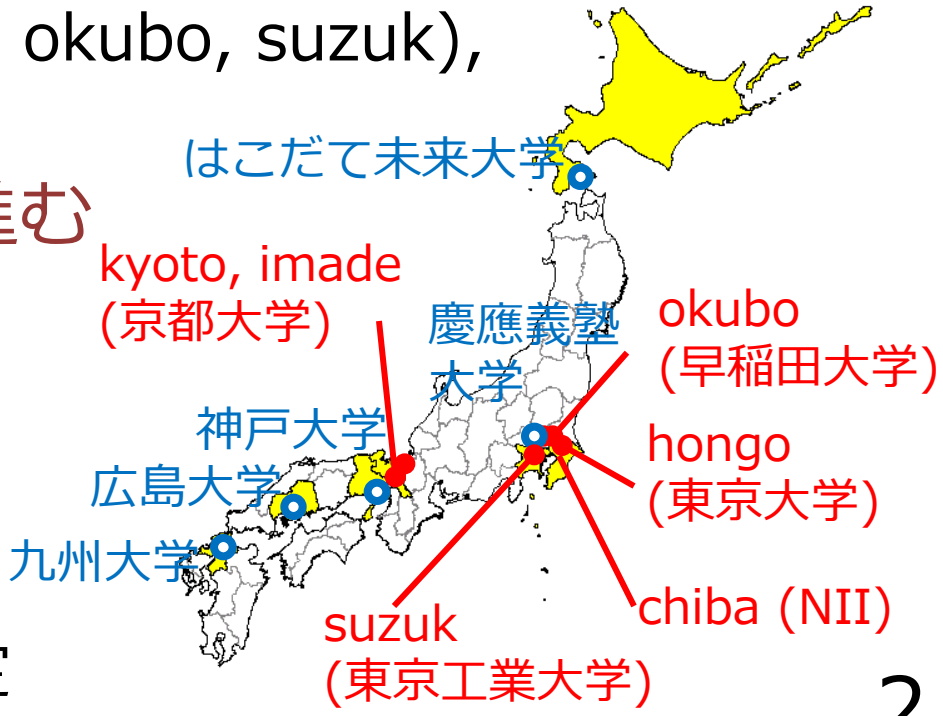
InTriggerの概要・利用ルール

東京大学　近山・田浦研究室　高橋 慧
kay_at_logos.ic.i.u-tokyo.ac.jp

InTrigger環境とは (1)

日本中に設置されたクラスタを統合的に管理

- ▶研究目的に巨大な計算資源を提供
- ▶最終的には20-30拠点、数千台規模を目指す
- ▶6拠点・324台(504プロセッサ)が稼働中
  - ◘千葉(chiba), 東京(hongo, okubo, suzuk), 京都(kyoto, imade)
- ▶今後も引き続き拡張が進む



● 2006年度導入済

○ 2007年度導入予定

InTrigger環境とは (2)

- 各拠点の環境は基本的に同一
  - ▶アカウントは全拠点で共通
  - ▶全ノードへのソフトのインストールが簡単にできる
    - 全拠点についてソフトウェアごとに管理者を設定
    - OSなどのシステムも簡単にアップデート可能
- 自由度の高い資源利用ポリシー
  - ▶様々な利用方法に対応したい
  - ▶特定のシステムに依存しない「利用ルール」を設定

[発表の流れ]

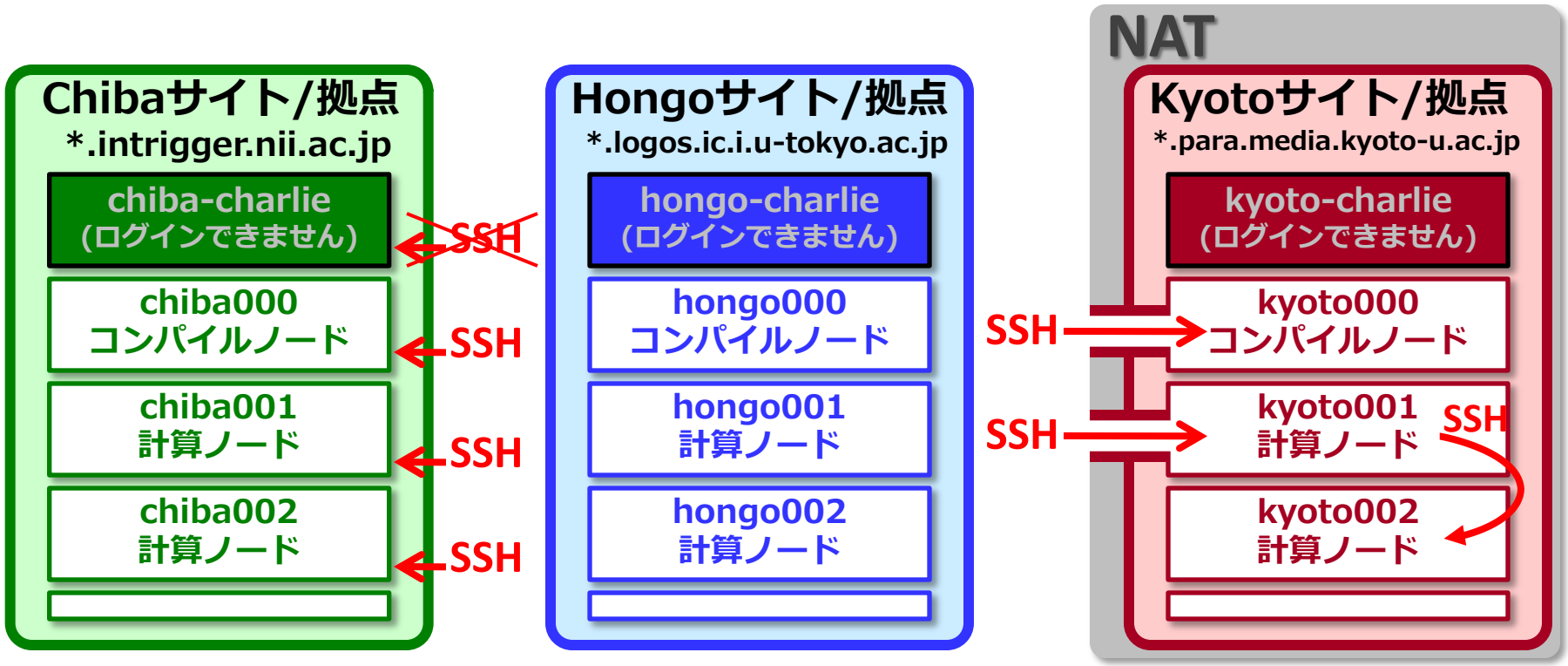
1. InTrigger環境の概要
   - ▶ハードウェア構成
   - ▶情報監視ソフトウェア (Ganglia, VGXP)
2. 資源利用ルールについて
   - ▶CPU資源の利用について
   - ▶ディスク利用について

InTrigger環境の構成

- 多数の拠点(クラスタ)をインターネットで接続
- 一拠点の構成:
  - ▶1台のNFS/NISノード (chiba-charlieなど)
  - ▶1-2台のコンパイルノード (chiba000, chiba100など)
  - ▶多数の計算ノード (chiba001,002…)

SSHでログインするには

- NIS/NFSサーバにはログインできません
- NAT環境では入り口ノードのみ直接ログイン可能
- InTrigger内のノード間ではフィルタリング無し



ハードウェア構成

- 大部分のノードは64bit Dual core
- chiba, hongoでは32bitと64bitが混在
  - ▶chiba000-069は32bit, 100-157は64bit
  - ▶バイナリの互換性が無いので注意
- ローカルHDDを積極的に使って下さい(後述)

Hosts	Address	CPU (32bit)	RAM	HDD (local)	HDD (nfs)
hongo000-hongo069	Global	Pentium M 1.8GHz	1GB	80GB	2TB
hongo100-hongo113		Core2Duo 2.33GHz	4GB	500GB	
chiba000-chiba069	Global	Pentium M 1.8GHz	1GB	80GB	9TB (raid)
chiba100-chiba157		Core2Duo 2.33GHz	4GB	500GB	
okubo000-okubo113	Global	Core2Duo 2.33GHz	4GB	500GB	9TB (raid)
suzuk000-suzuk035	Global	Core2Duo 2.33GHz	4GB	500GB	9TB (raid)
kyoto000-kyoto034	Private (NAT)	Core2Duo 2.33GHz	4GB	500GB	9TB (raid)
imade000-imade029	Private	Core2Duo 2.33GHz	4GB	500GB	9TB (raid)

情報サイトなど

ユーザーポータル

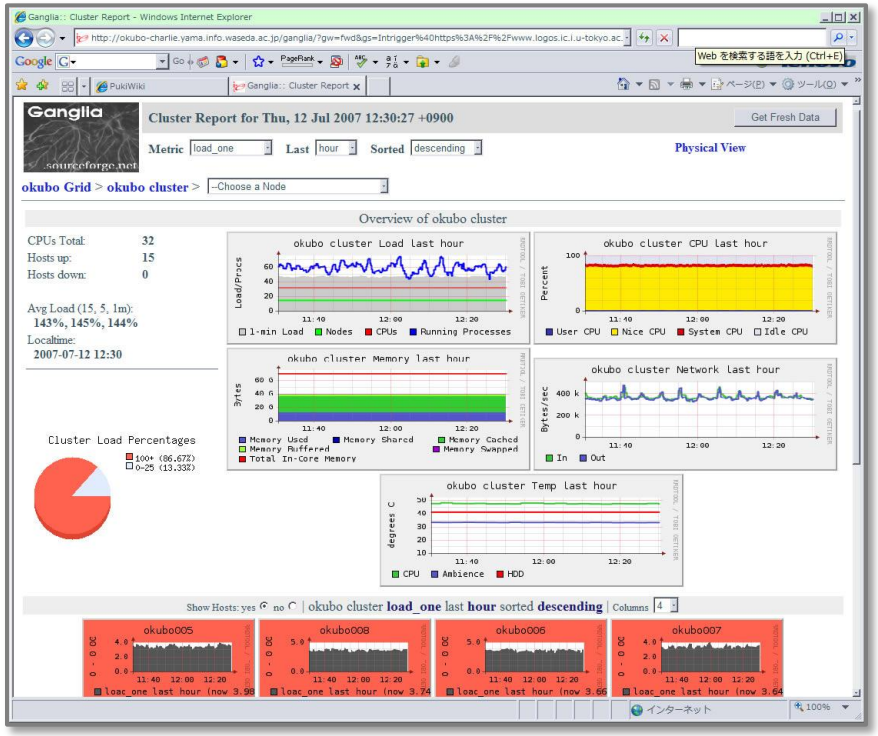
- ノードの障害情報などをご案内します

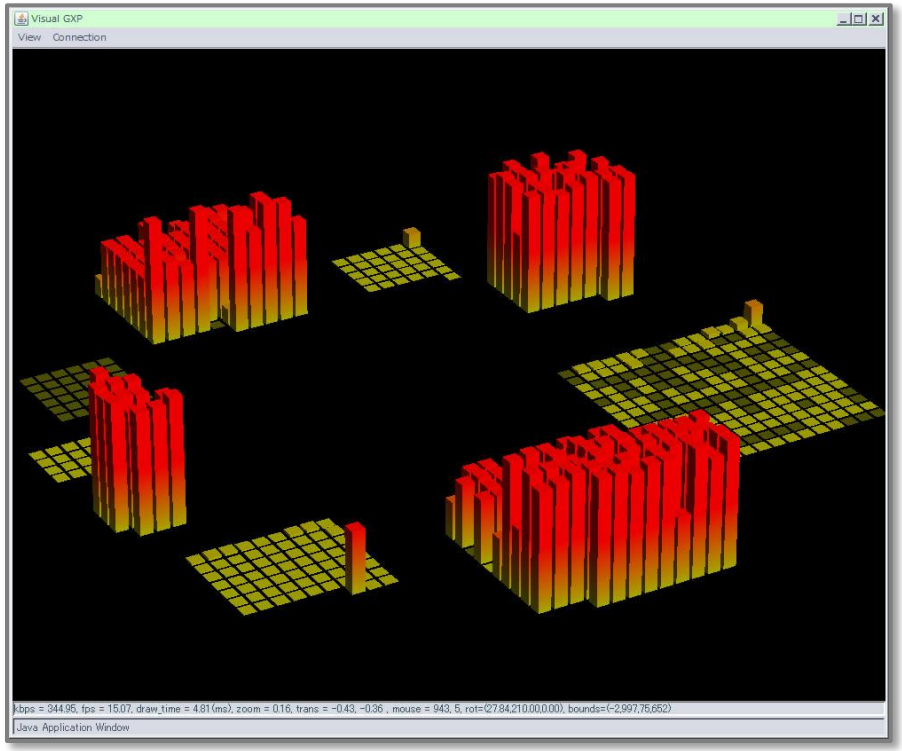
メーリングリスト

- intrigger-users__at__logos.ic.i.u-tokyo.ac.jp
- 重要なお知らせを流します
- 登録メールアドレスの変更は、ポータルの「登録情報編集」からお願いします

InTriggerの状態監視

- Ganglia: 各サイトでの負荷を記録・表示
- VGXP: 3Dで各ノードの負荷をリアルタイムで表示
  - ►どちらもユーザポータルからリンクされています





[発表の流れ]

1. InTrigger環境の概要
   - ▶ハードウェア構成
   - ▶情報監視ソフトウェア (Ganglia, VGXP)
2. 資源利用ルールについて
   - ▶CPU資源の利用について
   - ▶ディスク利用について

InTrigger資源の利用

使いやすい環境を提供したい

- ▶「バッチキュー専用」より柔軟なシステム

様々な用途のユーザが共存

- ▶短時間占有してパフォーマンスを測定するユーザ
- ▶長期に渡って、空きCPUを利用したいユーザ
- ▶対話的に並列プログラムを開発したいユーザ

避けたい状態

ルール無しでコマンド実行可能にすると…

- ▶一部のユーザーが長期に亘って資源を占有

キュー専用にすると…

- ▶デバッグのたびに順番待ち
- ▶普段はがらがら、混雑時は1時間単位で実行待ち
- ▶新しいMiddlewareの開発・テストが出来ない

これらをふまえて、
以下の利用方法・ルールを設定した

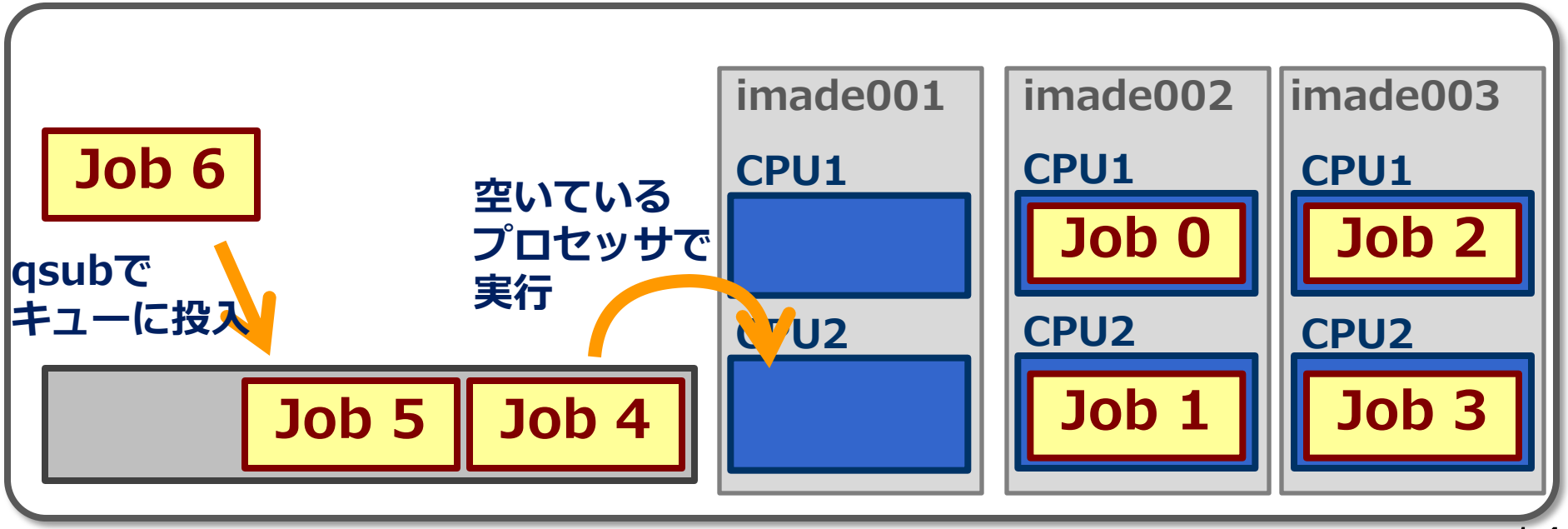
共存する仕組み

- プロセス起動の手続きは限定しない
  - ►バッチキュー・対話式シェル・独自のミドルウェアなど
- ルールを設定する
  - ►キュー経由のプロセスに優先権を与える
  - ►優先権のあるプロセスの邪魔をしてはいけない
  - ►優先権を濫用しない・並列度の高い実行を推奨
  - ►コンパイルノードは除外
- 各ユーザはルールを守るよう努める
  - ►(よろしくおねがいします)
- ルール違反を検出し、週報とし公開する(準備中)
  - ►アカウントごとに「ポイント」として記録する

キューを通したプロセス起動

Torque(キュー)をインストールしました

- ►$ qsub task.sh のようにジョブを投入
- ►順番が来たら、クラスタ内のどこかのノードで排他的に実行される
- ►一般のプロセスに対し優先権を持つ (占有できる)



ルール1: キュー優先

- キュー以外から起動されたプロセスは、キュー経由のプロセスの邪魔をしてはいけない
  - ►キュー以外のプロセスが利用できるCPU使用率を、(CPU数 – キュー経由のプロセス数)*100%
    と定義する
  - ►キュー経由以外のプロセスの使用率が100%を越えたら、100%を越えないようCPU使用率を減らす

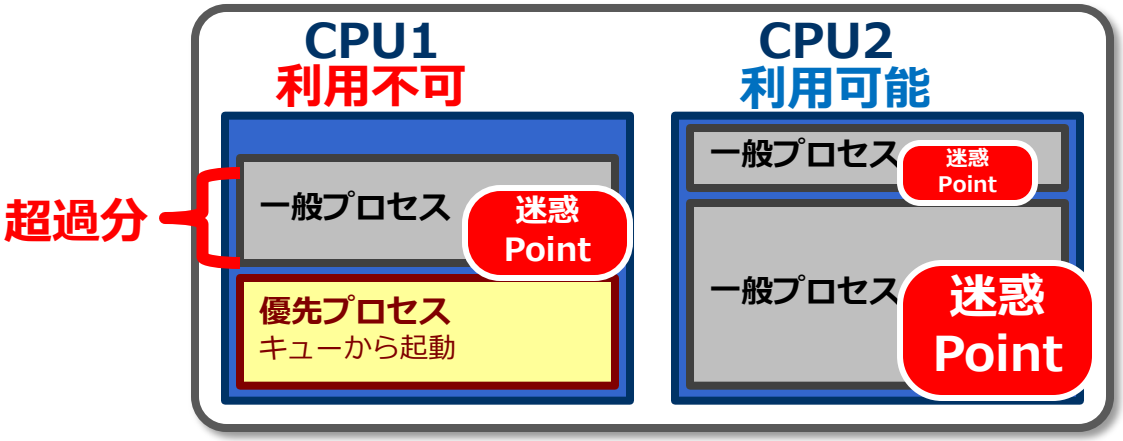
ルール2: キューを占有しない

- キュー (Torque)を長時間占有しない
  - ►時間に制限を設ける
- 並列実行を推奨する
  - ►[sqrt(使用プロセッサ数) * 使用時間]
    を基準に用いる
- 全プロセッサを1時間/週 占有できる値を設定
  - ►400プロセッサ →1時間
  - ►100プロセッサ →2時間
  - ►4プロセッサ　　→10時間

(キュー以外のプロセスはこの制限を受けない)

ルールの運用について

- アカウントごとにルール違反をポイント化
  - ►ルール1に関して: 利用可能分を超過したCPU使用率を、その時点で使っている一般プロセスで比例配分
- ユーザポータルで週報を公開 (システム準備中)
- 週報の公開が始まったら、各自で定期的にチェックするようお願いします。



超過分をその時点での
CPU利用率に応じて
比例配分し、迷惑
ポイントとする

ルール違反をしないために…

短時間でパフォーマンス測定など

- ►利用時間を各自で把握してキュー経由で実行
- ►キューの残り利用時間今後コマンドを提供する

開発・デバッグなど

- ►TopやGanglia, VGXPで監視して、対話的に実行

長期間実行するプロセス

- ►キューやプロセスの状態を監視しつつ実行する仕組みを自作する
- ►Nicerを用いる

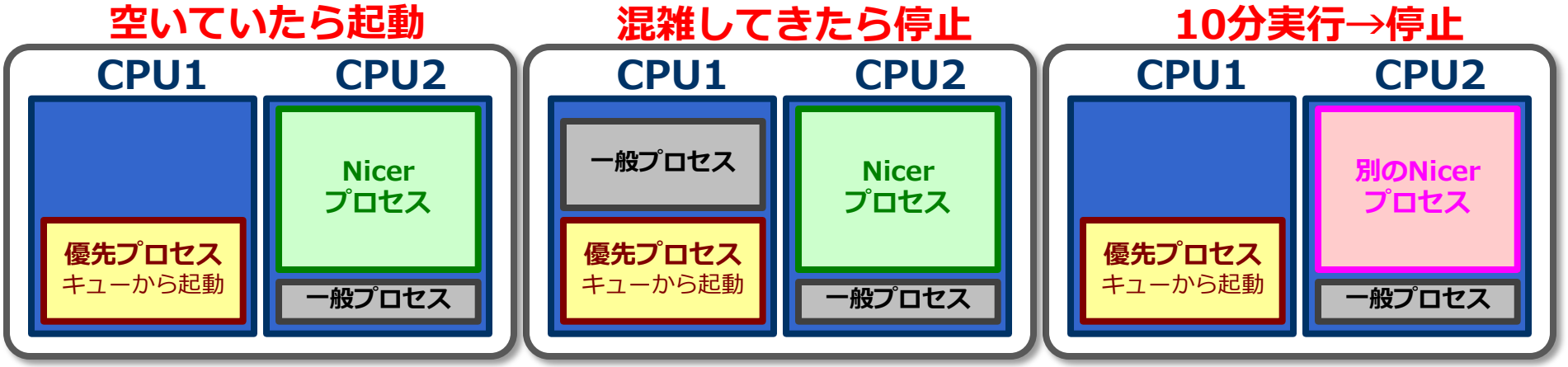
Nicerとは? 今から説明します

Nicerの概要

- ルールを常に守って資源を使うためのツール
  - ▶実行方法: $ nicer ./a.out
  - ▶InTriggerの各ノードにインストールされます (準備中)
- 動作:
  - ▶nicerを通じて実行されるプロセス(ジョブ)を制御
    - シグナルを送って、ジョブプロセスを開始/停止する
    - 一般プロセスの合計CPU使用率が上限を超えたら、ジョブプロセスをsleepさせる
    - 一般プロセスの合計CPU使用率が十分低くなったら、ジョブプロセスを再開させる
  - ▶定期的にジョブプロセスをsleepさせ、自律的に他のnicerプロセスとの公平性を保つ

Nicerの詳しい動作

- 常に約1分間隔で使用率をチェック
- ジョブプロセス停止時:
  - ▶利用可能な空きCPUが90%以上で起動
- ジョブプロセス実行時:
  - ▶一般プロセスのCPU使用率の合計が、使用可能なCPU使用率を越えたら停止
  - ▶10分以上走ったら一度停止させる



システム整備の予定

- ルール自体は既に適用されています
- 監視システム・週報
  - ►準備中です
- Nicer
  - ►GXPに古いバージョンが入っています
    (少し動作が異なります)
  - ►今後、今回の発表に即したものにアップデートします
  - ►InTriggerの各ノードにもインストールされる予定です

使えるようになり次第、メールでご連絡します

ディスクの利用について

/home/$USER: NFS (raid)

- ▶永続的な保存場所 (バックアップは取っていません)
- ▶全員で2TB ・Quota無し

/data/1, /data/2, /data/3: NFS

- ▶大きなデータの一時的な置き場所
- ▶各自で/data/1/$USER のような
サブディレクトリを作り、自由に用いる

プログラムの置き場所
データの置き場所

/data/local: 計算ノードのローカルディスク

- ▶プログラムが読み書きするデータ
- ▶/data/local/$USERのような
サブディレクトリを作り、自由に用いる

プログラムの作業場所
(適宜削除して下さい)

NFSを作業ディレクトリにすると…

- 実験: 10台のノード(hongo100-110)が1GBのファイルを同時に書き込み
  - ▶1台の場合: 300-600Mbps * 1
  - ▶10台の場合:30Mbps * 10
- 性能が出ないだけでなく、他のユーザにも影響
  - ▶NFSサーバのload averageが常時10以上
  - ▶ホームでlsすると結果が返ってこない
  - ▶他のユーザがsshログインするのに10秒以上かかる (/home/$USER/.ssh/authorized_keysを読めないため)

ローカルディスクの活用

- NFS上のデータを多数のノードが読み書きすると、クラスタ全体のパフォーマンスが低下
  - ▶アクセスが遅く、プログラムの性能が出ない
  - ▶その他のユーザも、ディスクアクセス速度の低下、ログイン出来ないなどの影響を被る
- データを頻繁に読み書きする場合は、ローカルディスク (/data/local)を使って下さい
  - ▶データコピーには、GXPのBCPなどを用いる

終わりに

1. InTrigger環境の概要を説明した
   - ▶ユーザポータルをチェックしてください
2. 利用ルールを説明した
   - ▶ルール1:キュー経由のプロセスを邪魔しない
   - ▶ルール2:キュー経由のプロセスを乱発しない
   - ▶各自で気をつけるか、Nicerを用いる
   - ▶大きなデータを読み書きする場合は、NFSではなくローカルディスクを用いる
3. 今後週報の作成などシステム面の整備を進める

☆ご協力よろしくお願いします☆