PluS予約機構のCondorへの適用

1.産業技術総合研究所、2.数理技研

中田秀基[1]、竹房 あつ子[1]，大久保 克彦[1,2]

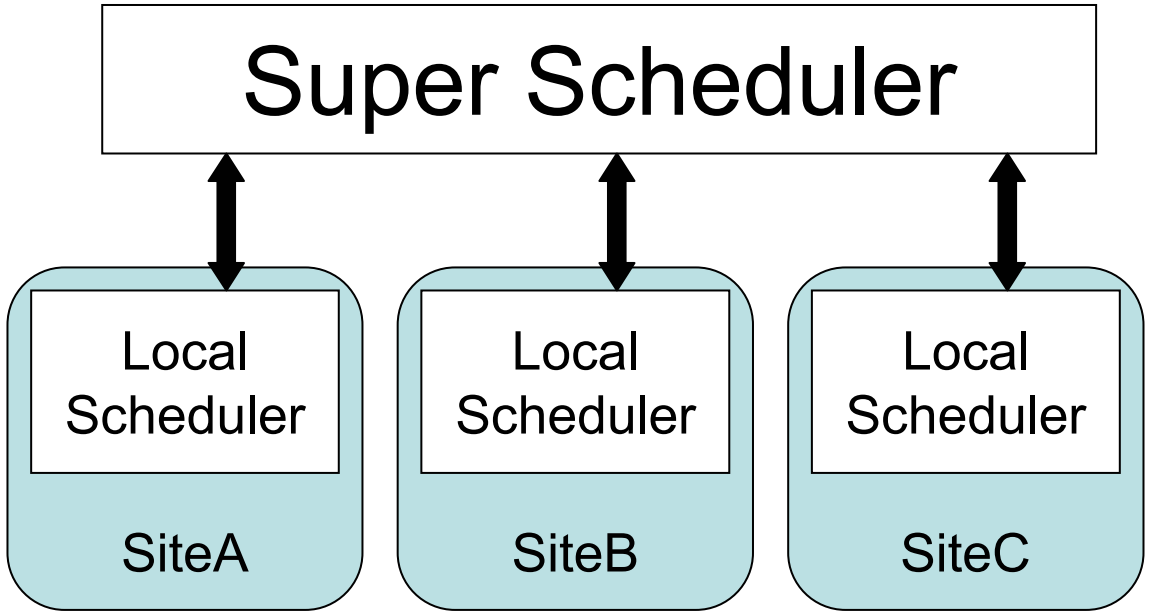
工藤 知宏[1]、田中 良夫[1]、関口 智嗣[1]

Grid Technology Research Center AIST

AIST

背景

- 複数ジョブの同時実行
  - 事前予約によるコスケジューリング
  - 上位に存在するスーパースケジューラがすべてのローカルスケジューラに事前予約を入れることによって，同時実行を実現



予約管理機構PluS

- 既存のローカルスケジューラに事前予約機能を付加
  - 可能な限りさまざまなローカルスケジューラに対応することが目的のひとつ
  - 現状ではSGE と TORQUEのみに対応
- 二つの動作モード
  - スケジューラ置換動作モード
    - 可搬性は低い
  - キュー操作動作モード
    - ターゲットローカルスケジューラを変更する必要がない
    - 理論的には可搬性が高い
    - **検証はされていない**

研究の目的

- 予約管理機構PluSの可搬性を確認
  - ローカルスケジューリングシステムのひとつであるCondorに対してPluSを適応
    - 記述量を評価
  - なぜCondor?
    - 広く用いられているため実用的な意味がある
    - SGEやTORQUEと構造がまったく異なるため，可搬性を確認するのに好適
      - Condorに対して可搬であれば他のローカルスケジューラにも可搬であろう

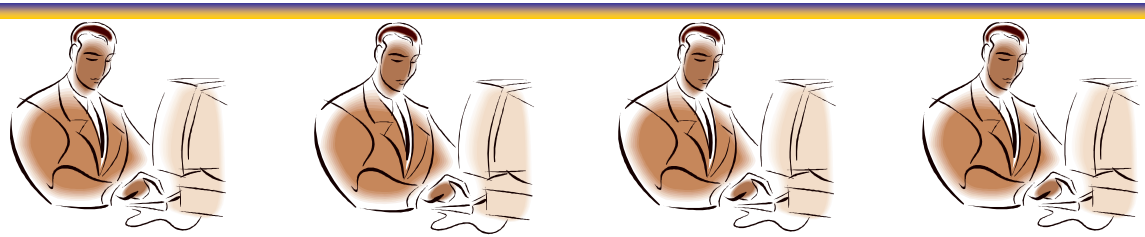
発表の概要

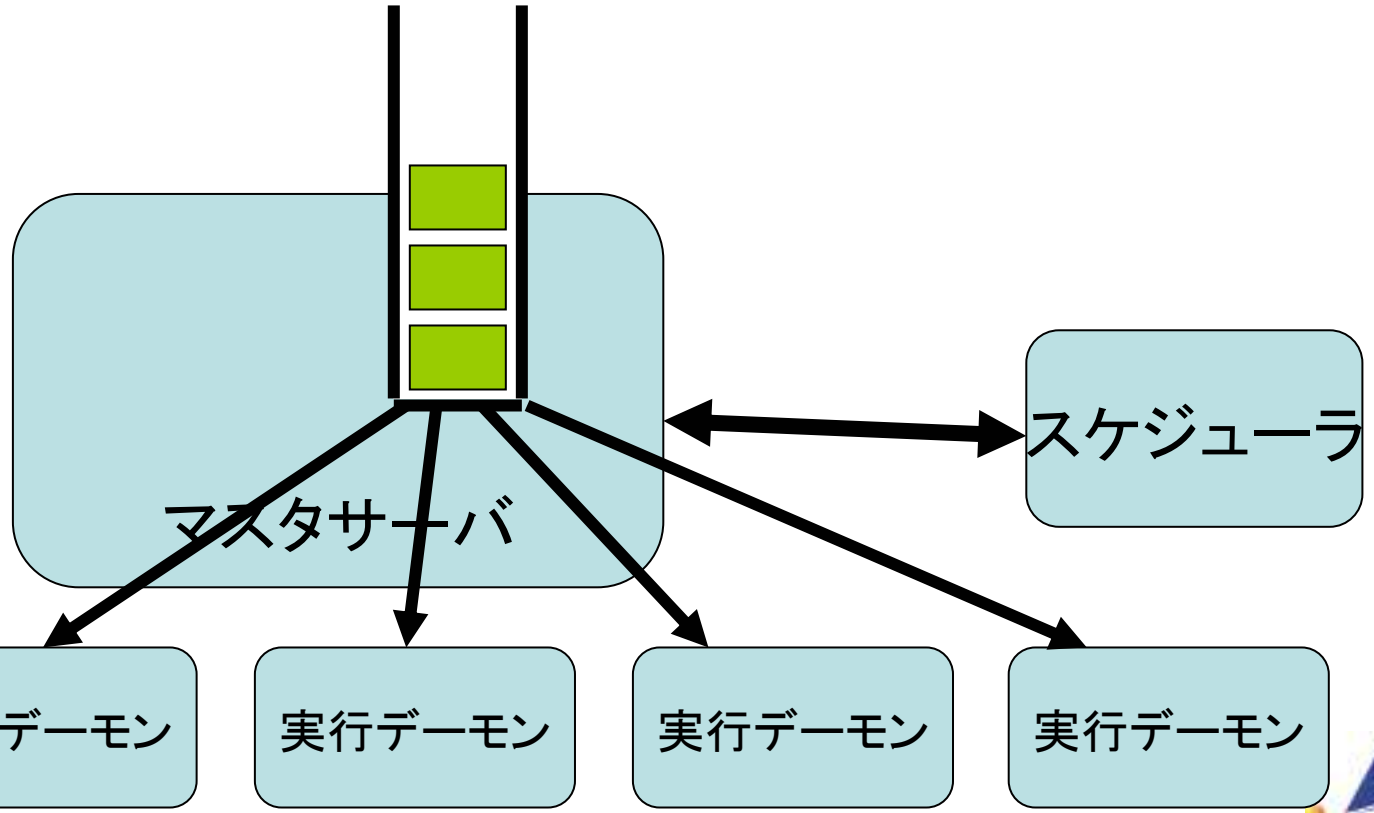
- 一般的なキューイングシステムとPluSの動作
- Condorの概要
- PluSのCondor適用
  - 設計，実装
- 測定
  - コード行数
  - 実行時間
- まとめと今後の課題

PluS予約管理機構

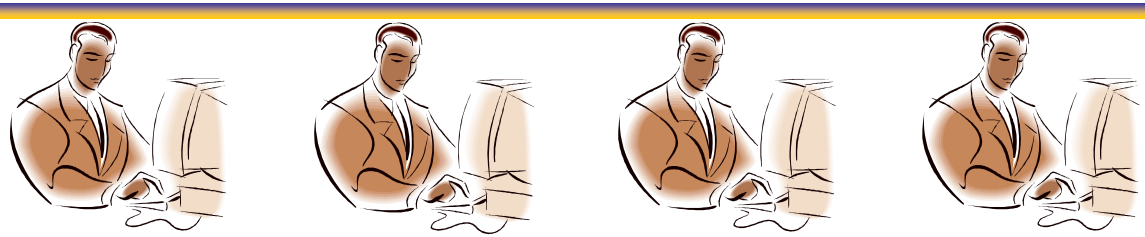
- 既存のバッチキューイングシステムと連携して動作
  - TORQUE, GridEngine
  - 事前予約機能を提供
- 2つの動作モード
  - スケジューリングモジュール置換法
  - キュー制御法
    - 既存のスケジューラをそのまま利用するので侵食性が低い

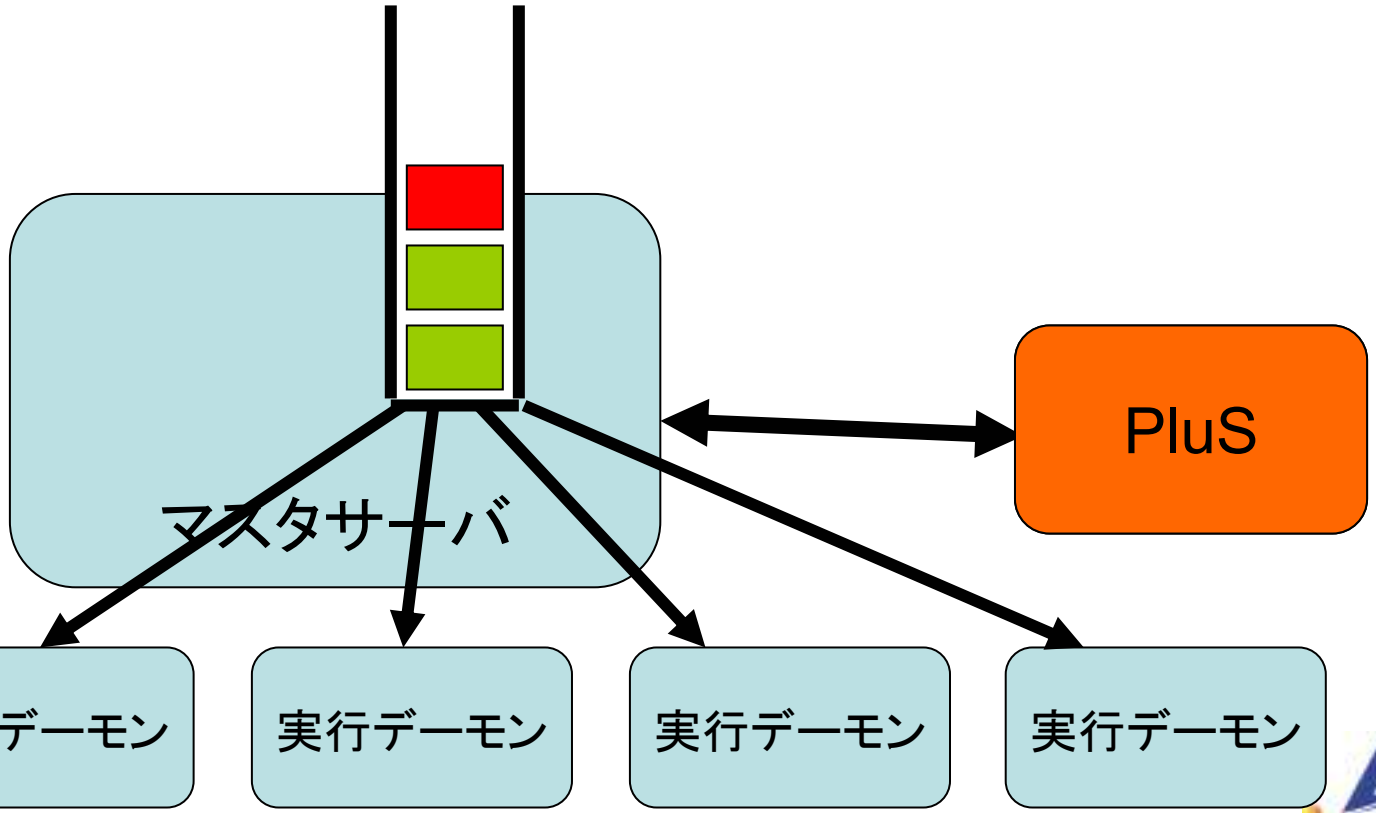
キューイングシステムの標準的な構成



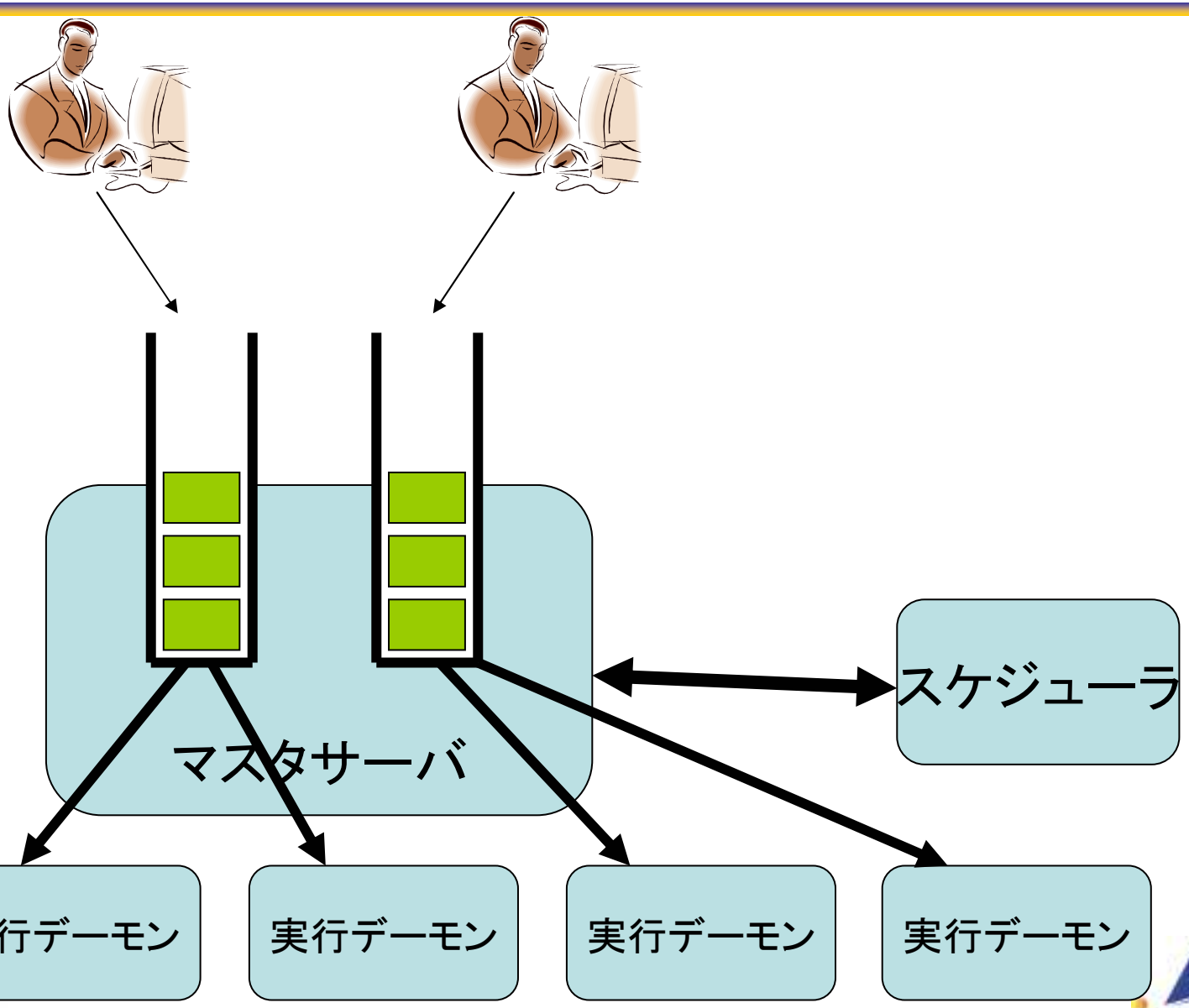


スケジューリングモジュール置換法

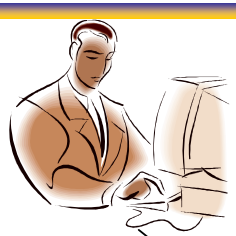




キュー制御による事前予約(1) キューとは



キュー制御による事前予約(2)



予約リクエスト

PluS
予約管理

スケジューラ

マスタサーバ

実行デーモン

実行デーモン

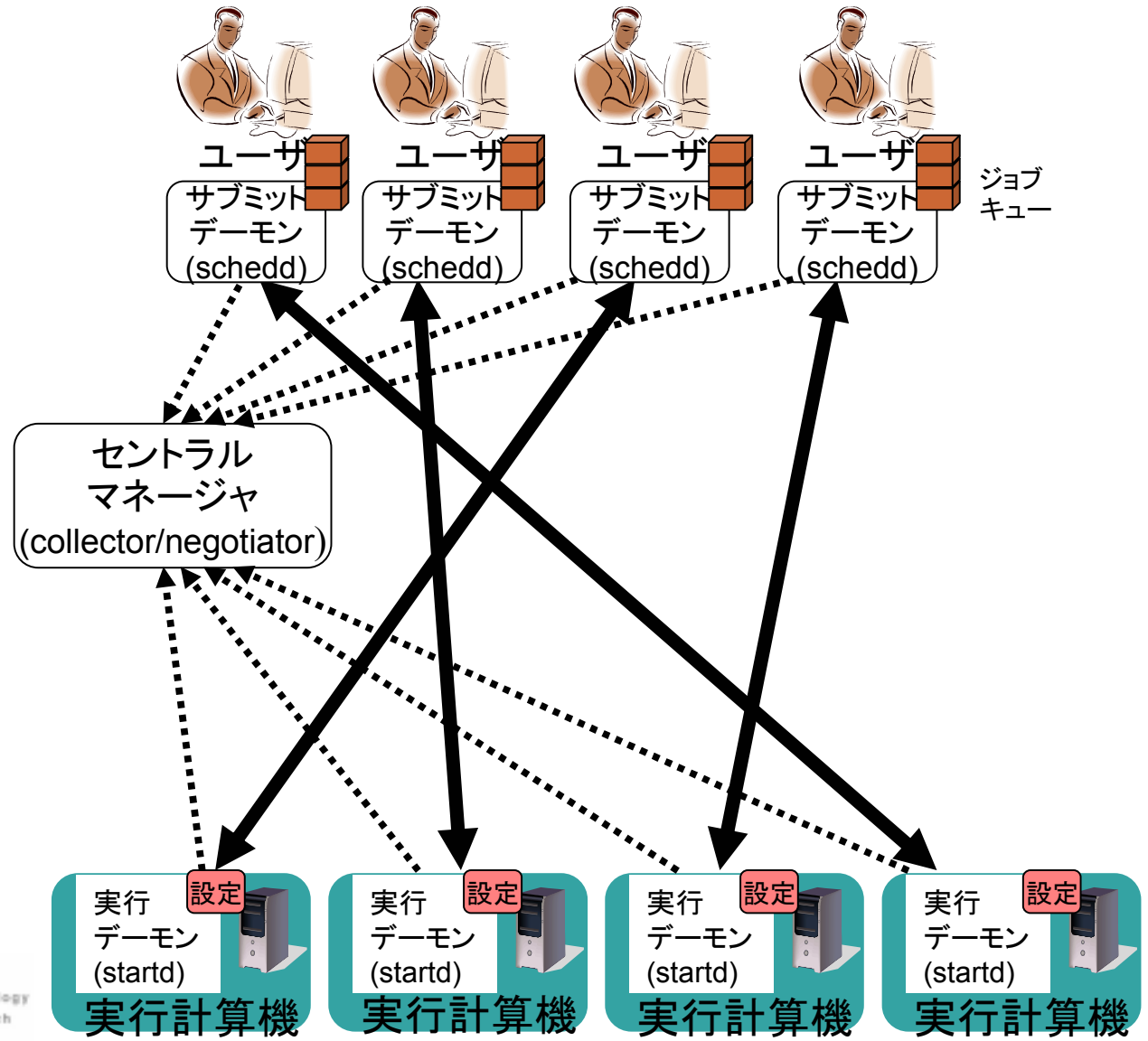
実行デーモン

実行デーモン

Condorの概要

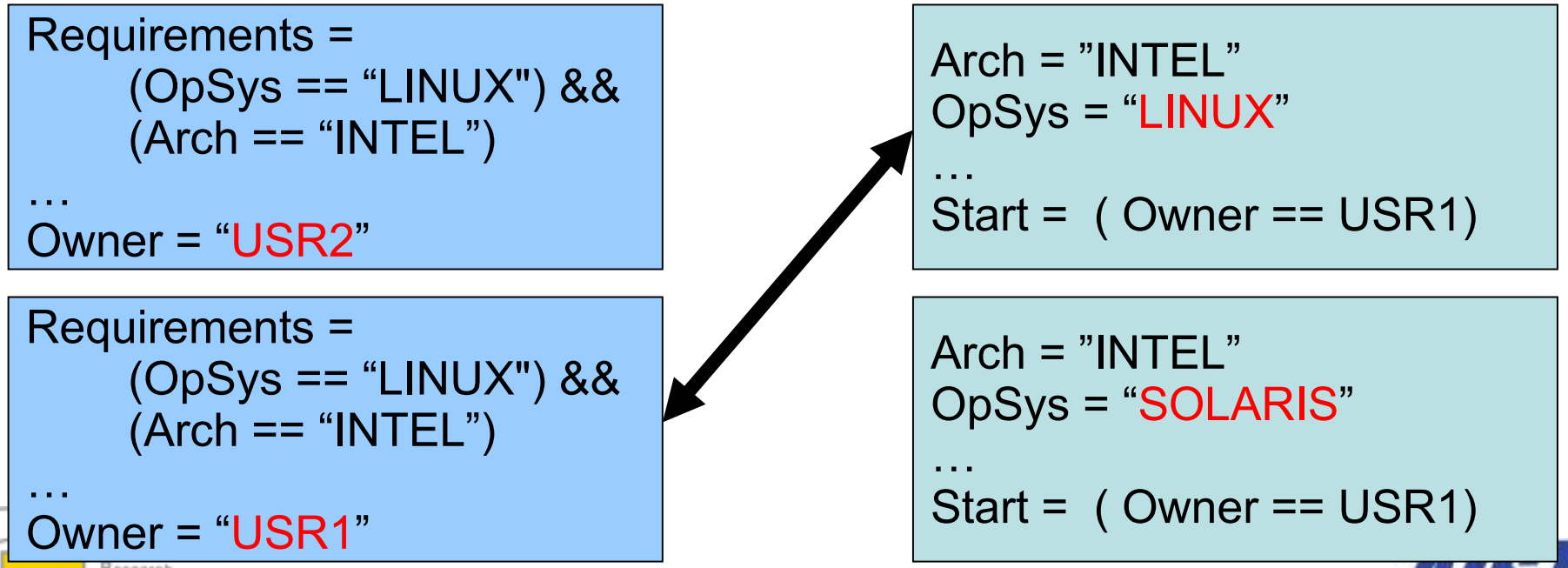
- ウィスコンシン大学で開発されたローカルスケジューリングシステム
  - 1985年から22年の歴史
- 遊休計算機の有効利用が当初のコンセプト
  - グリッド上のメタスケジューラとして
  - 専用クラスタのジョブスケジューリングシステムとして
- ClassAdによる柔軟な管理が特徴

Condorのノード構成



ClassAdの概要

- ある種のポリシ言語
  - ジョブと実行計算機のそれぞれに属性を定義
  - ジョブと実行計算機がそれぞれpreferenceを定義
  - 自由に拡張可能

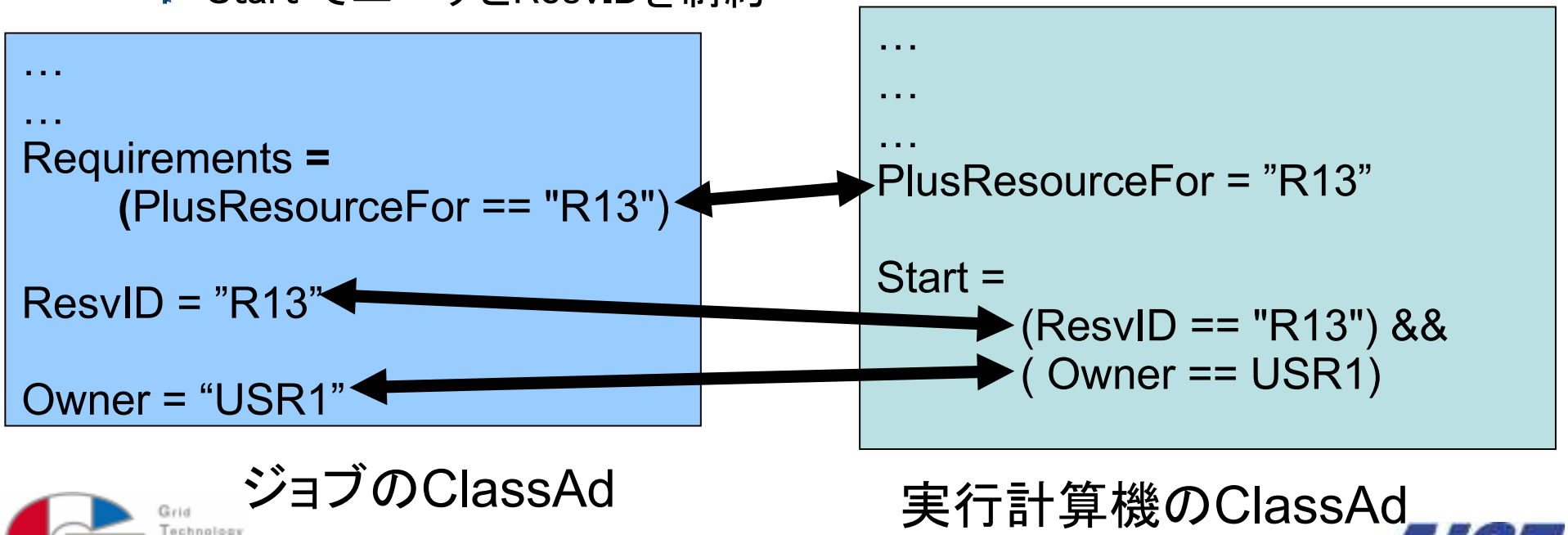


PluS for Condor の設計

- Condorには「キュー」という概念がない
  - 特定の実行計算機群を特定のユーザ群による特定のジョブ群に対して占有させる機構があればよい.
  - ジョブと実行計算機群のClassAdを変更することで実現
- 実現方法
  - 予約IDをタグとして使用

ClassAdによるジョブと実行計算機の束縛

- ジョブClassAd
  - 予約IDを ResvID で宣言
  - PlusResourceFor　がResvIDであるノードでだけ実行
- 実行計算機のClassAd
  - 予約IDをPlusResourceFor で宣言
  - Start でユーザとResvIDを制約



ジョブのClassAd

実行計算機のClassAd

Grid Technology Research Center AIST

AIST

ClassAdによるジョブと実行計算機の束縛

Owner = “USR1”

Requirements =
(PlusResourceFor == "R13")

ResvID = ”R13”
Owner = “USR1”

Requirements =
(PlusResourceFor == "R13")

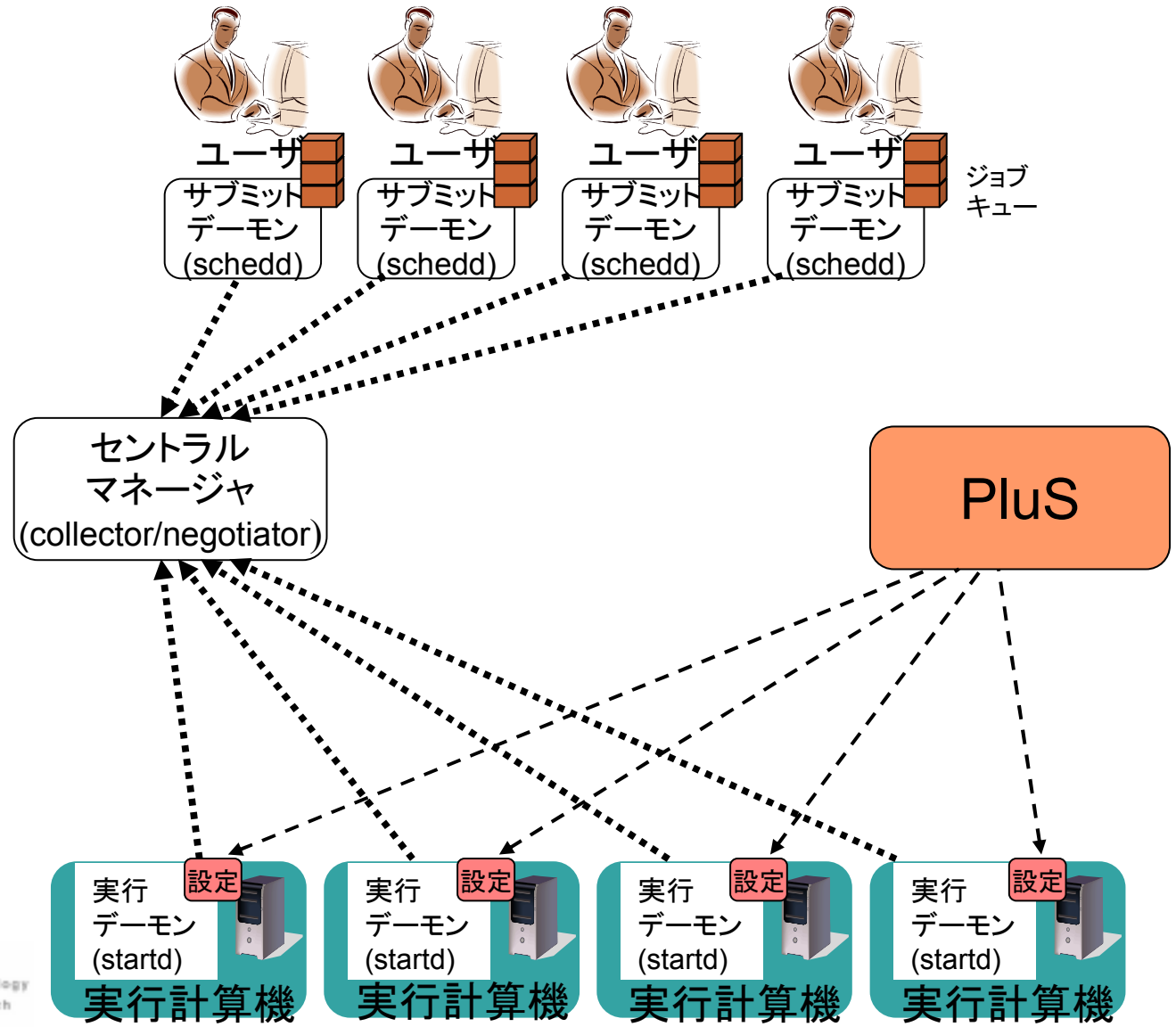
ResvID = ”R13”
Owner = “Other”

PlusResourceFor = ”R12”
Start =
(ResvID == "R12") &&
(Owner == USR1)

PlusResourceFor = ”R13”
Start =
(ResvID == "R13") &&
(Owner == USR1)

Start = (Owner == USR1)

実装



Grid Technology Research Center AIST

AIST

実装

- PluSの定義するSPI(Service Provider Interface)に対して実装を与える

名前	種別	意味
JobID	I/F	ジョブID
JobStatus	I/F	各ジョブの状態
NodeStatus	I/F	各実行計算機の状態
QueueStatus	I/F	予約を示すジョブキューの状態
QInstanceStatus	I/F	予約を示すジョブキューの各実行計算機上の状態
MainServer	I/F	プログラムのエントリポイント
ReserveInfo	A/C	各予約を表す
ReserveManager	A/C	予約時間開始時，予約時間終了時の動作を記述する
ServerStatus	I/F	PluSモジュールの状態

実装

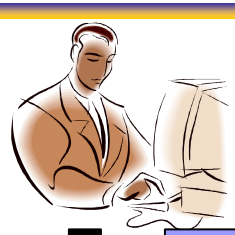
- 制御スクリプトはPythonで記述
  - condor_config_val –set
    - 各ノードのClassAd を動的に変更
  - condor_reconfig
    - 変更を反映
  - 各ノードにスレッドを割り当て
    - シリアライズされることをさけるため

サブミット時補助

- サブミット時に下記の作業が必要
  - Requirements に PlusResourceFor == RID を追加
  - ResvID == RID を追加
- サブミットファイルにいちいち書くのは非常に面倒
  - ポータルやジョブ投入サービス経由の場合には問題ないが，テスト等でも不便
- ラッパスクリプトで対処
  - plus_condor_submit -reserveId RID SUBMIT_FILE

サブミット補助

- HOLD 状態でジョブ投入
- ジョブのClassAdを改変
  - Requirements
  - ResvId
- HOLD状態を解除



executable = /bin/hostname

plus_condor_submit

executable = /bin/hostname
hold = on

condor_qedit

condor_submit

サブミット
デーモン

Requirements =
(Arch ==“INTEL”) &&
(OpSys == “LINUX”) &&
(PlusResourceFor == “R14”)
ResvId = “R14”
executable = /bin/hostname

測定

- 記述コード行数
  - 可搬性を確認
- 制御にかかる時間を計測
  - Condor向けPluSの実用性を確認

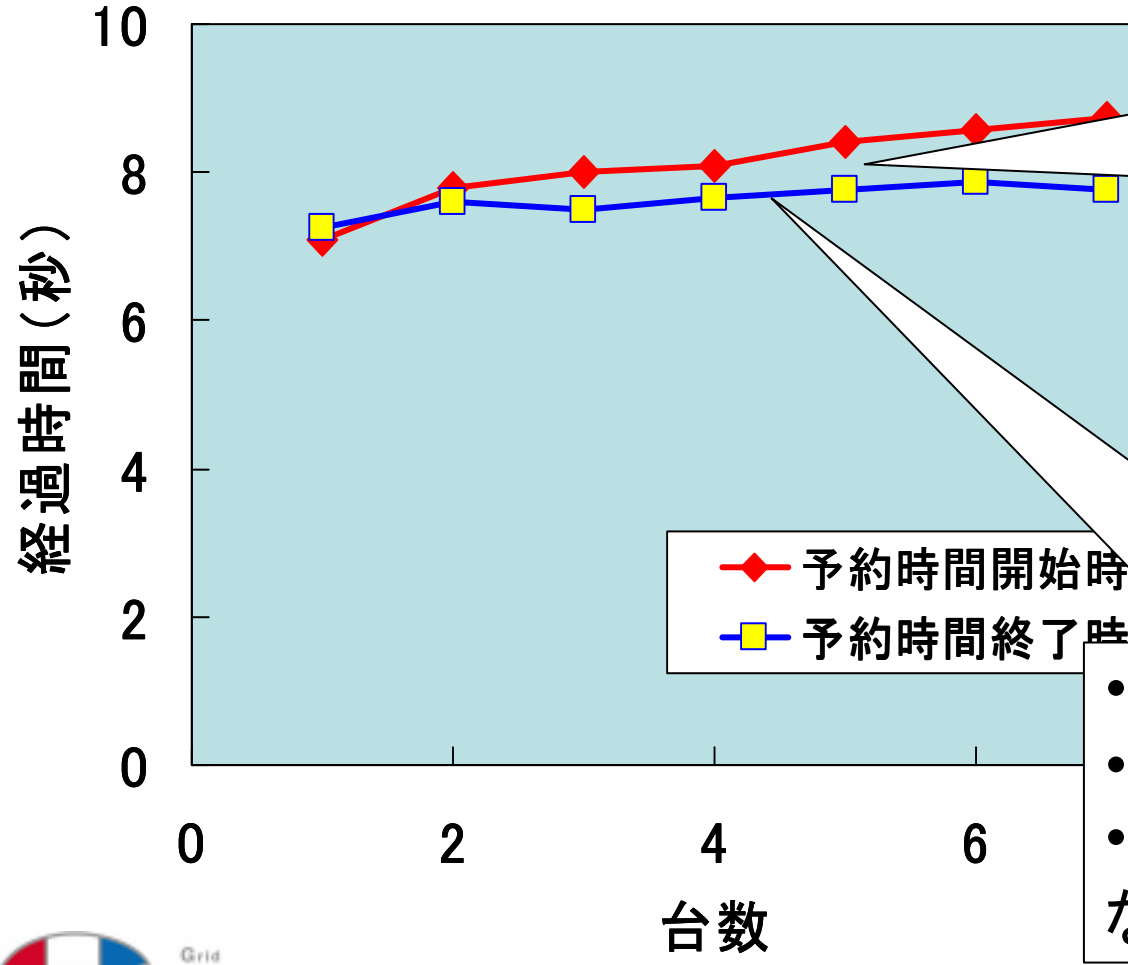
コード行数

種別	ファイル数	行数
Java	10	1068
python	2	324
sh	1	75
総計	13	1467
参考：SGE版総計		9800

- Javaコードのかなりの部分は，eclipse の「abstract method を実装」機能で自動的に生成

予約実現時のコスト

Pentium4



- 7秒-9秒程度
- ほとんどがcondor_reconfig

た測定プログラムから外部Pythonスクリ

- 台数増加の影響は軽微
- マルチスレッドの効果
- 100台規模では問題になる可能性がある

Grid Technology Research Center AIST

AIST

まとめ

- PluSをCondorに適用
  - Condor: SGEとはまったく異なるコンセプトに基づく
  - PluSの可搬性を確認
    - 実行速度は許容範囲
    - 所要工数は比較的小規模

今後の課題

- PluSのSPIのリファクタリング
  - キュー制御モードとスケジューラ置換モードがSPIを共有
  - 一方を実装する場合には，他方用の空メソッドを実装しなければならない．
  - 抽象クラスをうまく用いて整理
- セキュリティ設定
  - Condorのリモートからの管理設定を許可することで実現
  - 実運用するためには，認証を用いたセキュリティを導入する必要がある．

謝辞

- Wisconsin大学Condor チームの Jaime Frey 氏に感謝する.

- 本研究の一部は，文部科学省科学技術進行調整費「グリッド技術による光パス網提供方式の開発」による