2. 転送中に変更されたメモリ内容を再び転送
    3. 2を一定回数繰り返す
    4. 差分が一定量を下回ったらAでゲストOSをサスペンド
    5. 未転送の部分を全てBへ転送
    6. BでゲストOSをレジューム

