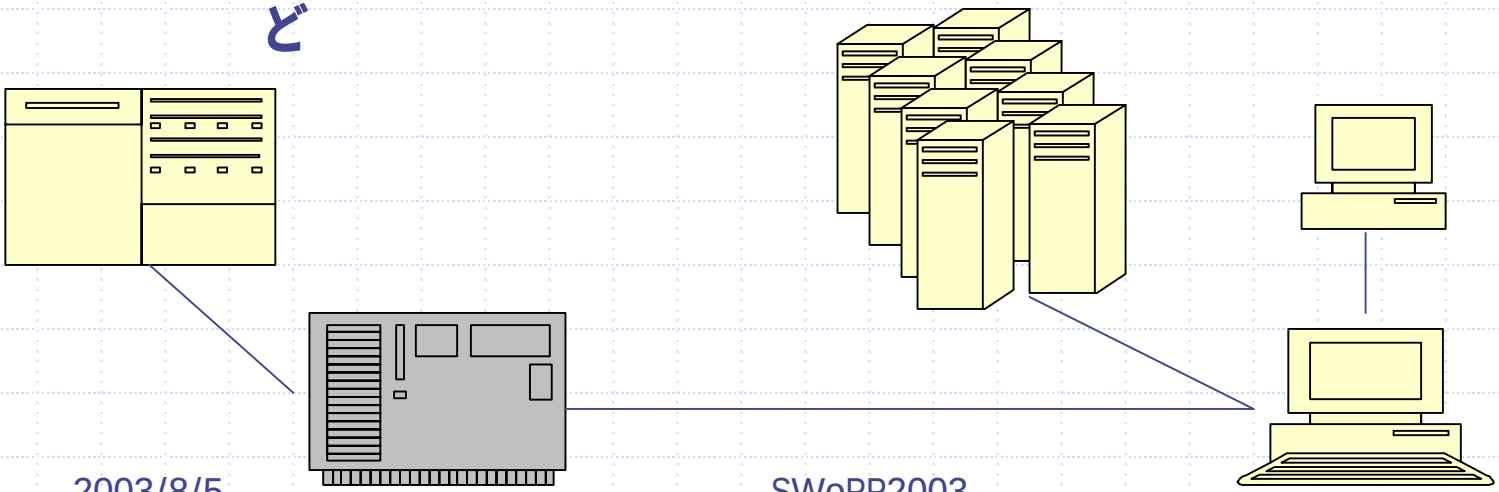
グリッドコンピューティングにおけるモニタリングシステムの自律的構成

白勢　健一郎[*1], 小川　宏高[*2]
中田　秀基[*1*2], 松岡　聡[*1*3]
[*1]東京工業大学
[*2]産業技術総合研究所
[*3]国立情報学研究所

本研究の背景

- グリッドコンピューティング
  - 広域の計算機資源を協調させて利用
- モニタリングの目的
  - 資源配分 … ジョブスケジューリングなど
  - 管理･運用 … プログラムや計算機の実行状況の把握、課金など



グリッドコンピューティングにおけるモニタリングシステム

- モニター対象
  - 計算機、ネットワーク、ミドルウエア、アプリケーション
- 特徴
  - 複数のコンポーネントから成立
    - データ収集、データ管理など役割を分割
    - 物理的に分散した複数の計算機からデータを収集
    - 幅広いモニター対象に対応
    - モニタリングシステムの障害発生の可能性が高くなる
  - スケーラビリティーが必要
  - 定期的にデータを供給することが求められる

モニタリングシステム管理の問題

- 複数の計算機上で稼動するコンポーネントに依存関係が存在
  - コンポーネント間のデータフローなど依存関係にもとづいて設定を決める煩雑さ
  - ネットワークトポロジーに基づいたコンポーネント配置
  - 障害、資源増減など状況の変化に応じた設定の動的変更への対応
- 短時間でコンポーネントの障害を復旧させることが必要
  - 人間が障害を見つけて修復するのを待つわけにはいかない

システムの自律化が必要

本研究の目的

- 自律的モニタリングシステムの設計と実現
  - モニタリングシステムの自律的管理の提案
    - 初期設定と障害復旧の自動化
  - モニタリングシステムの自律的管理機能の実装と評価
    - モニタリングシステムNWS [`99 Wolski et al.] を対象

モニタリングシステムの実装

- NWS [`99 Wolski et al.]
- MDS [Globus Project]
- R-GMA [EU Data Grid]
- Hawkeye [Condor Project]

共通性のあるのコンポーネントから成立[`03 Xuehai et al.]

Information Collector: **モニターデータ収集**
Information Server: **モニタデータを提供**
Aggregate Information Server: **複数の資源情報を集約**
Directory Server: **モニタする資源の探索**

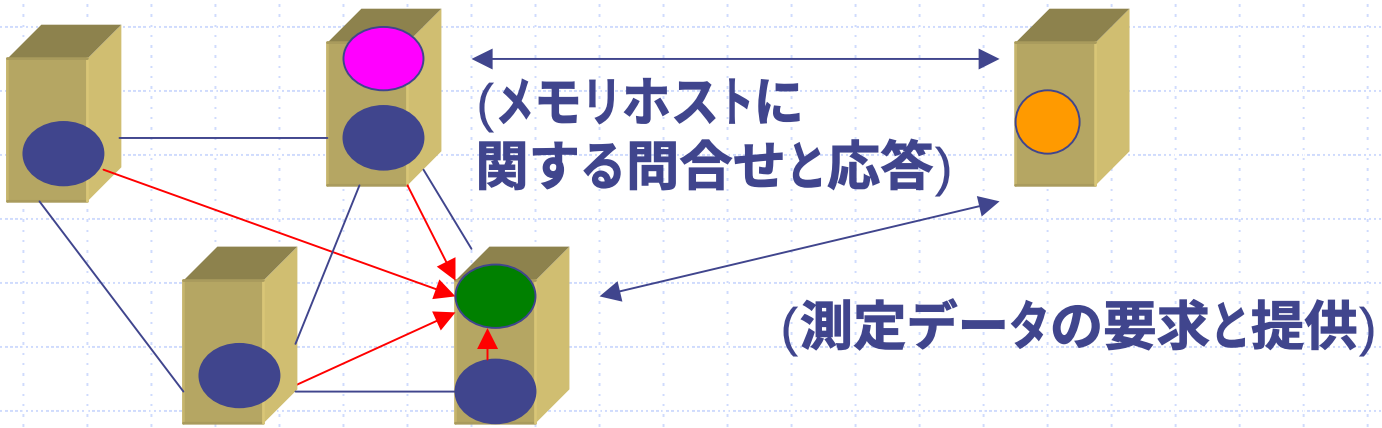
本研究では NWS **を対象としている**

NWSの概要

- CPU などの利用率やネットワークの性能を測定し、将来の利用予測を行う
- 3つのコンポーネントから構成
  - センサ[Information Collector]: データを集める
  - メモリホスト[Information Server]: センサーから送られてくるデータを一時的に保存、クライアントプログラムにデータを提供
  - ネームサーバ[Directory Server]: 他のコンポーネントに関する情報を管理
  - Aggregate Information Server に該当するものは無い

NWSでのデータ取得の仕組み

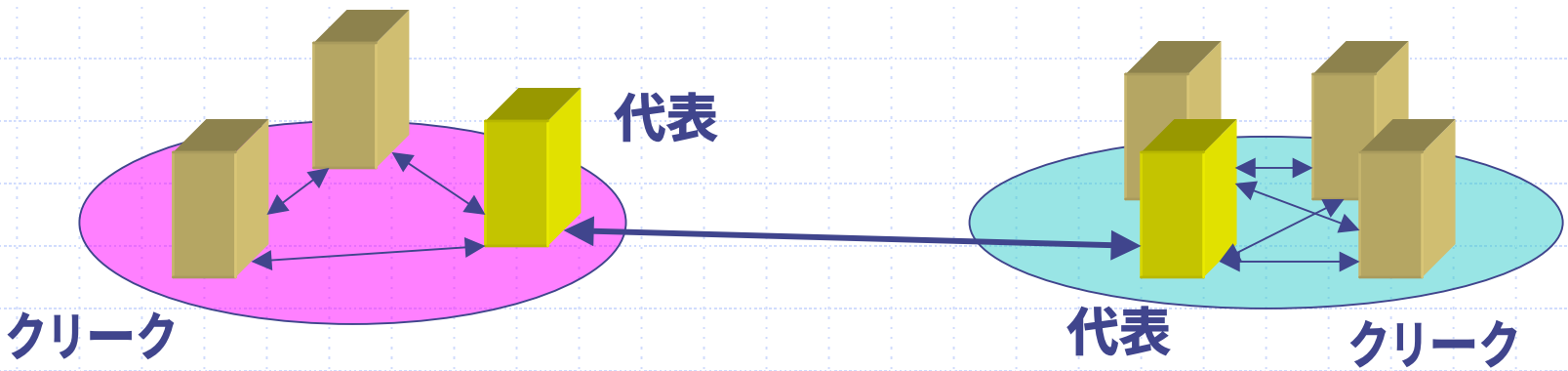
- クライアントのネームサーバへの問合せ
- 適切なメモリホストにへのデータの要求



：ネームサーバ　：メモリホスト　：センサ　：クライアント

クリークを用いたネットワーク性能計測

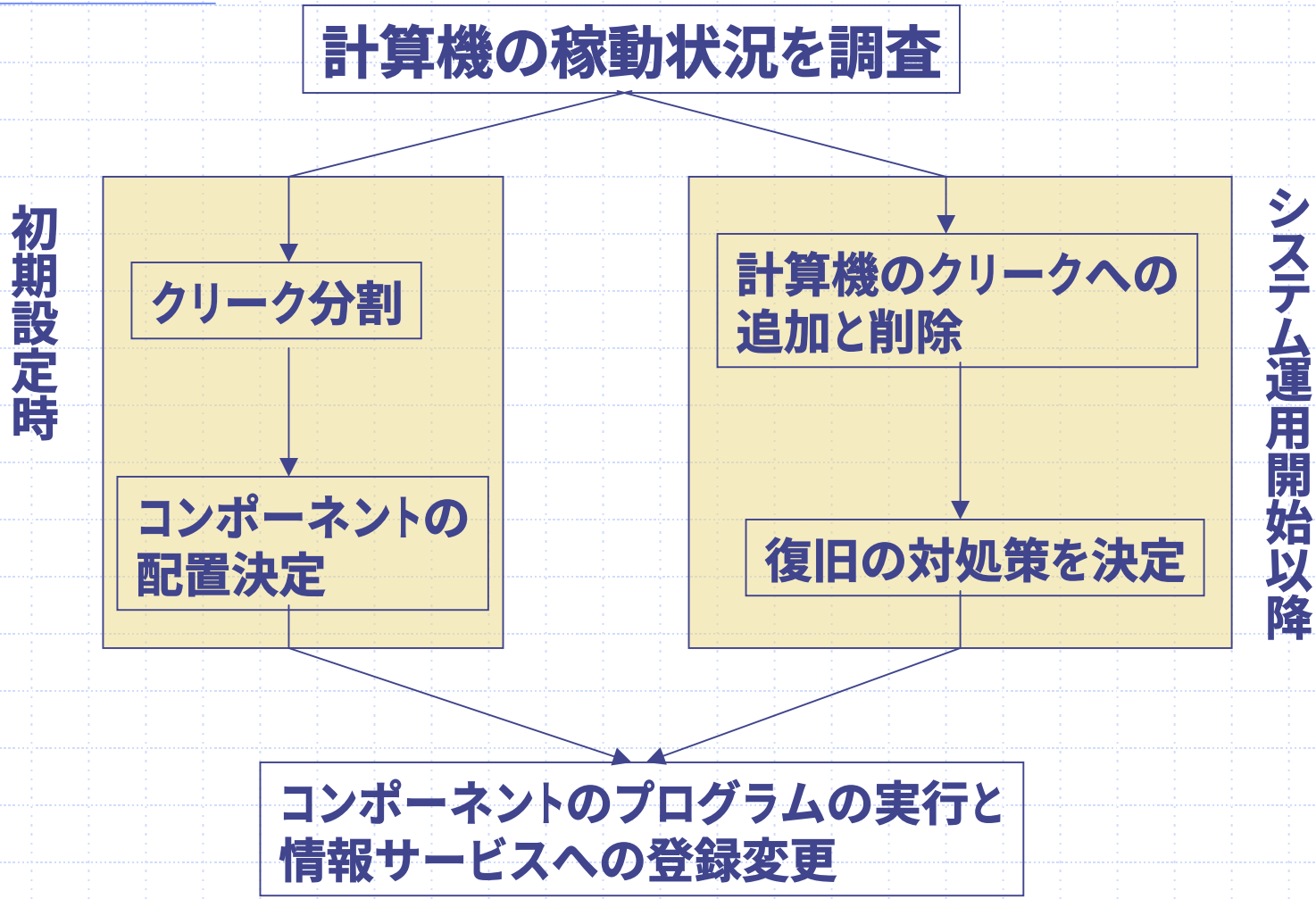
- 計算機をクリークに分割
  - End-to-end でのネットワーク性能計測にともなうトラフィックの悪化を軽減する事を目的
  - クリーク内では end-to-end でネットワークの性能を計測
  - クリーク間はクリークの代表間の測定結果から推定



自律的構成に求められること

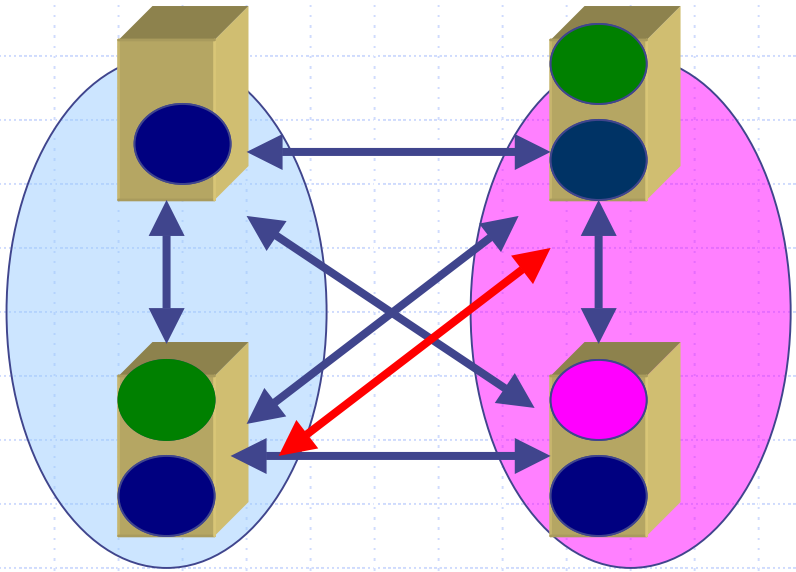
- 各コンポーネントが必要とする設定情報の自動決定
  - 計算機の現状、コンポーネントの依存関係から判断
  - ネットワーク性能計測のための計算機のクリーク分割を自動的に実行
- コンポーネントの障害発見と修復
  - コンポーネントの稼動状況を調べ、問題があるところは復旧

自律的構成の概要


計算機の稼動状況を調査
初期設定時
クリーク分割
コンポーネントの配置決定
システム運用開始以降
計算機のクリークへの追加と削除
復旧の対処策を決定
コンポーネントのプログラムの実行と情報サービスへの登録変更

自律的構成(1) -初期設定-

- 計算機やネットワークの現状を調査
- クリーク分割
- コンポーネントの配置決定
- コンポーネントの起動



NWS での例

：ネームサーバ

：メモリホスト　：センサ

：ネットワークの代表計測

自律的構成(2) -障害復旧-

- コンポーネント、計算機の稼動状況の調査
- クリーク操作
  - 不調な計算機をクリークから削除
  - 復旧または新規追加の計算機をグループに追加
- 障害復旧の対処法を決定
- コンポーネントの再起動

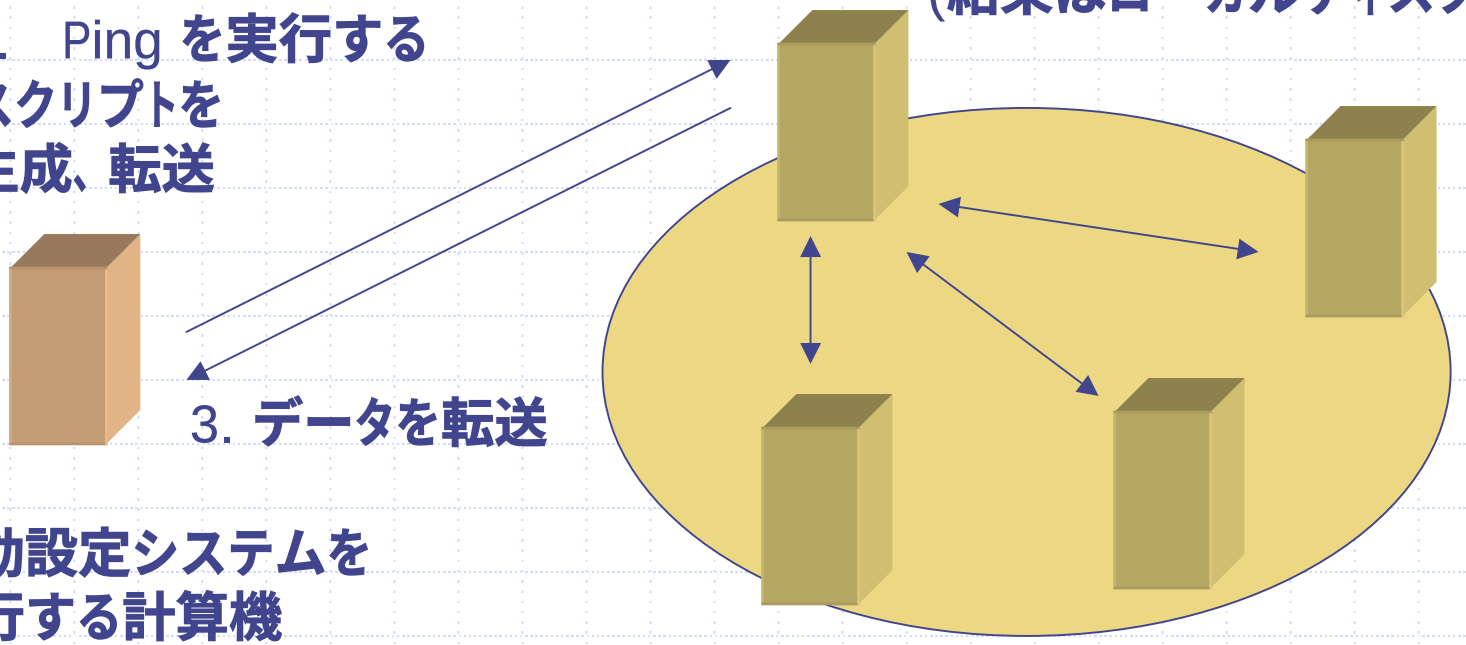
自律的構成システムの実装

- NWS のコンポーネントの配置決定と起動および障害復旧を自動で行うシステムを実装
- 必要な情報はNWSを動かす計算機のリスト
- 主な構成要素
  - 各ホスト間のRTTの測定
  - RTTに基づいたクリーク分割
  - コンポーネントの設定
    - ネームサーバ、メモリホストの配置
  - 障害復旧

クリーク分けに必要なデータ収集

2. 逐次処理でpingを実行
(結果はローカルディスクに保存)

1. Ping を実行する
スクリプトを
生成、転送



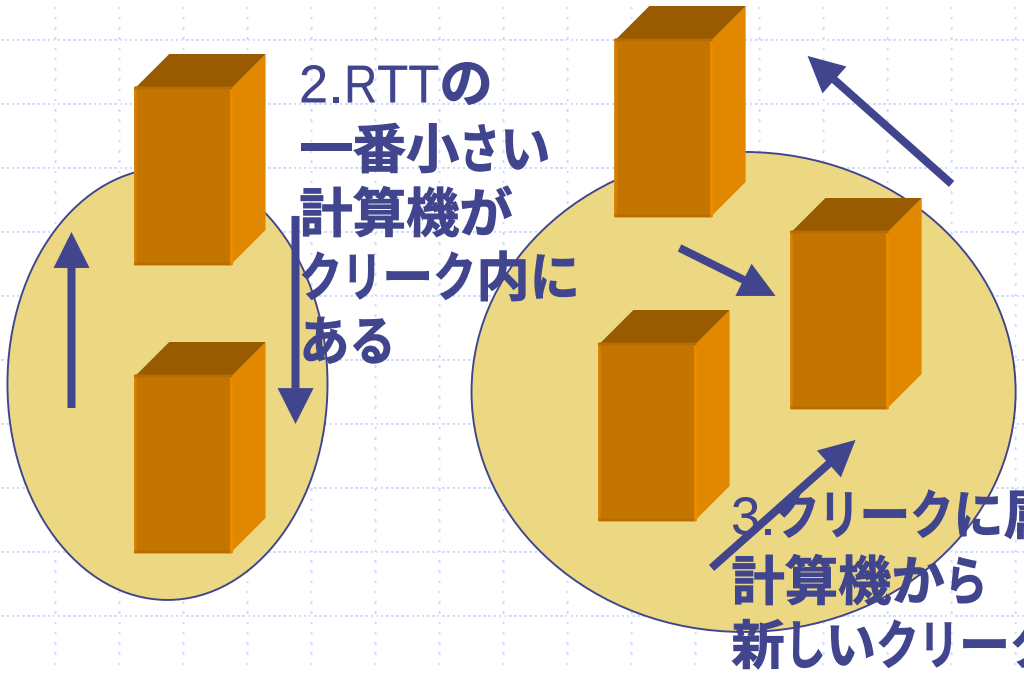
3. データを転送

自動設定システムを
実行する計算機

1. から3. をモニタリングシステムを設定するすべての計算機に
対して並列実行

クリーク分割

- RTTが小さい計算機でクリークを生成



:グループに所属した計算機

: RTTが一番小さい
計算機への指示

コンポーネントの設定と起動

- ネームサーバやメモリホストの稼動場所
  - ネームサーバ … 他の計算機とのRTTを一番多く測定できた計算機の中でRTTの平均が一番小さいもの
  - メモリホスト … クリーク内で「近い計算機」に一番多く選ばれたもの
    - クリークのネットワークの性能計測を行う代表も兼ねる
- 決まった設定をもとにシェルスクリプトを生成
- 生成されたスクリプトを実行しコンポーネントのプログラムを起動

障害復旧(1)

- 障害の判断基準
  - コンポーネントが動いている計算機の状態
    - SSHを用いてRTT計測シェルスクリプトを遠隔から実行できるかどうかで計算機が動いているかどうかを判断
  - コンポーネントが設定どおり動いているか
- 計算機が動いていて、動いているはずのコンポーネントは動いていない場合：既存の設定で再びコンポーネントを実行

障害復旧(2)

- 計算機が動いておらず、そこでネームサーバやメモリホストが動いている設定だった場合: 代替の計算機を決定
  - メモリホストの場合: 同じクリーク内で代替の計算機を決定
    - クリーク内のセンサを新しい設定で再起動
  - ネームサーバの場合: 全体の中から代替の計算機を選択
    - すべてのコンポーネントを再起動

実装したシステムの評価

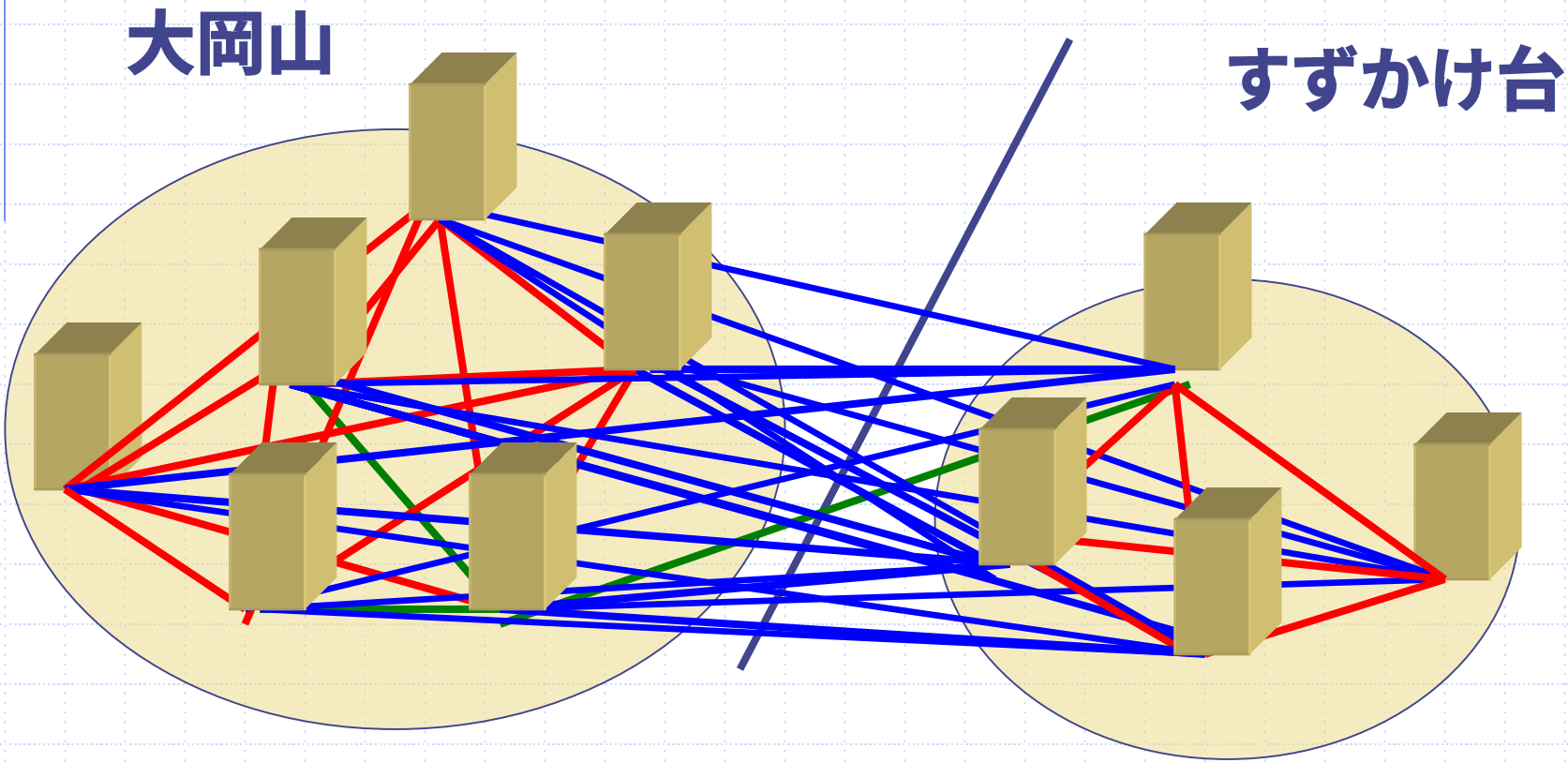
- 東京工業大学のTitech Gridを利用
  - PCクラスタをキャンパス15か所に配置
    - 全体で800を超えるプロセッサ、25TByteのストレージ
    - 各クラスタはギガビットのネットワークに直接接続
    - PCクラスタのノードのスペック
      - CPU: PentiumIII 1.4GHz X 2
      - メモリ: 512MB
      - NIC: 100Base-T
  - 各クラスタにつき1ノードを評価に利用

評価内容

- NWSの自動初期設定の結果と所要時間
- 障害復旧
  - メモリホストが動いている計算機が機能しなくなったときの対応結果
  - ネームサーバが動いている計算機が機能しなくなったときの対応結果

初期設定におけるRTT計測

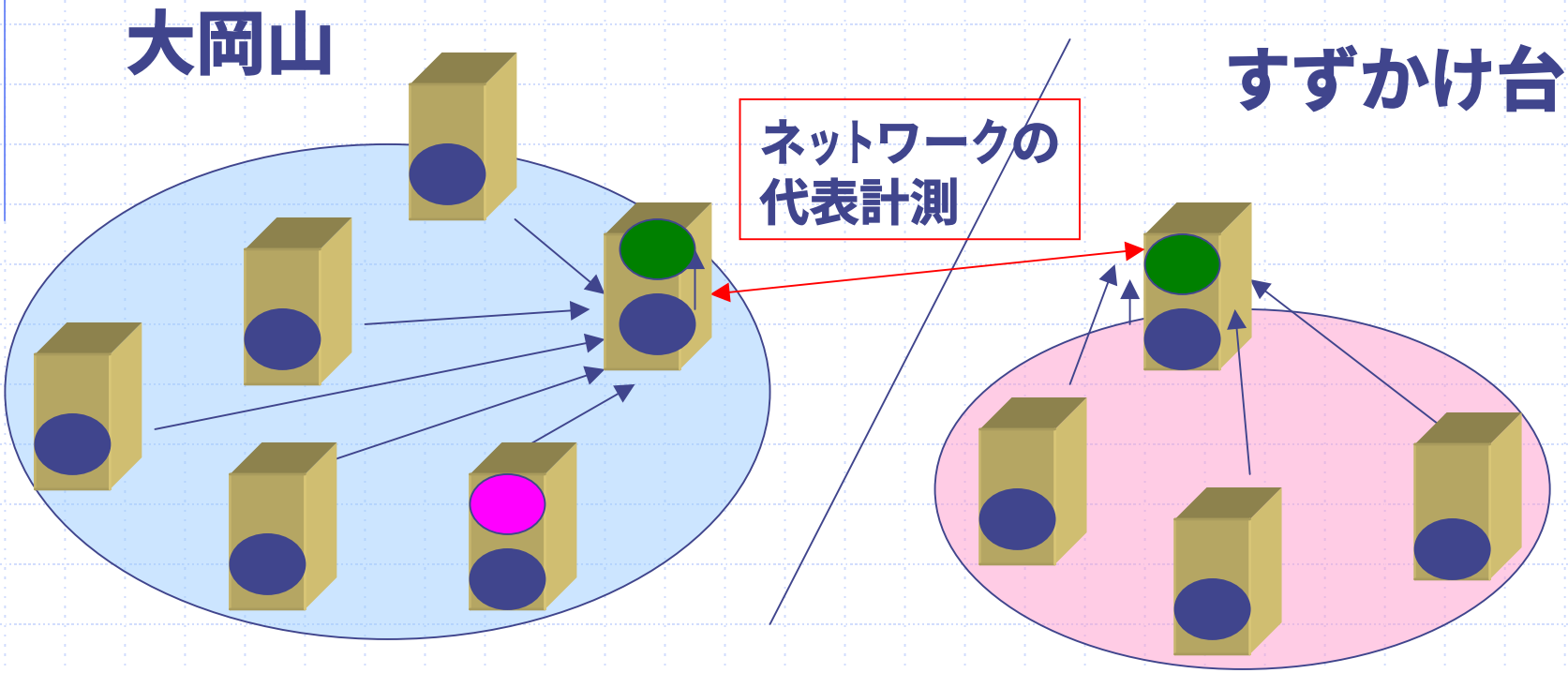
RTT測定値
- ：0.450ms未満
- ：0.450ms以上
- ：1000ms以上



初期設定の結果

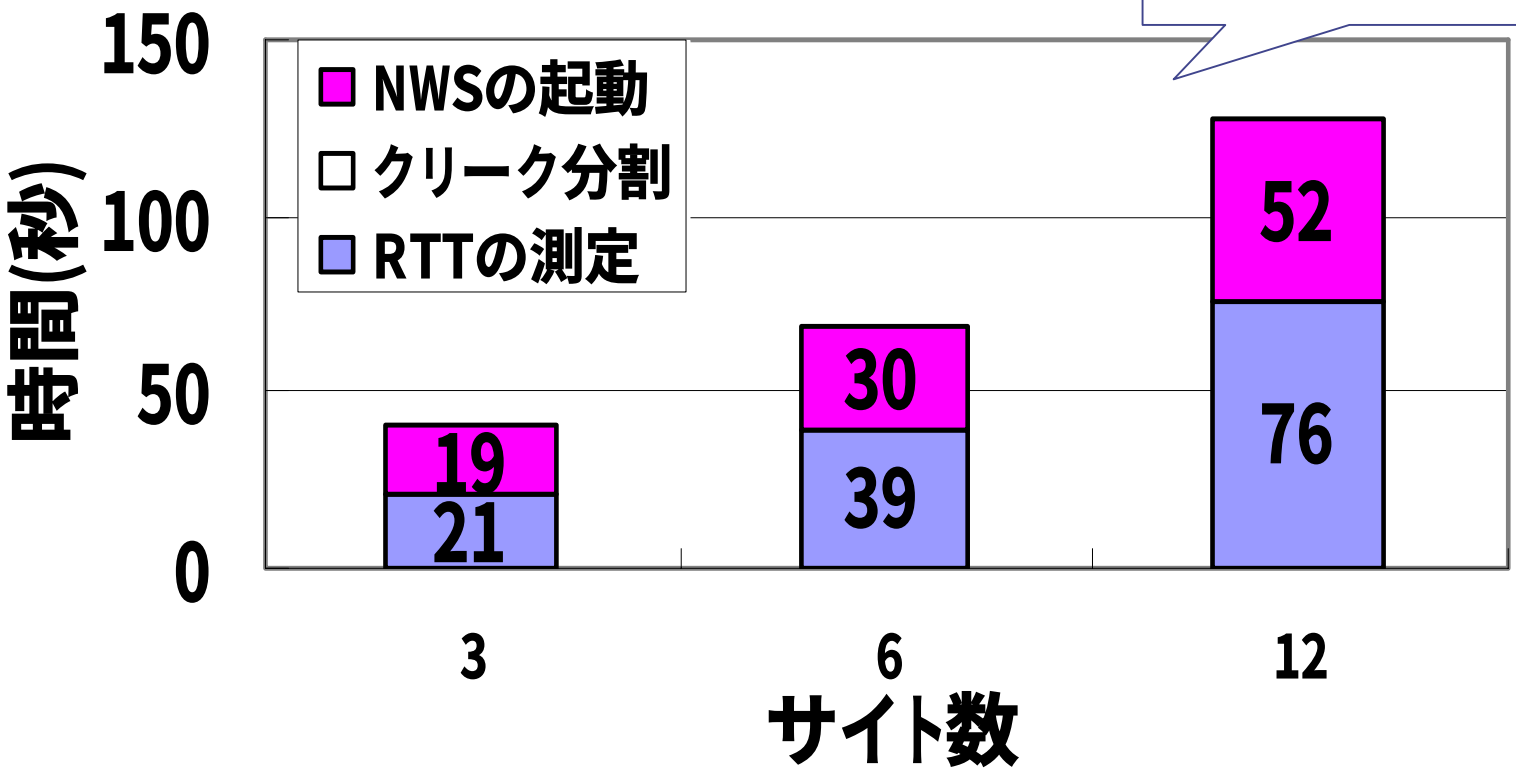
大岡山とすずかけ台で
クリーク分割が出来ている

ネームサーバ
メモリホスト
センサ
センサからメモリホスト
へのデータの流れ



自動設定の所要時間

サイトの数をNとすると実行時間がO(N)



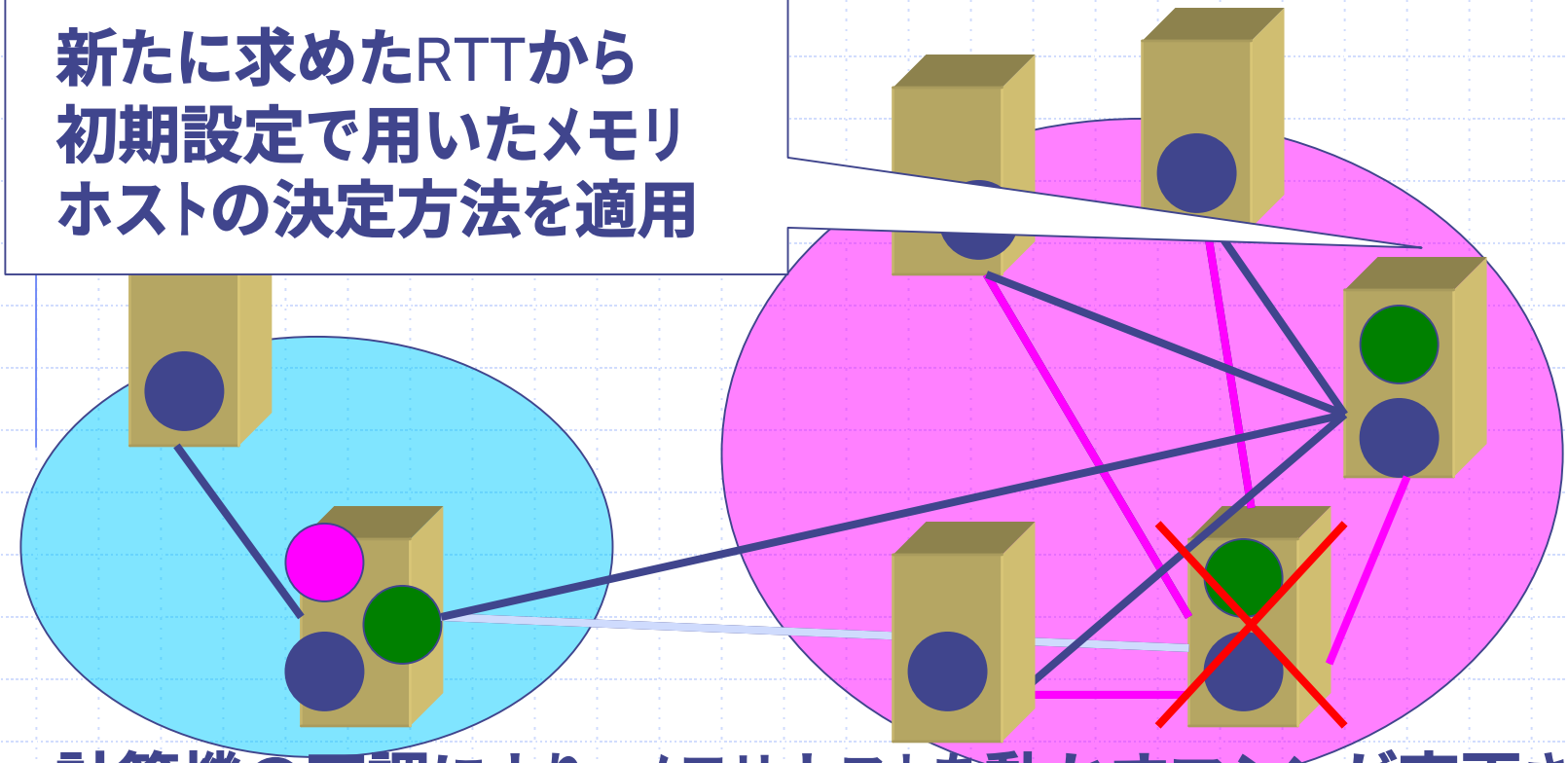
ほとんどの時間がRTT測定とNWSの起動に使われている

実行時間について

- 実行時間O(N)になる理由
  - 各計算機が行うRTTの測定、コンポーネントの起動を逐次処理で行っているため
- 本質的にはRTT測定やコンポーネントの起動にかかる時間は並列実行によってサイトの数に無関係となる

障害復旧:メモリホストの場合

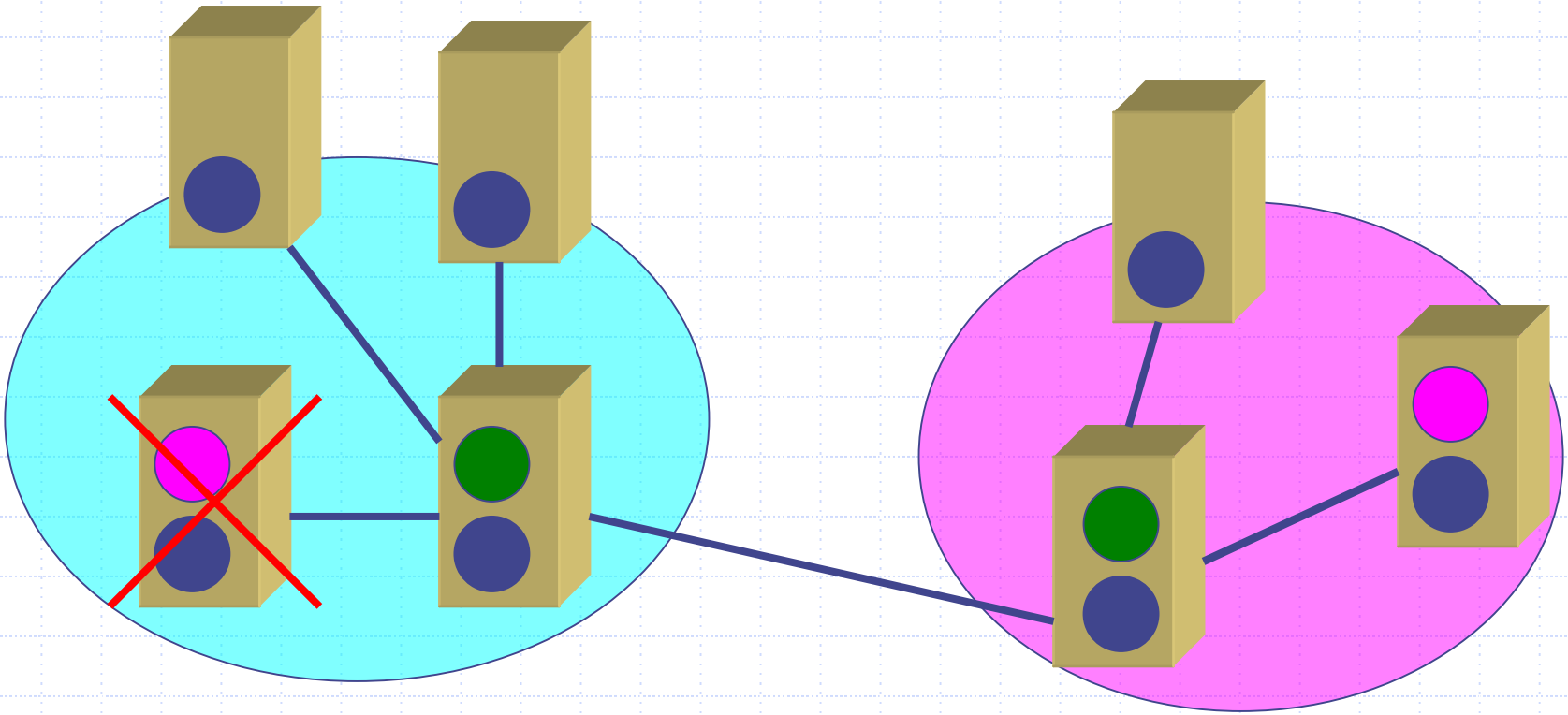
新たに求めたRTTから
初期設定で用いたメモリ
ホストの決定方法を適用



計算機の不調により、メモリホストを動かすマシンが変更された
グループ内のセンサーが新しいメモリホストにデータを提供

：ネームサーバ　：メモリホスト　：センサー

障害復旧:
ネームサーバの場合



ネームサーバの再配置が行われ、すべてのコンポーネントの再起動が行われた

：ネームサーバ　：メモリホスト　：センサー

障害復旧にかかる時間

- メモリホストが動いている計算機に障害が発生したときのケース (サイト数7)
  - 再設定: 1秒、再起動: 37秒
  - データ収集の部分は未測定 (意図的に計算機を不調な状態にすることができないため)

問題点

- モニタリングシステム内に single point of failure が存在
  - 原因は管理の集中化
  - 対処策
    - 管理機能の多重化
    - 計算機ごとに管理を行う
- RTT測定やコンポーネントの起動にかかる時間がO(N) (N; サイト数)
  - 現在逐次処理で行っている部分の並列化

まとめ

- モニタリングシステムの自律的構成の機構を提案
  - 計算機のグループ化、設定の決定、障害復旧
- NWSを対象とした自律的管理システムを実装
  - 15サイトの規模を持つキャンパス内テストベッドにおいて、初期設定と障害復旧をそれぞれ約2分で実行できることを確認
  - 手動設定より効率が良い事を確認

今後の課題

- Single Point of Failure の解消
  - モニタリングシステムの自律的構成の分散化
- NWSに特化した部分の汎用化と処理時間の改善
- 数百･数千のサイトの取り扱いへの対応
- 規模の大きな環境での実験･評価の拡充