

電気学会全国大会シンポジウム

# グリッドコンピューティングを用いた 並列分枝限定法

大角 知孝\*

東京工業大学

合田 憲人

東京工業大学 / 科学技術振興機構さきがけ

# 発表内容

## ■ グリッドコンピューティングを用いた分枝限定法

### ■ 分枝限定法と計算モデル

- 分枝限定法
- マスタ・ワーカ方式による並列化
- 階層的マスタ・ワーカ方式による並列化

### ■ グリッド上での実証実験

- 性能評価
- グリッド実験環境の構築

### ■ アプリケーションスケジューリング技術の開発

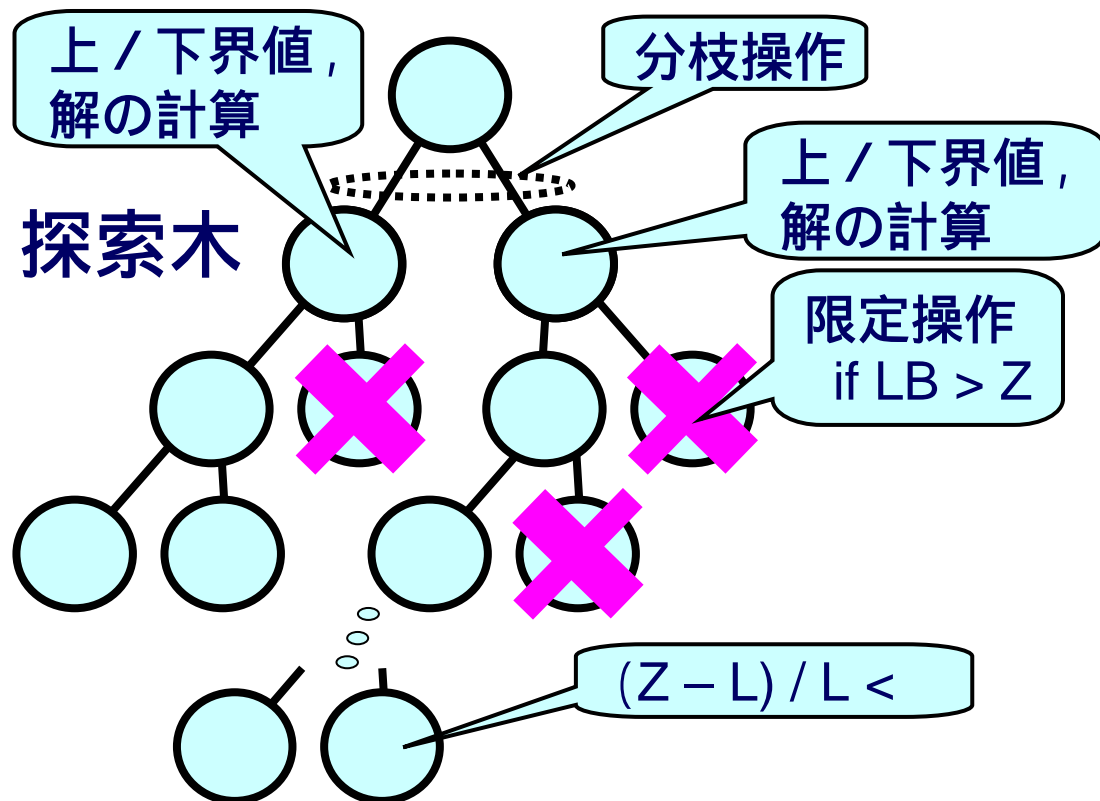
- PCクラスタ間の負荷分散

# 分枝限定法

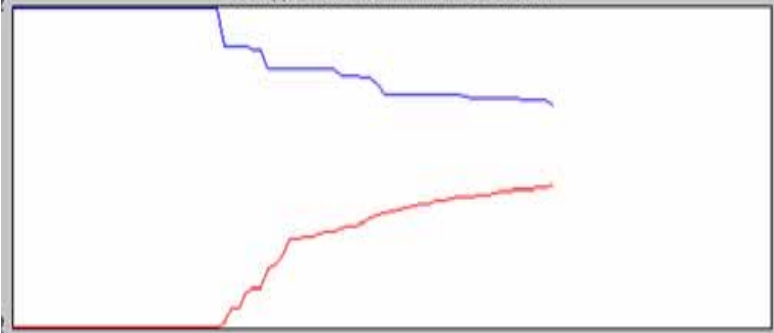
## ■ 最適解の探索

- 問題を複数の小規模な問題(子問題)に再帰的に分割して, 各子問題について解の計算を行う.
- 最適解の存在しない子問題(例: 下界 > 暫定値)は探索木から削除.

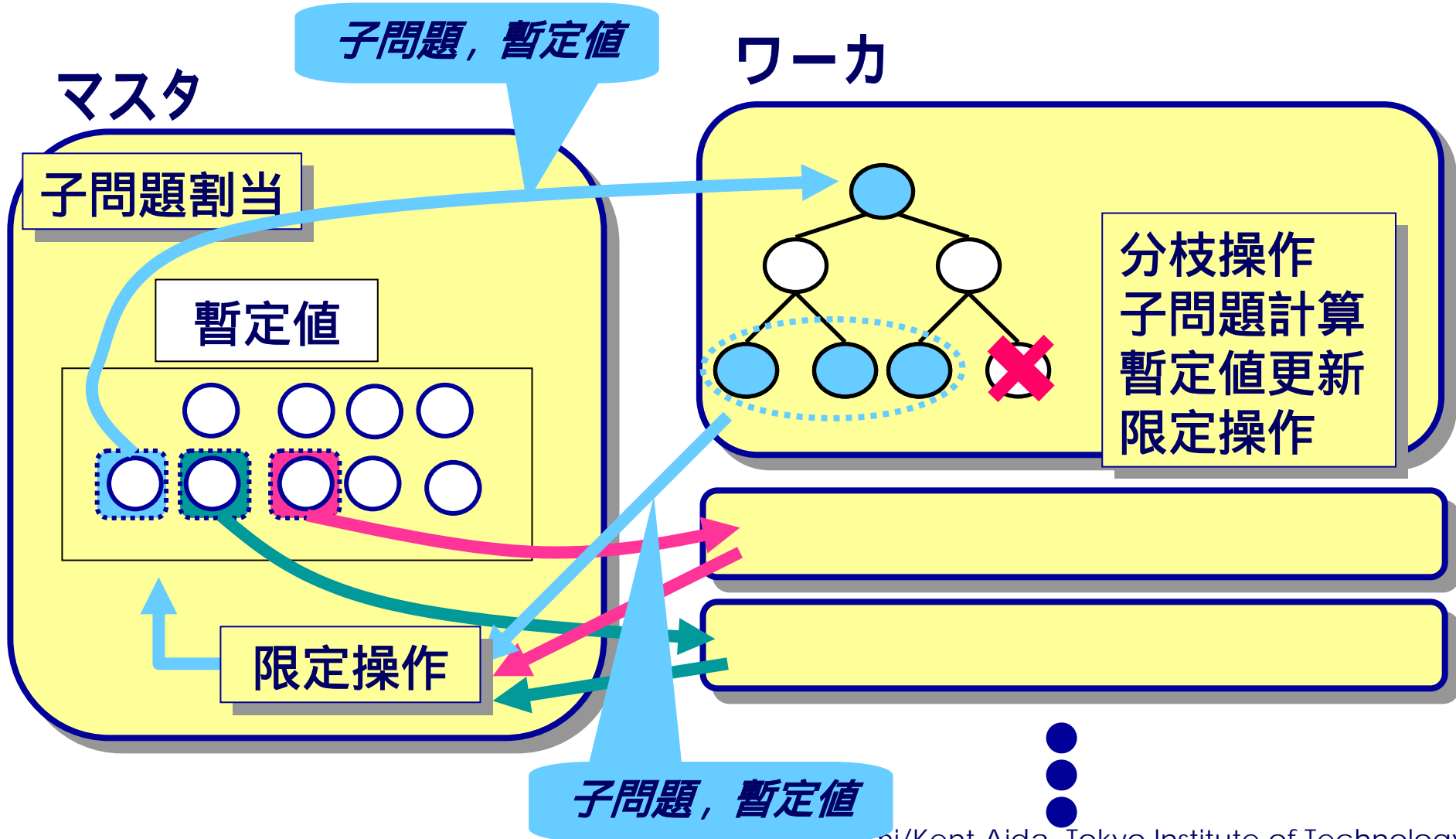
➤ 暫定値 = 子問題間の最小上界値 (最小化問題の場合)



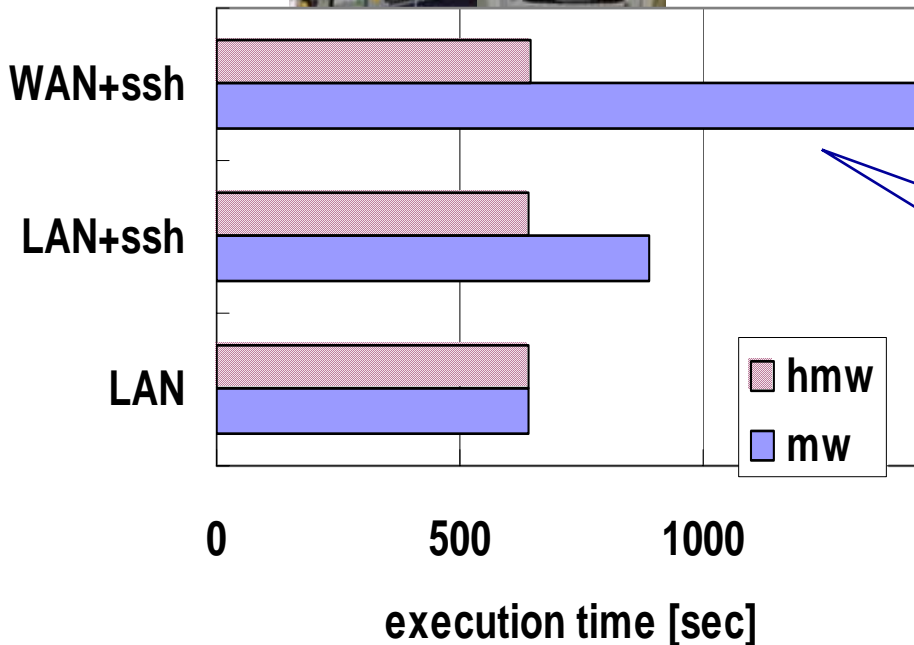
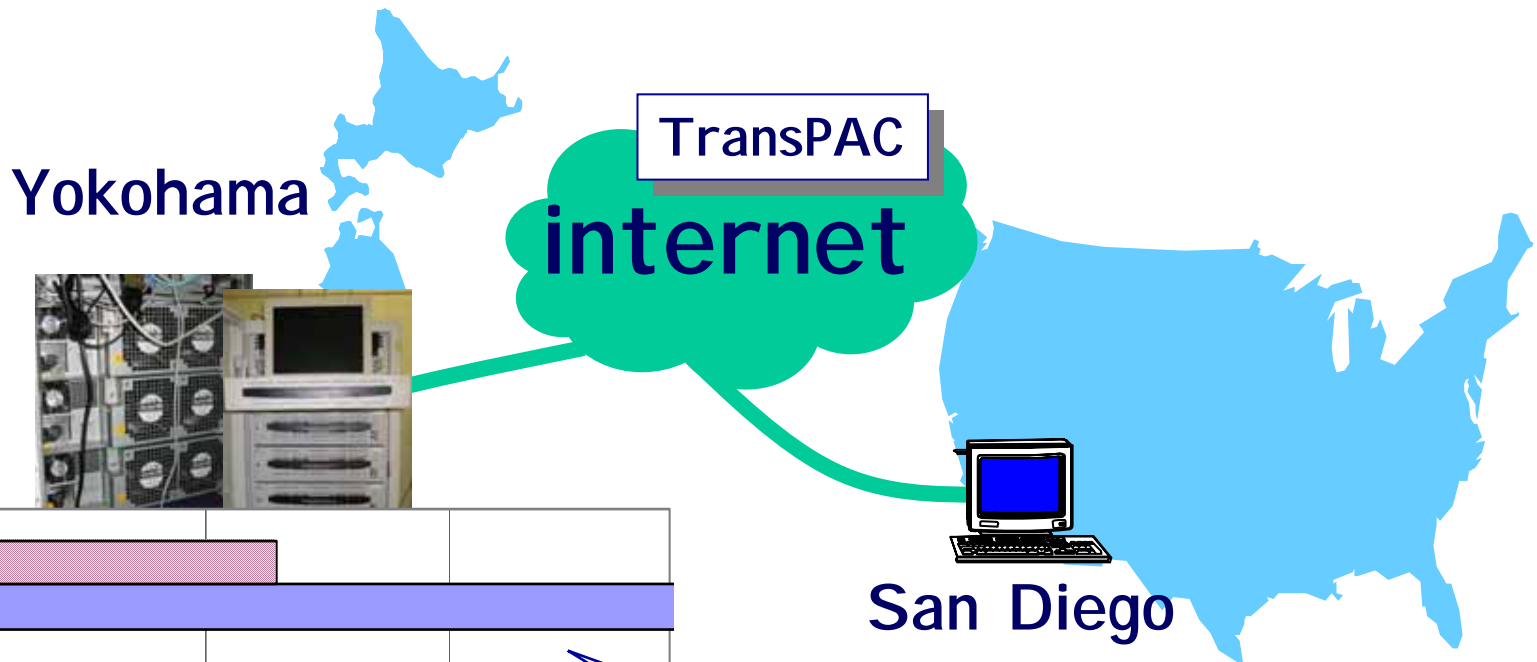
best upper bound & minimum lower bound



# マスタ・ワーカ方式による 分枝限定法の並列化



# グリッド上でのマスタ・ワーカ方式の問題



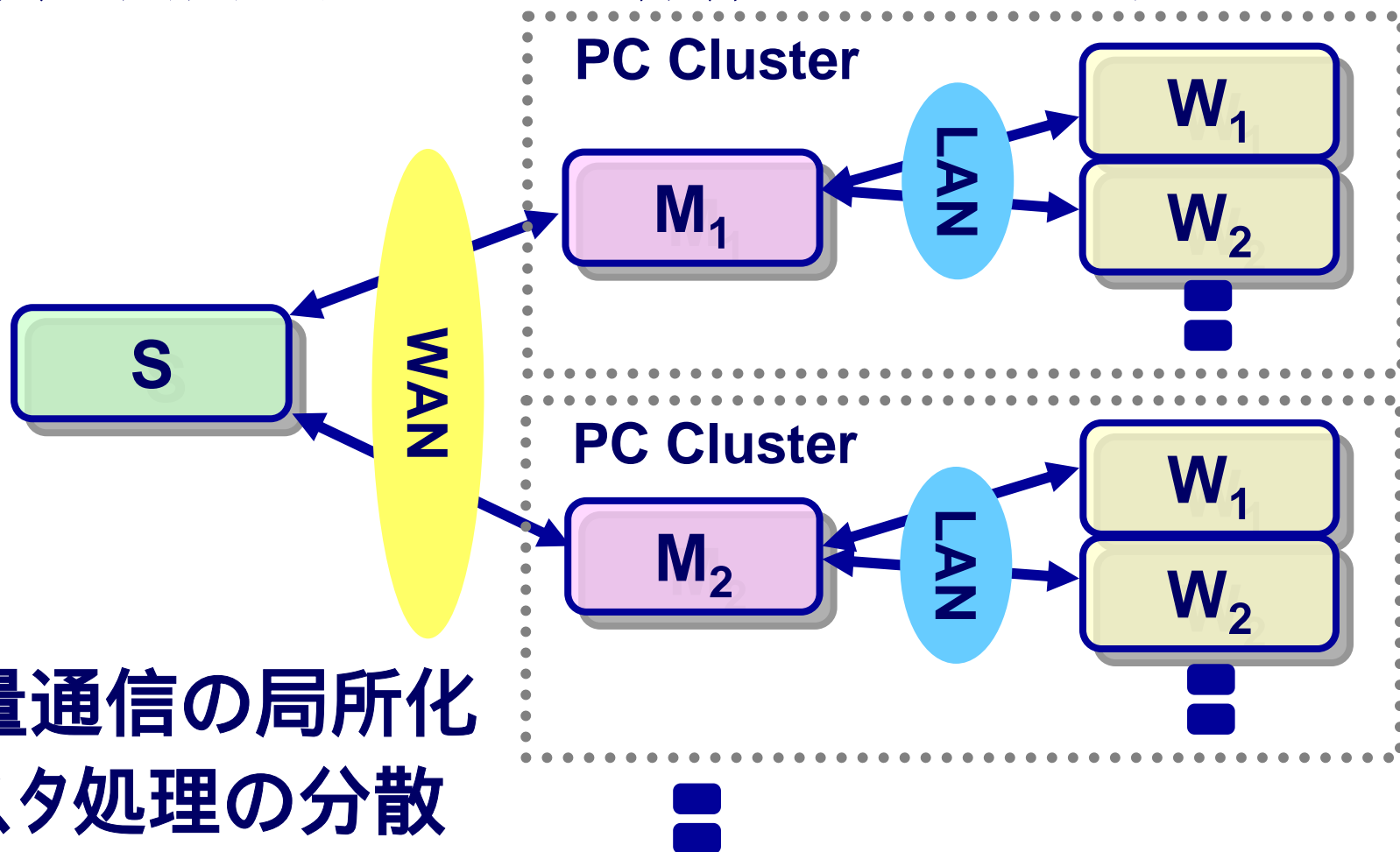
通信遅延, SSHトンネリング  
処理によるオーバーヘッド

# 階層的マスタ・ワーカ方式

スーパーバイザ

マスタ

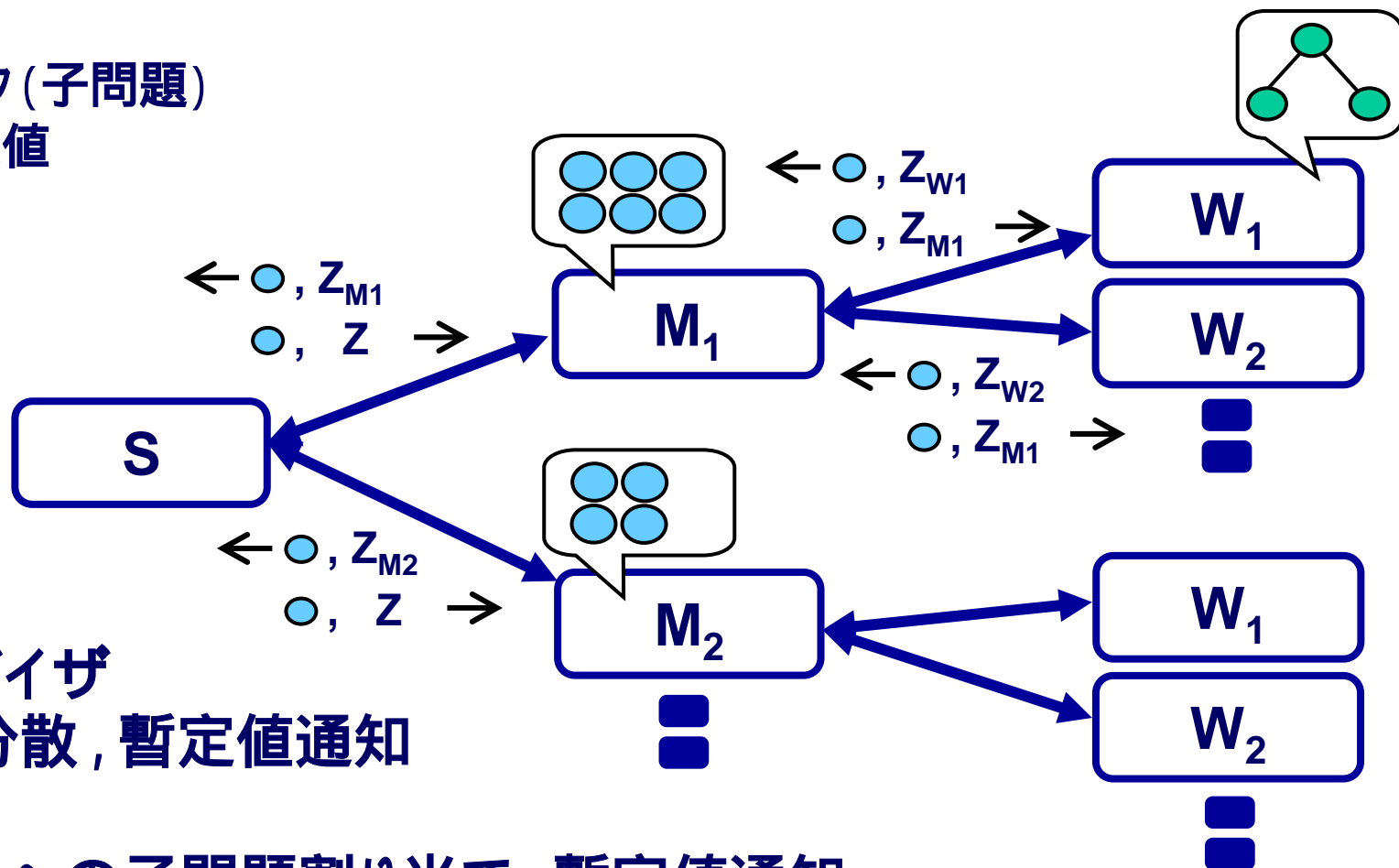
ワーカ



- 大量通信の局所化
- マスタ処理の分散

# 階層的マスタ・ワーカ方式を用いた 並列分枝限定法

● : タスク (子問題)  
Z : 暫定値



□ スーパーバイザ

負荷分散, 暫定値通知

□ マスタ

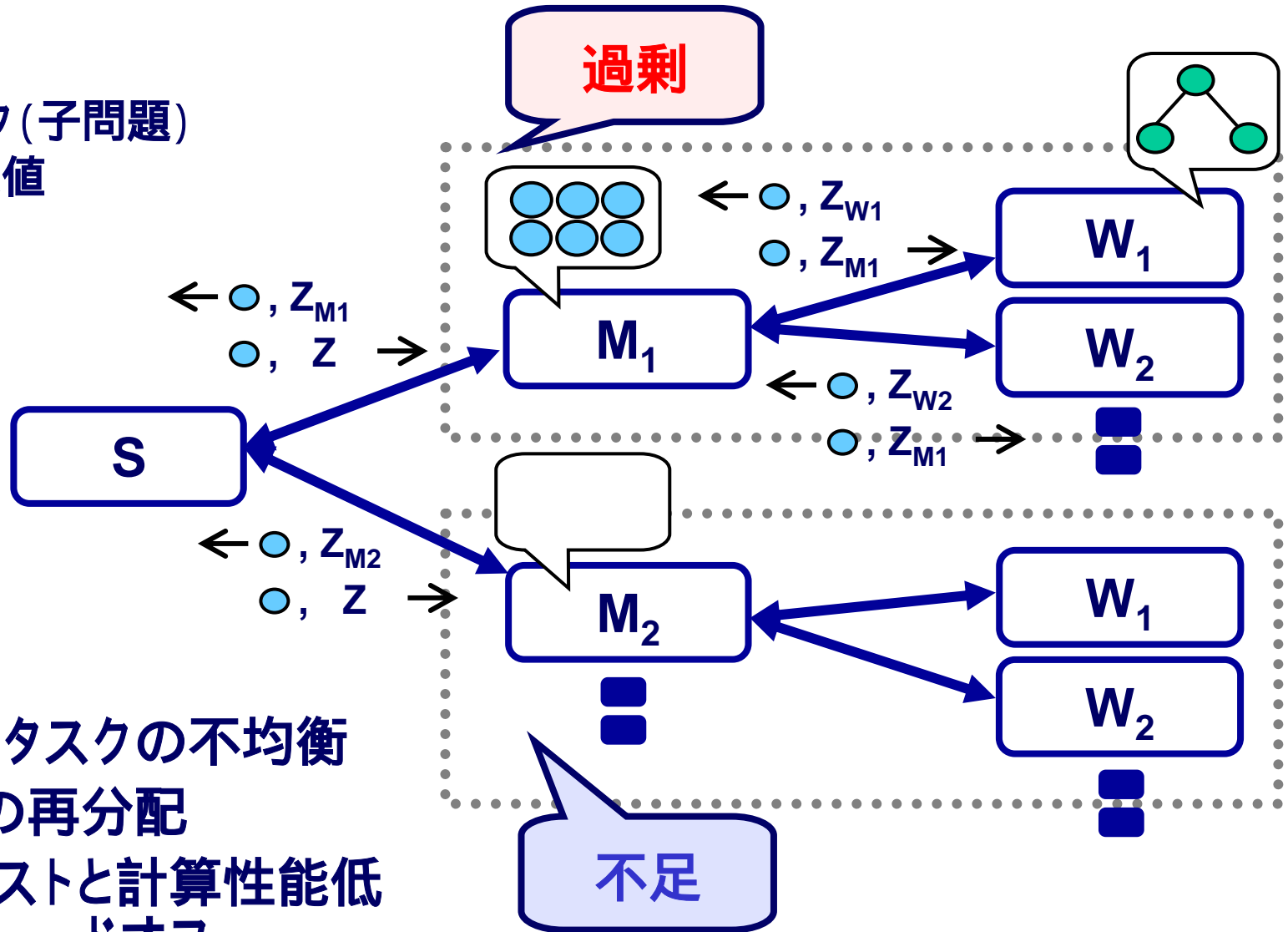
ワーカへの子問題割り当て, 暫定値通知

□ ワーカ

子問題計算

# タスクの負荷分散の必要性

● : タスク (子問題)  
Z : 暫定値



- 未処理タスクの不均衡
- タスクの再分配
- 通信コストと計算性能低下のトレードオフ

# GridRPC

## ■ GridRPC

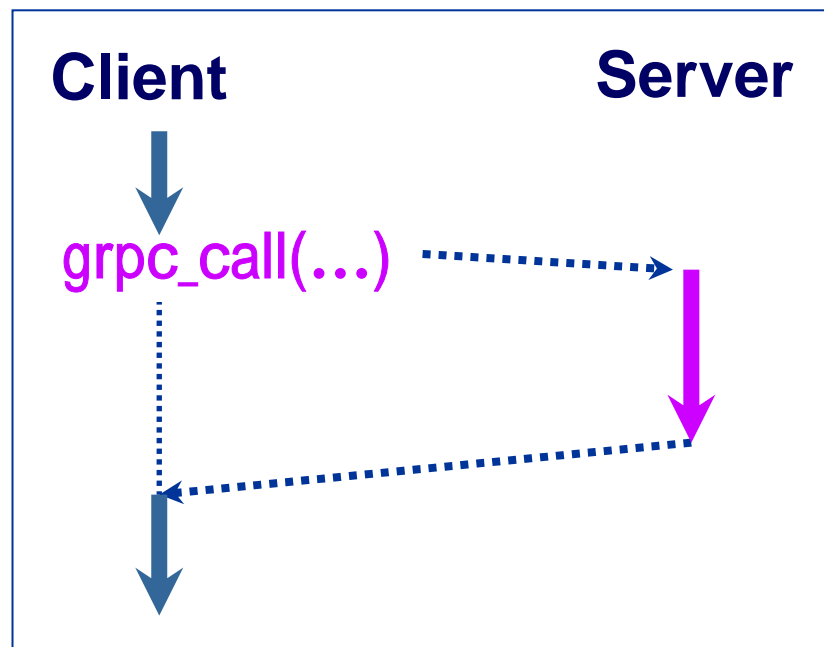
- グリッド上のクライアントサーバ型プログラミングツール
- GGFにおける標準化

## ■ Ninf-G

- GridRPCのリファレンスインプリメンテーション
- Globus Toolkit上に実装
  - GSIによるsecure RPC

## ■ Ninf

- Ninf-Gの前身
- 高速RPC

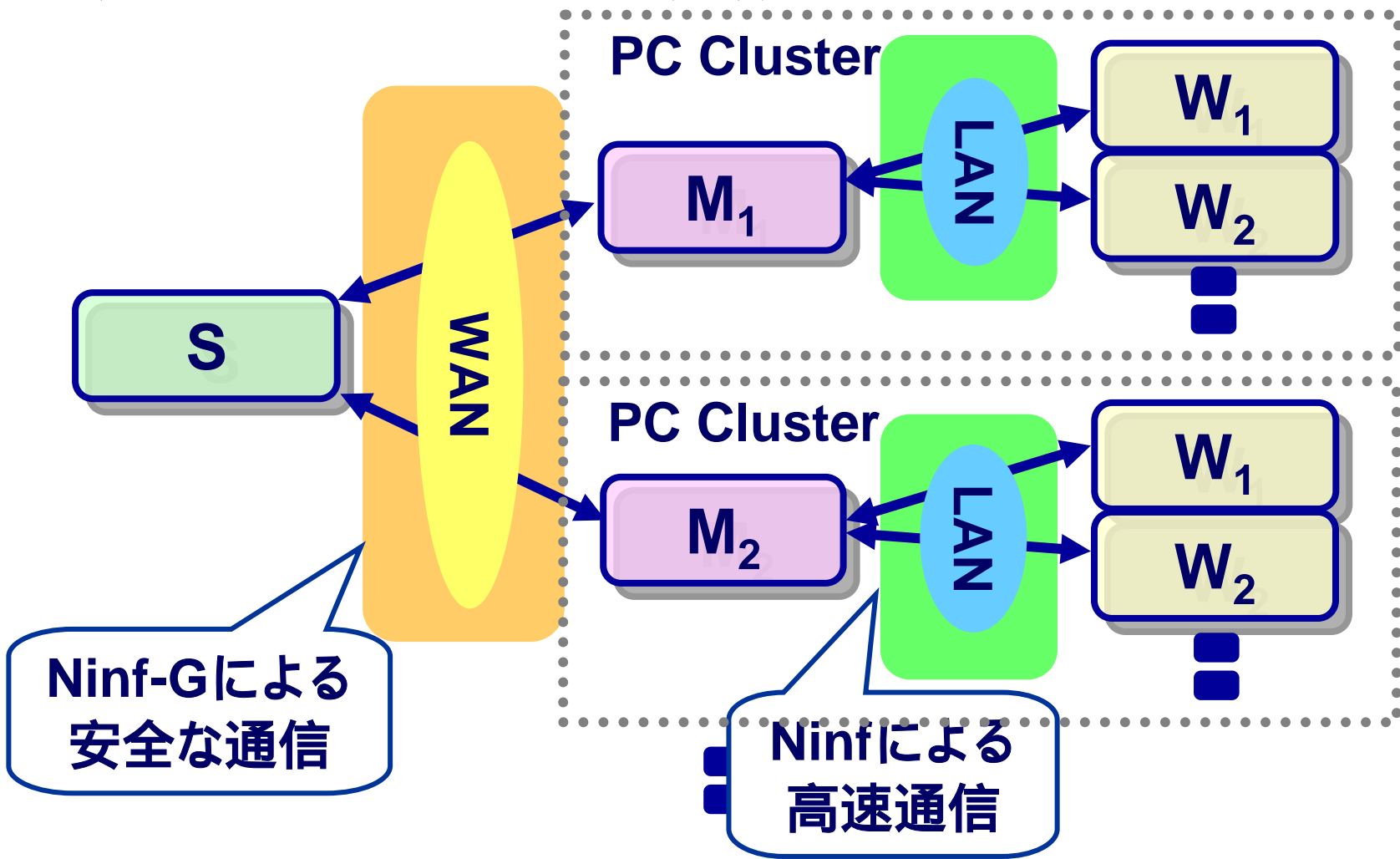


# GridRPCによる実装(続)

スーパーバイザ

マスタ

ワーカ



# グリッド実験環境

Globus Toolkit 2.4.\*  
Ninf-G 2.2.0

## Sdpa

dual Athlon 2GHz  
東京電機大 (埼玉)



## Mp

dual Athlon 2.0GHz  
徳島大 (徳島)



RTT=12ms

RTT=28ms

WIDE

SINET

client/  
supervisor



LAN

RTT=6ms

Tsukuba  
WAN



RTT=0.03ms



## PrestoIII

dual Athlon 1.6GHz  
東工大 (東京)

## Blade

dual PIII 1.4GHz  
東工大 (横浜)

# BMI固有値問題

## ■ 定義

双線形行列関数  $F(x,y)$  の最大固有値を最小化するベクトル変数  $x, y$  を求める問題.

$$F(x,y) := F_{00} + \sum_{i=1}^{n_x} x_i F_{i0} + \sum_{j=1}^{n_y} y_j F_{0j} + \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} x_i y_j F_{ij}$$

$$F: R^{n_x} \times R^{n_y} \rightarrow R^{m \times m}$$

$$F_{ij} = F_{ij}^T \in R^{m \times m} (i=0, \dots, n_x, j=0, \dots, n_y)$$

## ■ 応用

### ■ 制御工学

➤ ヘリコプター制御, ロボットアーム制御

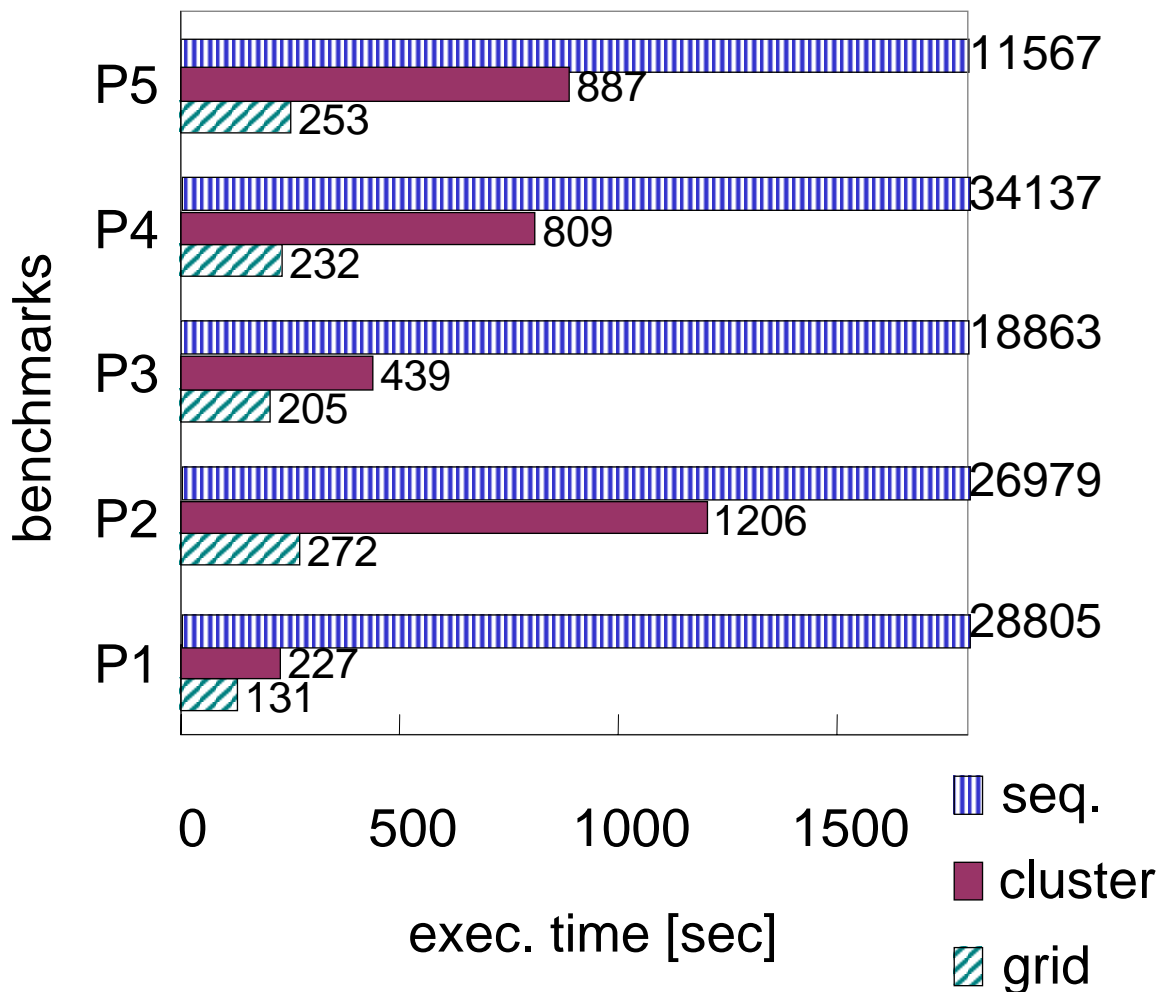


### ■ OR

➤ 大規模問題求解への挑戦

# グリッド上でのアプリケーション実行時間

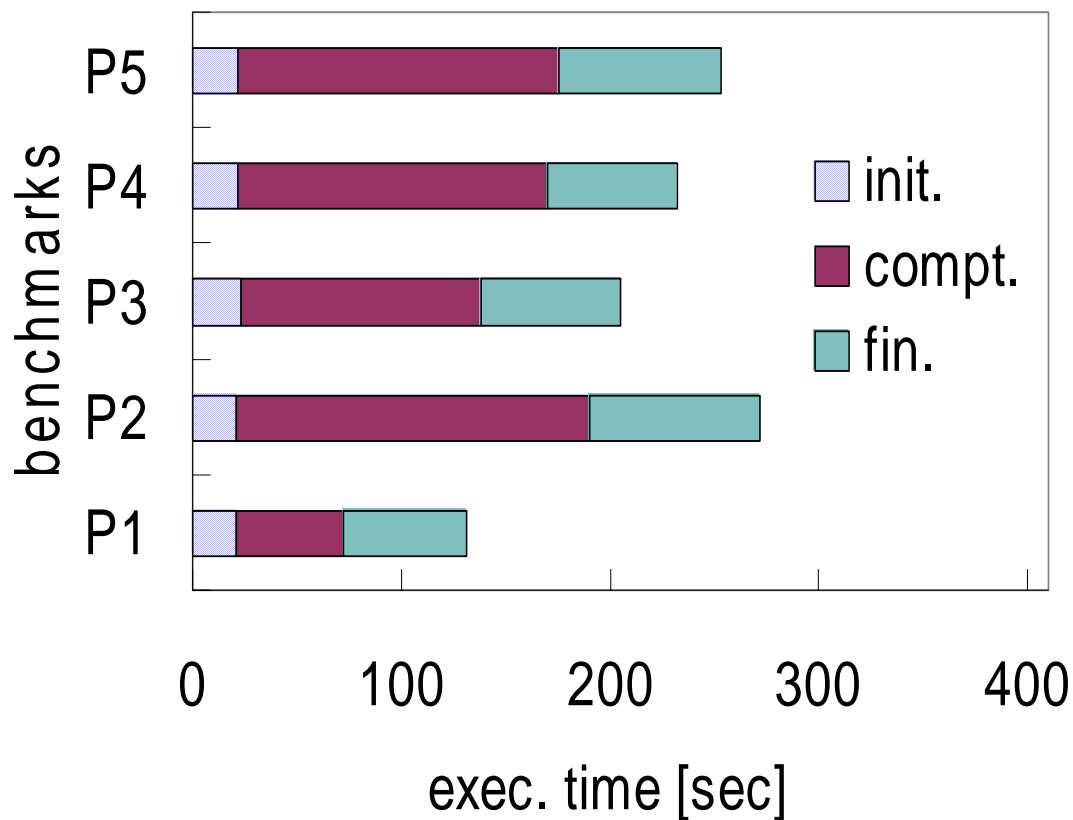
P1-P5:  $n_x=6, n_y=6, m=24$ ,  
タスク実行時間 = 0.5sec程度



seq: Blade (1CPU)  
cluster: Blade (73CPUs)  
grid: 4sites (412CPUs)

- 例:P2  
逐次計算: 9.5h  
グリッド計算: 4.5min
- 細粒度アプリに対する階層的マスタ・ワーカ方式の有効性を確認.

# グリッド上での実行時間内訳



- 終了処理オーバーヘッドによる性能低下大
- 解は終了処理以前にユーザに提示される。

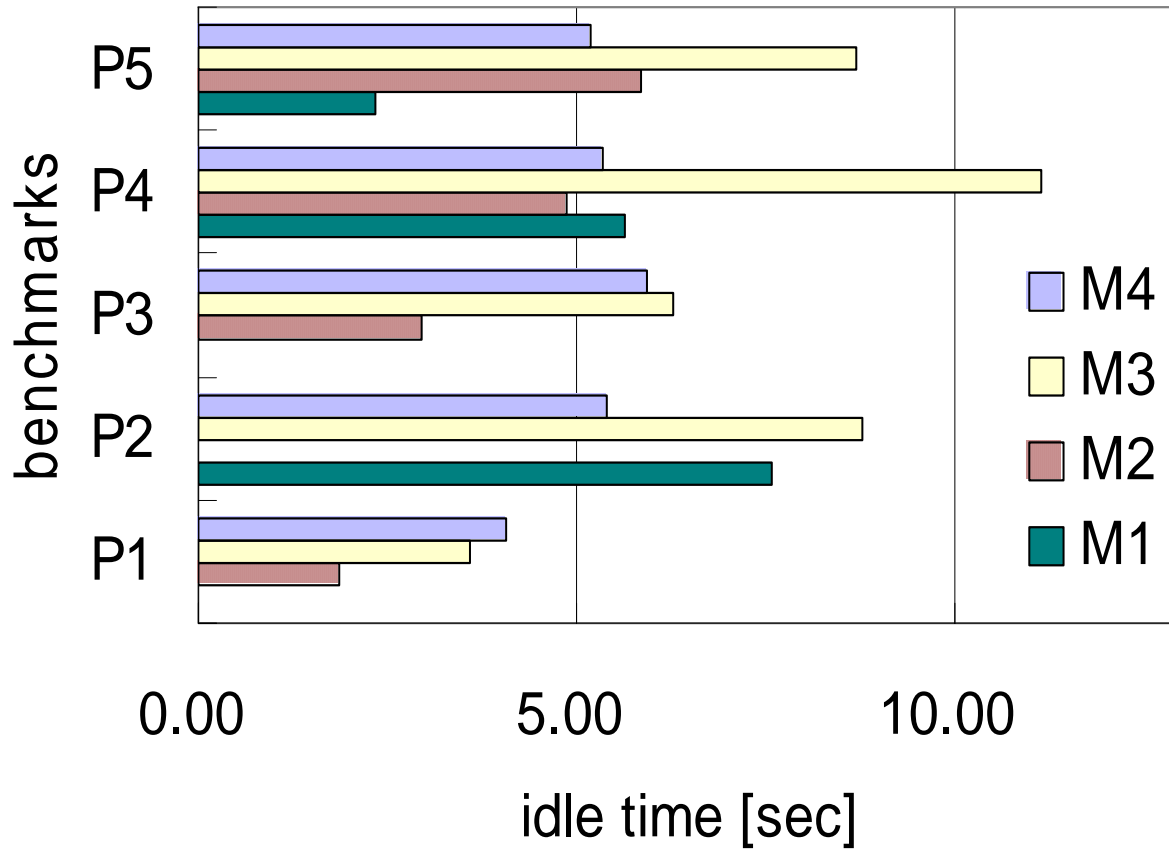
# PCクラスタ間負荷分散

## ■ supervisor

- master (= PCクラスタ) 上の負荷 (=未処理子問題数) を問い合わせ
- アイドル状態のマスタを発見する度に, タスクを再分散.
  - タスク実行履歴の保存
  - PCクラスタの性能に応じて分配タスク数を決定.

# PCクラスタ上でのアイドル時間

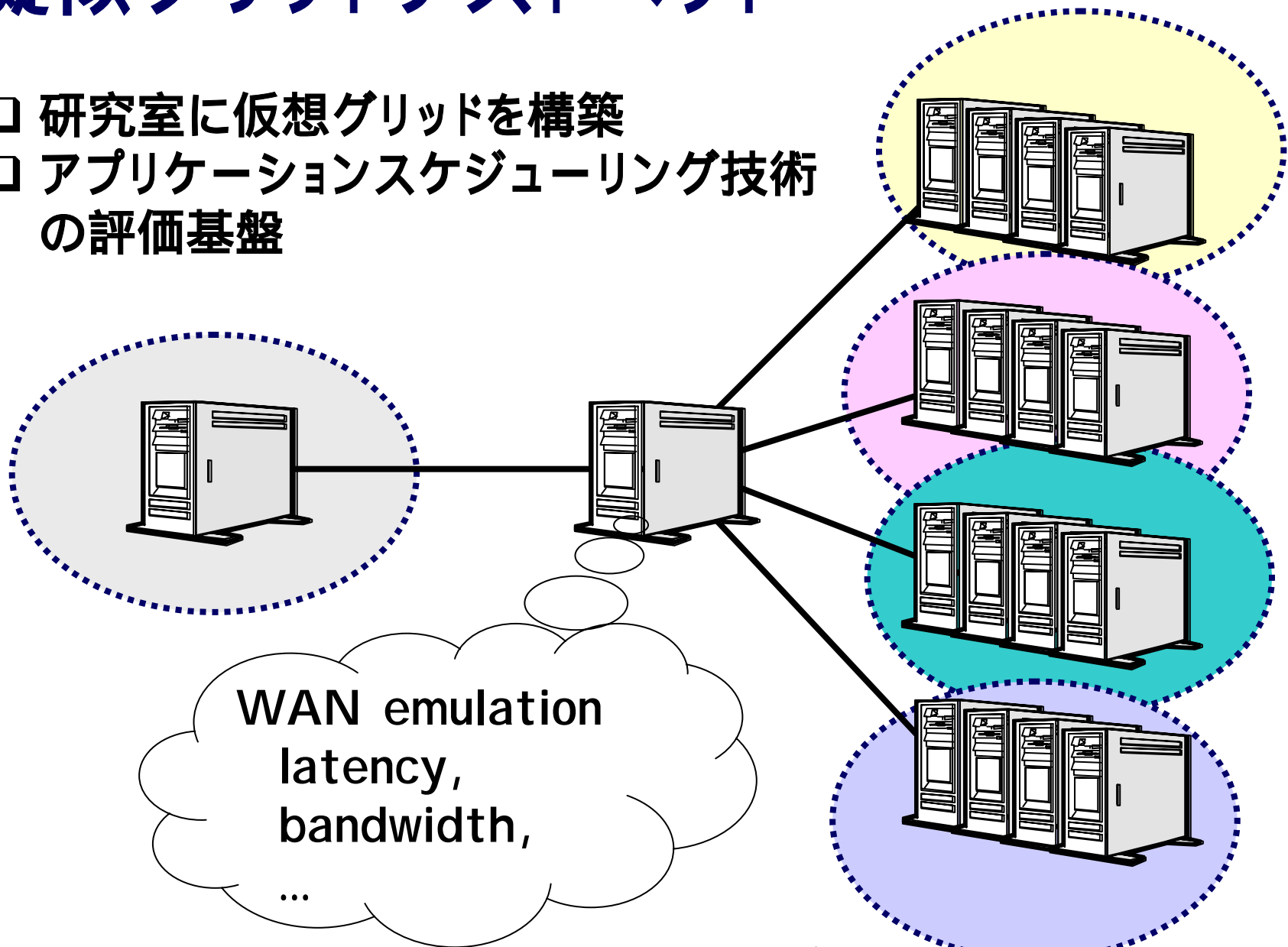
P1-P5:  $n_x=6$ ,  $n_y=6$ ,  $m=24$



- アイドル時間が少ないほど、負荷分散が成功.
- 0~11秒程度のアイドル時間
- より詳細な評価が課題.

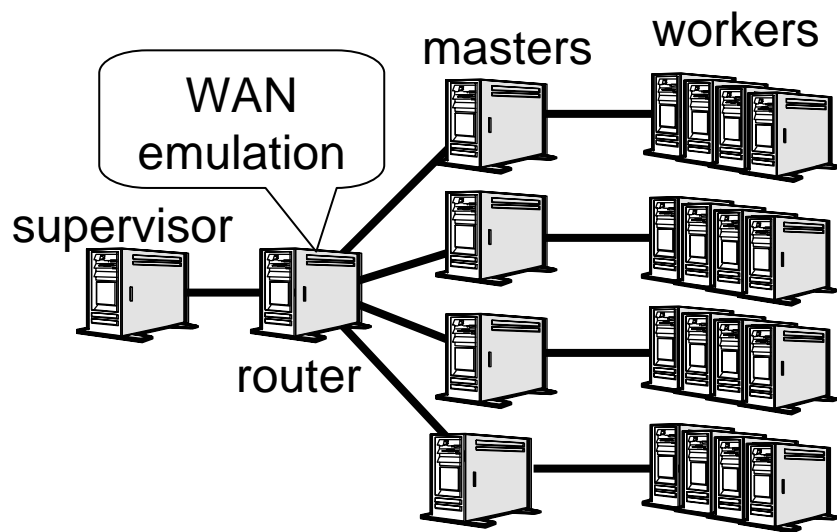
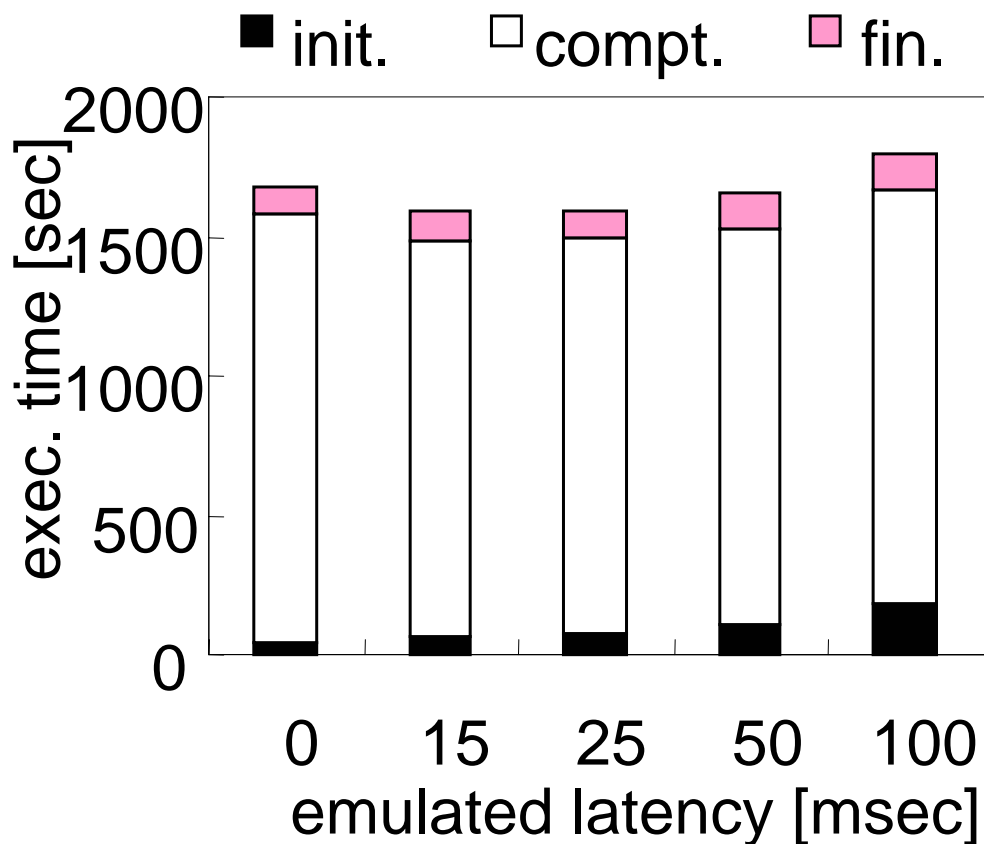
# 擬似グリッドテストベッド

- 研究室に仮想グリッドを構築
- アプリケーションスケジューリング技術の評価基盤



# 通信遅延の実行時間への影響

P2:  $n_x=6$ ,  $n_y=6$ ,  $m=24$

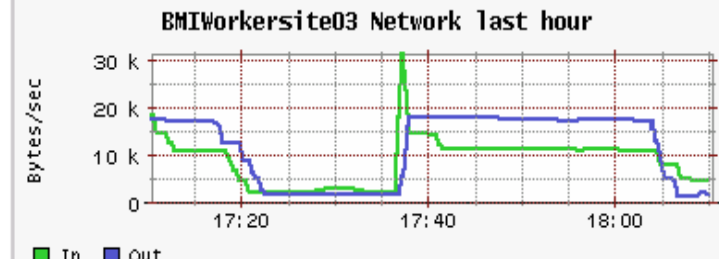
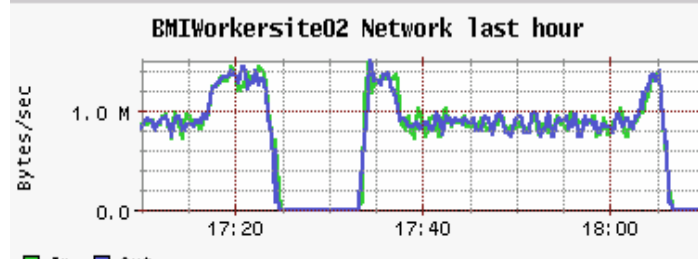
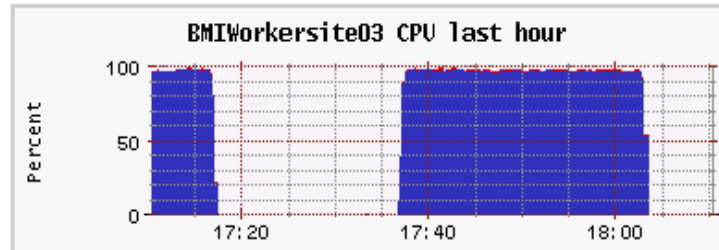
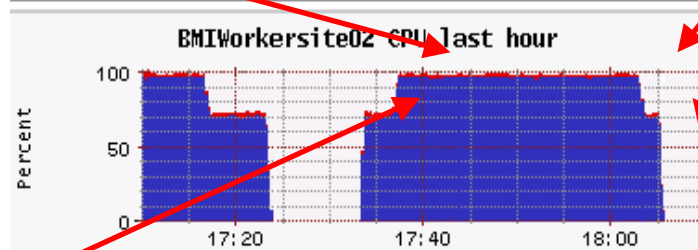
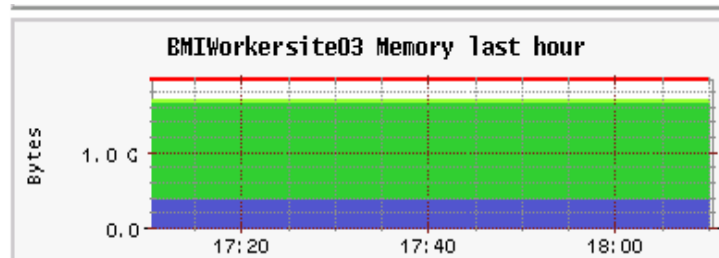
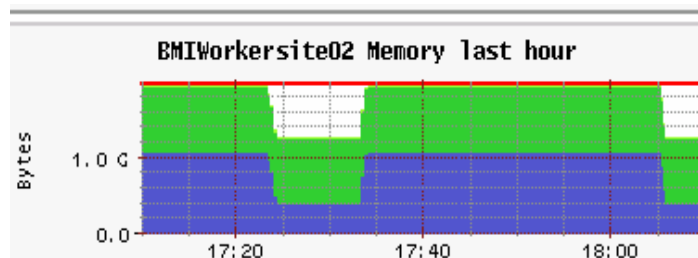


■ 高レイテンシの条件下でも、アプリケーションの性能低下は小さい。階層化の効果

# バックグラウンドジョブの実行状況

## (Ganglia)

例) タスク移動アルゴリズム: (A) 問題: RD06-02



BMI実行

BMI終了

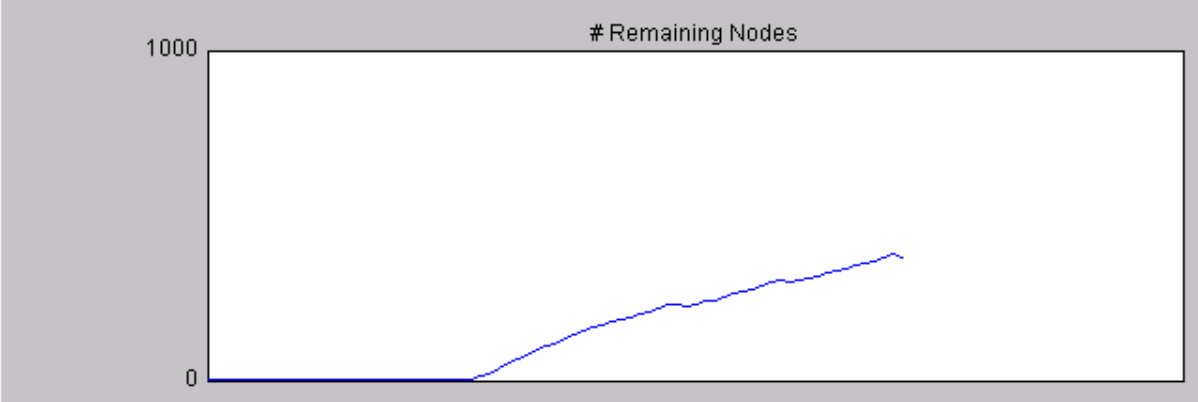
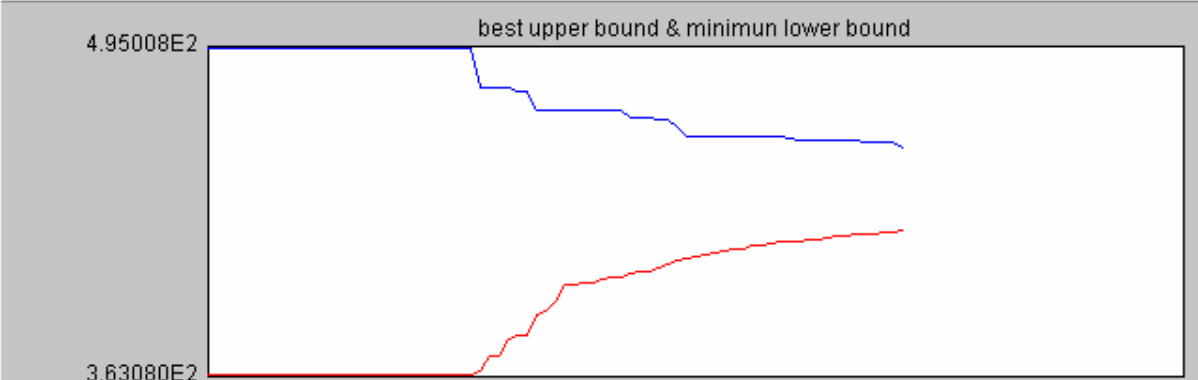
LU実行

LU終了

グループ2(他ジョブ有)

グループ3(他ジョブ無)

InputFile: RD06-01.dat [START] [STOP]



best upper bound: 4.54434E2 Minimum lower bound: 4.21040E2 # Remaining Nodes: 374

**solution:**

```
x=[  
8.895049E-02,  
-1.806776E-01,  
3.170630E-02,  
-1.240352E-01,  
-1.770128E-01,  
-4.314229E-02  
];  
...r
```

# まとめ

## ■ 成果

- PCクラスタおよびグリッド上での並列分枝限定法アプリケーションの実行方式を確立.
- グリッド上での階層的マスタ・ワーカ方式の有効性を示した.
- 最適化問題分野に対して, クラスタ / グリッド計算技術利用への道を開く.

## ■ 研究協力

産総研, 東工大, 東京電機大, 京大, 徳島大, 東大, UCSD

## ■ 今後の課題

- PCクラスタ間負荷分散アルゴリズム
- 擬似グリッドテストベッド

# 謝辞

- 本研究の一部は、科学技術振興機構計算科学技術活用型特定研究開発推進事業 (ACT-JST) 研究課題「コモディティグリッド技術によるテラスケール大規模数理最適化」の援助による
- 本研究について日ごろよりご助言いただいているNinfプロジェクトの皆様には感謝いたします

**Thank you.**