PACS-CSの利用法

朴　泰祐
計算科学研究センター／システム情報工学研究科
taisuke@cs.tsukuba.ac.jp
http://www.hpcs.cs.tsukuba.ac.jp/~taisuke/

内容

- PACS-CSシステムの構成
- PACS-CSシステムと周辺環境
- ジョブ投入と入出力の概念
- ジョブ実行の様子
- 昨年度利用状況
- まとめ

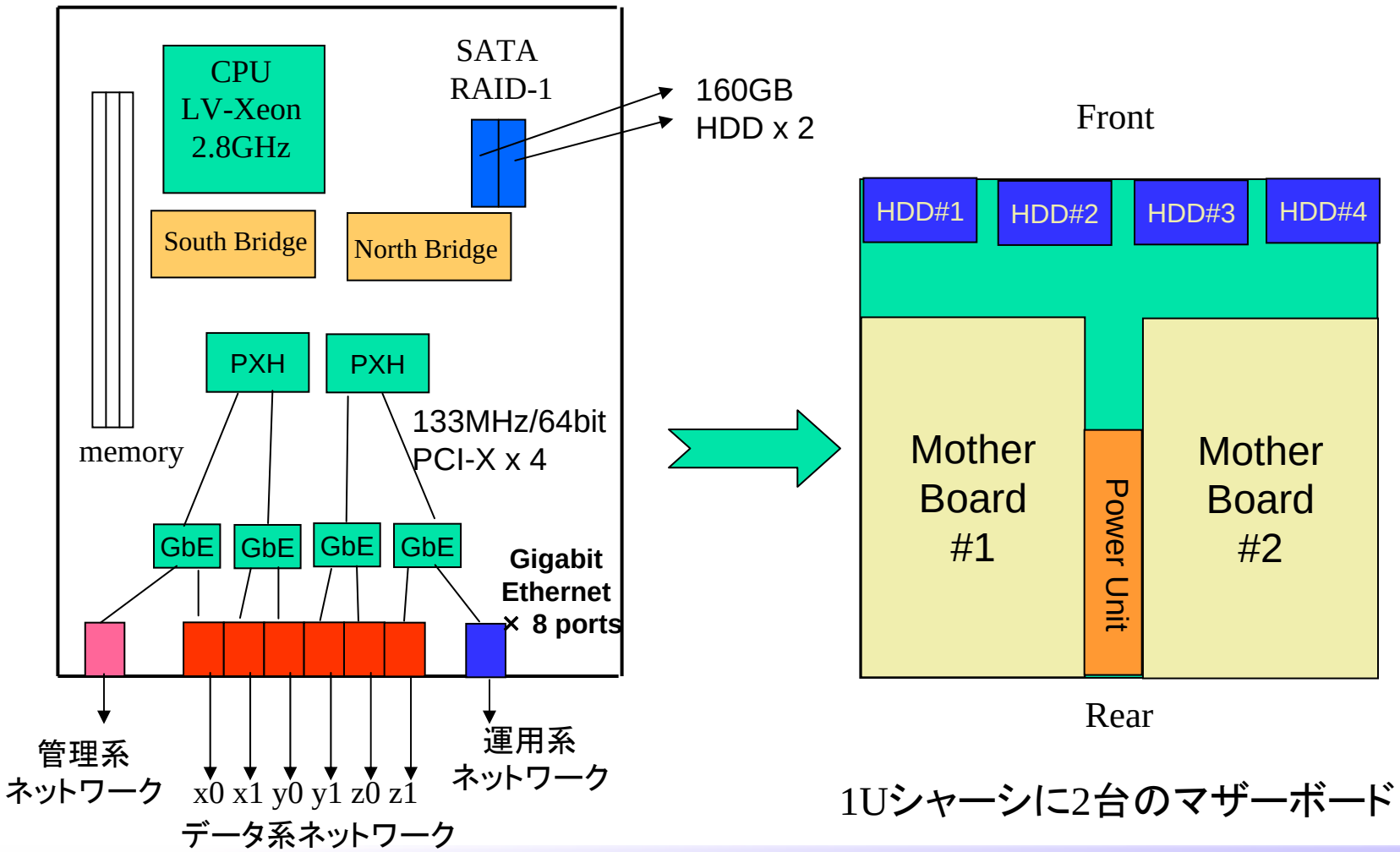
PACS-CSの特徴

- 科学技術計算向け超並列クラスタ計算機
  - メモリとネットワークのバンド幅を重視
  - single-core / node
    ⇒ 1つのCPUコアがノードのメモリとネットワーク性能を占有
  - 3次元ハイパクロスバ網による3次元直交ネットワーク
    ⇒ 物理モデルの直接マッピングに最適
  - 一般のPCクラスタと同程度の実装密度
  - Gigabit Ethernet を多用したマルチリンク・多次元通信
- 2560ノードの超並列システム
  - 従来のMPPをコモディティベースで開発
  - 大規模計算科学シミュレーションがターゲット
  - ジョブ当たり128～2048ノードを利用
  - 数TFLOPS/ジョブ程度の高性能志向運用
  - 比較的長時間のジョブ実行：最大48時間／ジョブ

PACS-CSシステム諸元

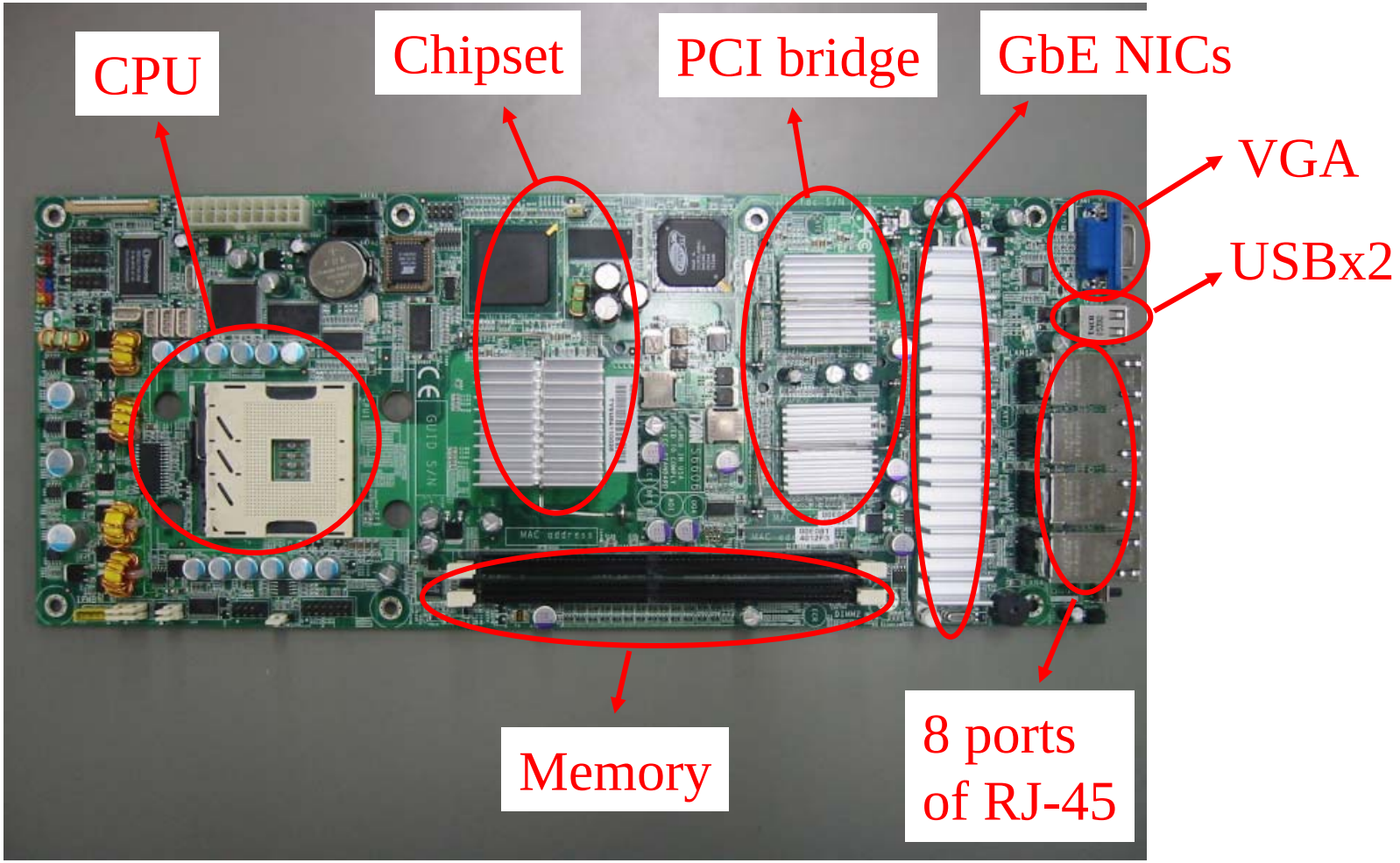
ノード台数	2560 (16 x 16 x 10)
理論ピーク性能	14.3 Tflops
ノード構成	単一CPU / ノード
CPU	Intel LV Xeon EM64T, 2.8GHz, 1MB L2 cache (5.6 GFLOPS peak)
メモリ容量・バンド幅	2GB/CPU　6.4GB/sec/CPU
並列処理ネットワーク	3次元ハイパクロスバ網
リンクバンド幅	単方向 250MB/s/次元 単方向 750MB/s （3次元同時転送時）
ローカルHDD	160 GB/ノード（RAID-1）
総システムサイズ	59ラック
総消費電力（推定）	545 kW

専用マザーボード開発

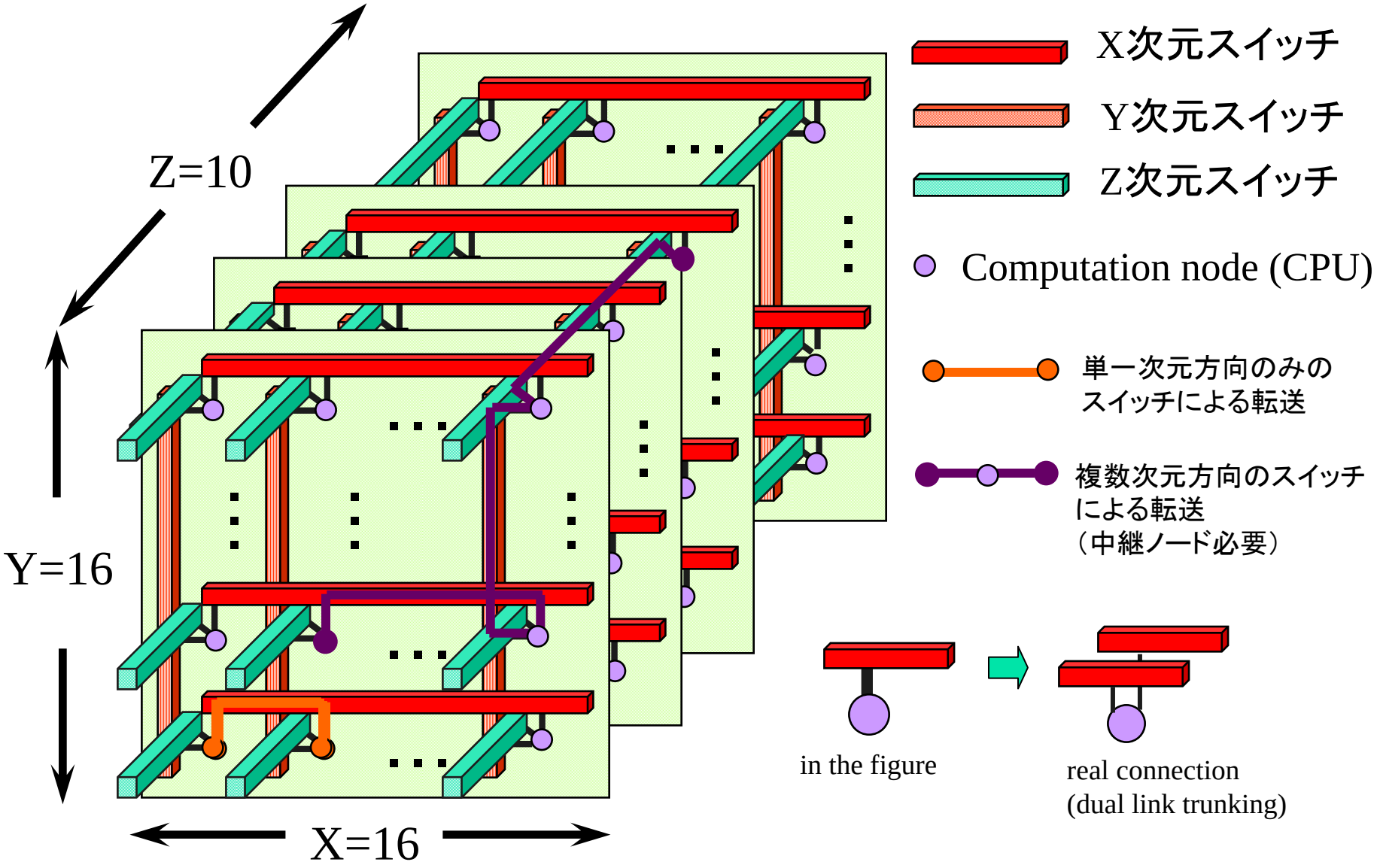


1Uシャーシに2台のマザーボード

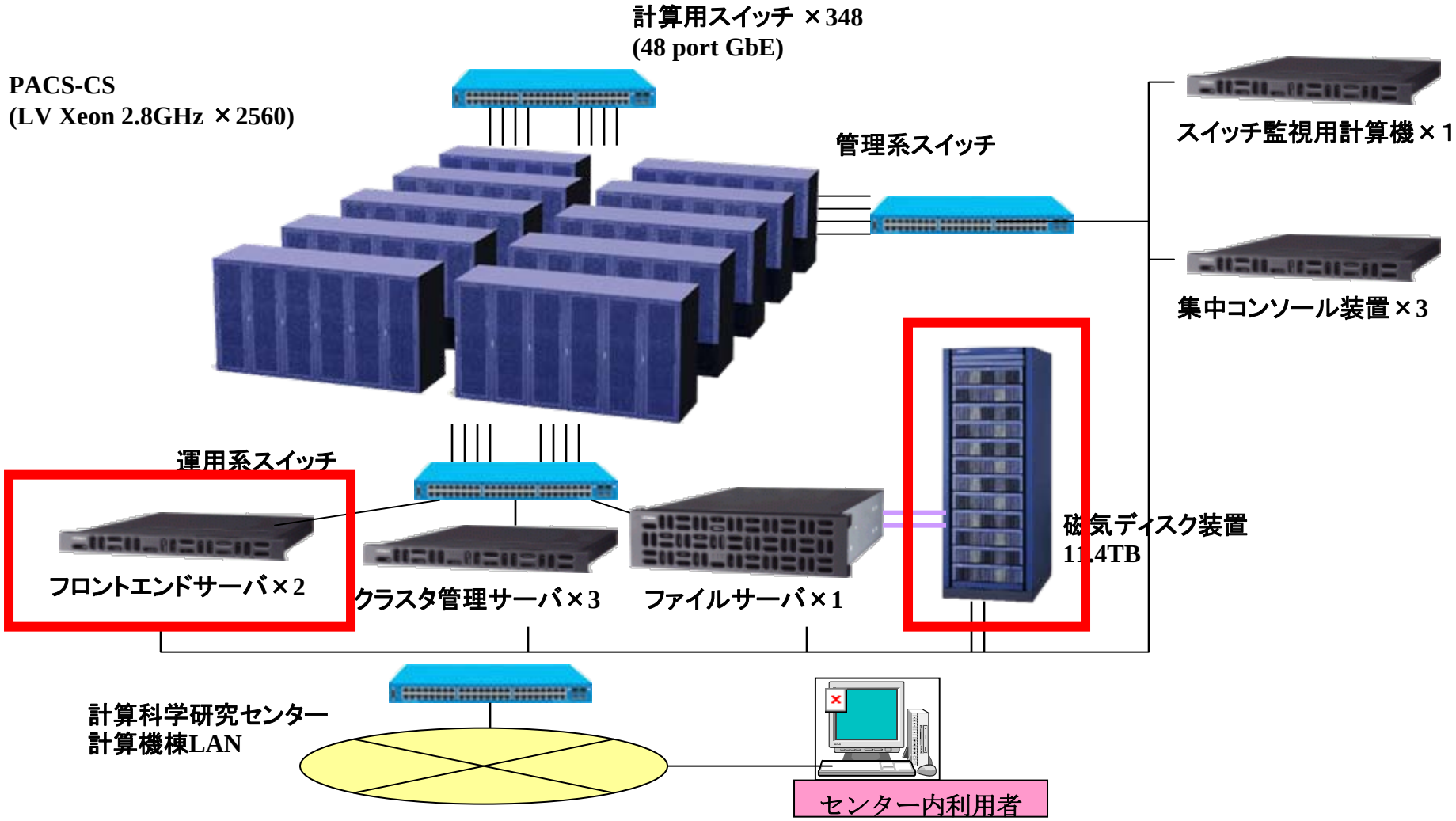
マザーボード写真


CPU
Chipset
PCI bridge
GbE NICs
VGA
USBx2
Memory
8 ports
of RJ-45

3次元HXBネットワーク(16x16x10=2560 node)



PACS-CS全体システム構成



フロントエンドサーバ

- センター外（学外）からのアクセスポイント
  - ホスト名：ccshfr1.ccs.tsukuba.ac.jp
    （代替ホスト：ccshfr2.ccs.tsukuba.ac.jp）
  - sshでのみログイン可能
- 基本的に全てのユーザ処理はこの上で
  - コンパイル
  - ファイルサーバ上のファイル編集等
  - ジョブ投入、モニタリング
  - scp等による外部とのファイルのやり取り

ファイルサーバ

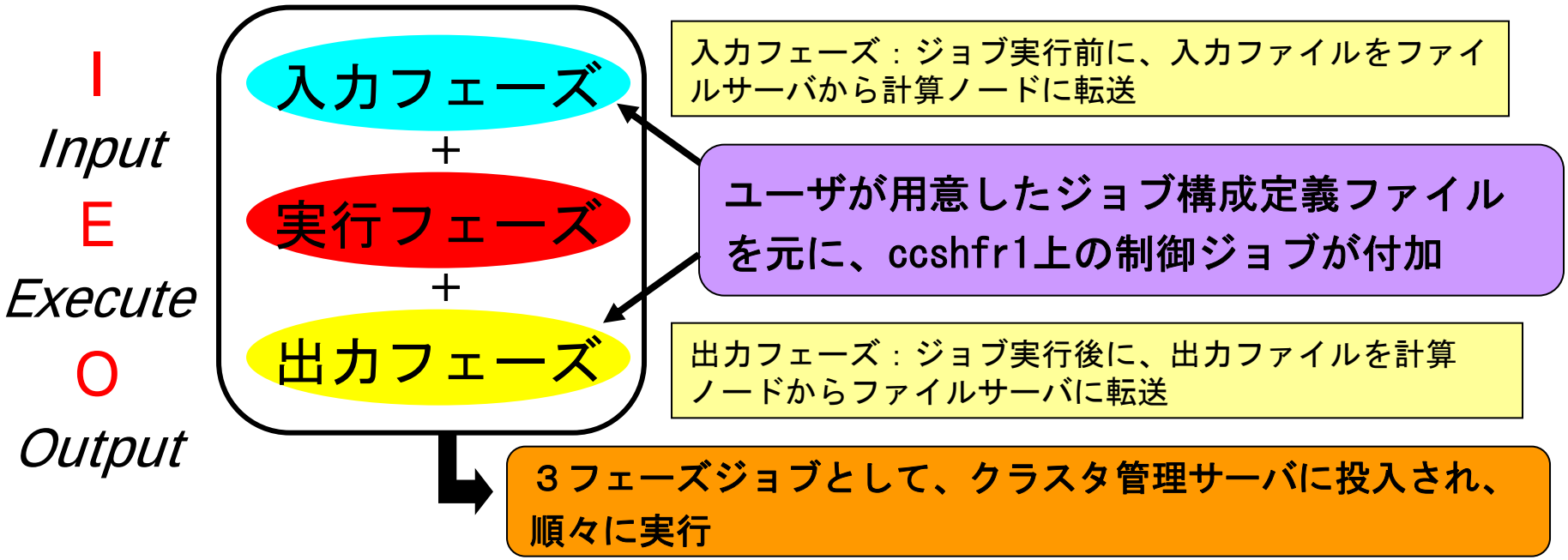
- 約110TBの大容量ファイルサーバ
  - フロントエンドサーバからアクセス可能
  - PACS-CSの代表ノードからアクセス可能
  - PACS-CSの代表ノード以外のノードからは、「入出力フェーズ」（後述）において間接的にアクセス可能
- 共同利用においては各プロジェクト単位でグループディレクトリを提供（容量制限あり）
  - /work1/*group*/*user*, /work2/*group*/*user*, /work3/*group*/*user*

ジョブの実行とファイルアクセス

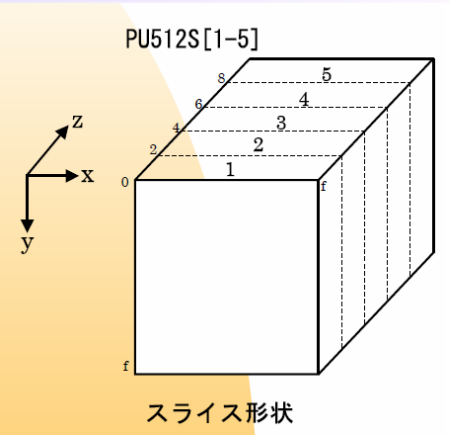
- PACS-CSジョブは原則としてMPIによる並列プログラム
  - MPICH及びYAMPIを提供
  - 通常のMPIプログラムと互換

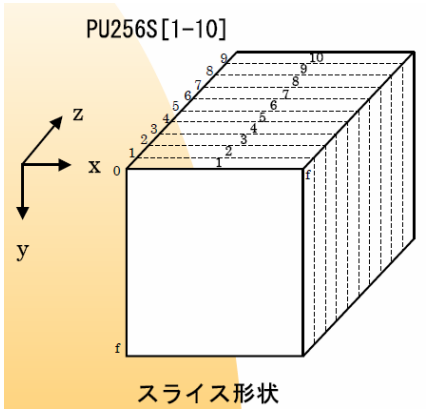
- PACS-CSにおけるファイル処理の原則
  - ファイルサーバを直接参照できるのは「代表ノード」のみ（64ノードに1台）
  - 原則として、ファイルサーバ上のファイルを各計算ノードのローカルディスク（/work 領域）にコピーして参照
    ローカルディスク＝160GB (RAID-1)
  - 計算終了後、各計算ノードのローカルディスク上のファイルをファイルサーバに転送
  - ジョブ終了後は各計算ノードのローカルディスク上にはファイルを残さない

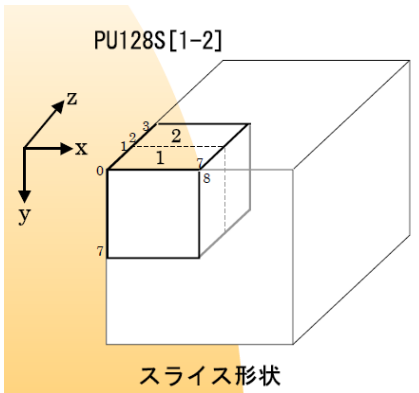
3フェーズ実行

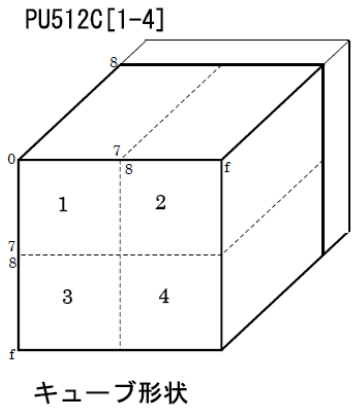

I
Input
E
Execute
O
Output
入力フェーズ
+
実行フェーズ
+
出力フェーズ
入力フェーズ：ジョブ実行前に、入力ファイルをファイルサーバから計算ノードに転送
ユーザが用意したジョブ構成定義ファイルを元に、ccshfr1上の制御ジョブが付加
出力フェーズ：ジョブ実行後に、出力ファイルを計算ノードからファイルサーバに転送
３フェーズジョブとして、クラスタ管理サーバに投入され、順々に実行

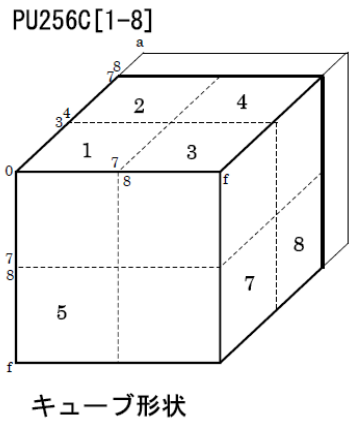
パーティション構成

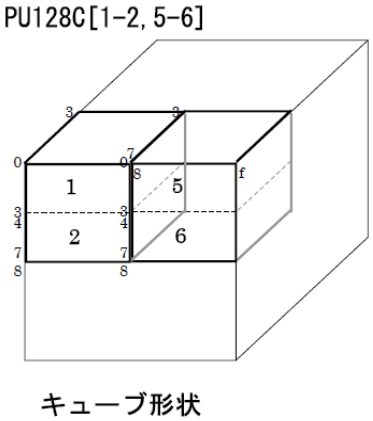












ユーザの状況に応じ組み合わせを変えて運用

パーティションサイズとキュー構成

キュー	パーティション	run limit	ノード座標	Memory limit(GB)	cpu limit	elaps limit
q2560[sa]S1	PU2560S1	1	(0-15, 0-15, 0-9)	3584	s:15min a:24hrs	s:30min a:48hrs
q2048[sa]S1	PU2048S1	1	(0-15, 0-15, 0-7)	2867		
q1024[sa]S1	PU1024S1	1	(0-15, 0-15, 0-3)	1434		
q1024[sa]S2	PU1024S2	1	(0-15, 0-15, 4-7)	1434		
q512[sa]S[1-2]	PU512S[1-2]	1	(0-15, 0-15, 0-1) (0-15, 0-15, 2-3)	717		
q512[sa]S[3-4]	PU512S[3-4]	1	(0-15, 0-15, 4-5) (0-15, 0-15, 6-7)	717		
q512[sa]S5	PU512S5	1	(0-15, 0-15, 8-9)	717		
q512[sa]C[1-2]	PU512C[1-2]	1	(0-7, 0-7, 0-7) (8-15, 0-7, 0-7)	717		
q512[sa]C[3-4]	PU512C[3-4]	1	(0-7, 8-15, 0-7) (8-15, 8-15, 0-7)	717		

パーティションサイズとキュー構成（続き）

キュー	パーティション	run limit	ノード座標	Memory limit(GB)	cpu limit	elaps limit
q256[sa]S[1-4]	PU256S[1-4]	1	(0-15, 0-15, 0) … (0-15, 0-15, 3)	358	s:15min a:24hrs	s:30min a:48hrs
q256[sa]S[5-8]	PU256S[5-8]	1	(0-15, 0-15, 4) … (0-15, 0-15, 7)	358		
q256[sa]S[9-10]	PU256S[9-10]	1	(0-15, 0-15, 8)　(0-15, 0-15, 9)	358		
q256[sa]C[1-2]	PU256C[1-2]	1	(0-7, 0-7, 0-3) … (0-7, 0-7, 4-8)	358		
q256[sa]C[3-4]	PU256C[3-4]	1	(8-15, 0-7, 0-3) … (8-15, 0-7, 4-8)	358		
q256[sa]C[5-6]	PU256C[5-6]	1	(0-7, 8-15, 0-3) … (0-7, 8-15, 4-8)	358		
q256[sa]C[7-8]	PU256C[7-8]	1	(8-15, 8-15, 0-3) … (8-15, 8-15, 4-7)	358		
q128[sa]C[1-2]	PU128C[1-2]	1	(0-7, 0-3, 0-3) … (0-7, 4-7, 0-3)	179		
q128[sa]S[1-2]	PU128S[1-2]	1	(0-7, 0-7, 0-1) … (0-7, 0-7, 2-3)	179		
q128[sa]C[5-6]	PU128S[5-6]	1	(8-f, 0-7, 0-1) … (8-f, 0-7, 2-3)	179		
q64[sa]C[1-2]	PU64C[1-2]	1	(0-3, 0-3, 0-3) …　(4-7, 0-3, 0-3)	89		
q64[sa]C[3-4]	PU64C[3-4]	1	(0-3, 4-7, 0-3) …　(4-7, 4-7, 0-3)	89		
q64[sa]S[37-38]	PU64S[37-38]	1	(0-7, 0-7, 9) … (8-15, 0-7, 9)	89		
q64[sa]S[39-40]	PU64S[39-40]	1	(0-7, 8-15, 9) … (8-15, 8-15, 9)	89		
q8[sa]	D8C1	1	miniPACS-CS　(0-1, 0-1, 0-1)	11		

ジョブ構成定義ファイル

ジョブ構成定義ファイル例

```
* : type = mkdir -p %s
/work/HITACHI/hitachi2/sample

0 : type = put
include = inputlist

job_step_1: type = job
  class = q512sS3
  job_name = sample.jcl

0 : type = get
include = outputlist
```

PU512S3のノード全体に、ジョブ実行用のディレクトリを作成

mpiランク０対象ノードに、“inputlist”の指定に従いファイル転送

PU512S3で、“sample.jcl”を実行

mpiランク０対象ノードから、“outputlist”の指定に従いファイル転送

入力(出力)ファイルリスト

入力ファイルリストの例

```
/hfs-work1/COMP/taisuke/sample/input.0 /work/COMP/taisuke/sample/input.0
/hfs-work1/COMP/taisuke/sample/input.1 /work/COMP/taisuke/sample/input.1
/hfs-work1/COMP/taisuke/sample/dat.$mpirank   /work/COMP/taisuke/sample/d
```

“$mpirank” の指定がある場合は、MPI上のランク番号に対応するノードにのみ、そのファイルが転送される

ファイルサーバ上のワークディレクトリ以下にある入力ファイルを、計算ノードのワークディレクトリ以下に転送する。

ジョブ実行スクリプト

ジョブ実行スクリプトファイルの例

```
#! /bin/csh
set pe="512"
set PROG="./a.out"
cd $PBS_O_WORKDIR
scout -wait -g $NODE_GROUP -e scrun -nodes=${pe}x1,checkpoint $PROG
```

ジョブ実行ディレクトリにcdする。変更しないこと。

$NODE_GROUPは、制御ジョブにより定義される。

ジョブ実行、モニタリング、打ち切り

```
$ cpsub -f sample.cfg
Your job will be assigned [PU512S3]
put_0002 file list check [OK]
3302.ccshfr1


$ cpstat -a
ccshfr1:                                                    Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID NDS TSK Memory Time  S Time
--------------- -------- -------- ---------- ------ --- --- ------ --  -----
3302.ccshfr1    hitachi2 batch    sample.jcl  16783  --  --    --    --  R 00:00 [ q512sS3]
ccshcms-op2:                                                Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID
--------------- -------- -------- ---------- ------
3303.ccshfr1    hitachi2 q512sS3  IE03302


$ cpdel -k 3302.ccshfr1
```

現在キューに積まれているジョブと実行中のジョブを表示

投入済み（実行中）ジョブをキャンセル

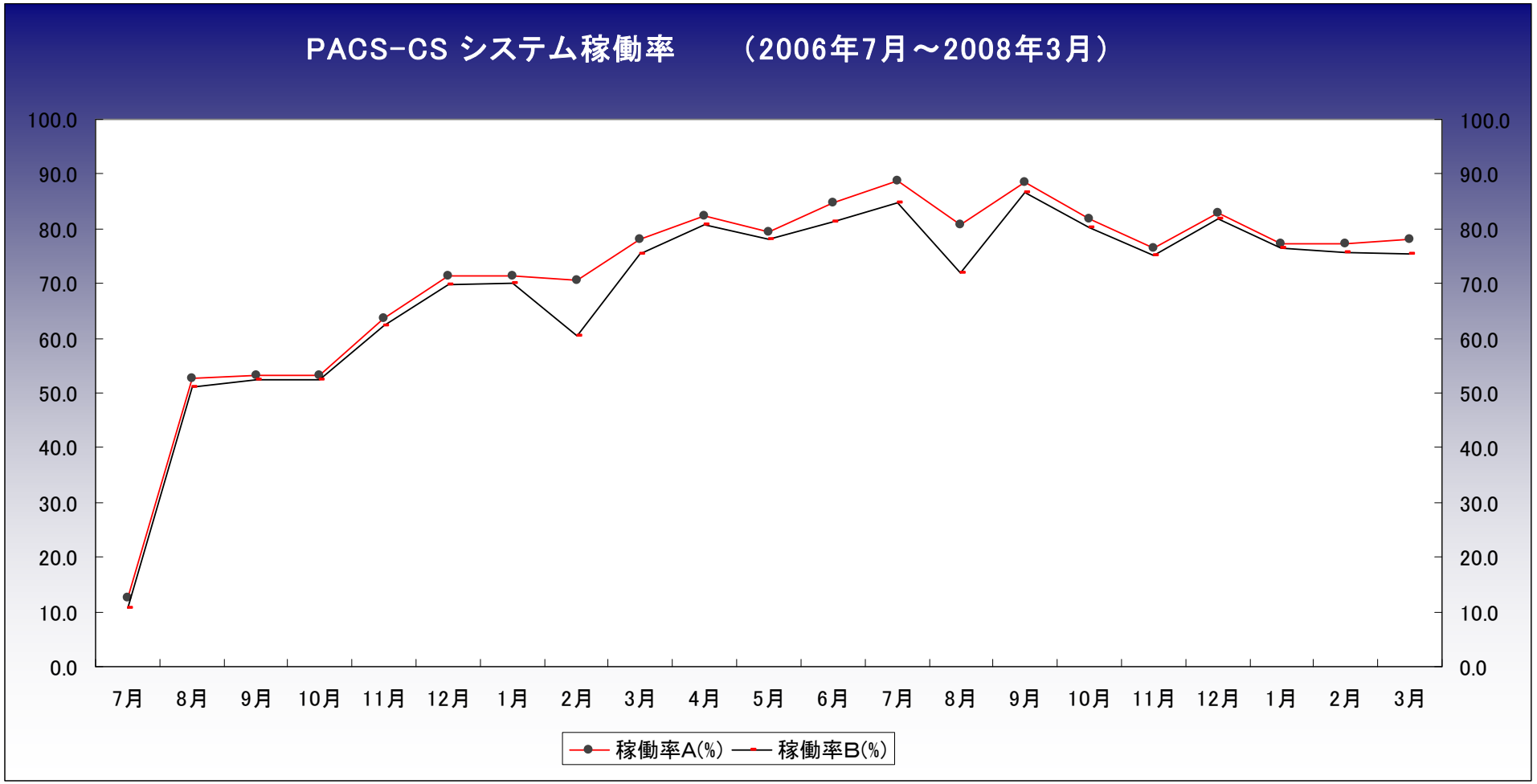
数値計算ライブラリ

- 数値演算ライブラリとして、Intel Cluster MKLが利用可能
  - Linux 対応の SCALAPACK (スケーラブル LAPACK)
  - 線形代数(BLAS, LAPACK, DSS)
  - 離散フーリエ変換(DFT)
  - PARDISO直接法スパースソルバ
  - ベクトル・マス・ライブラリ(VML)
  - ベクトル・スタティスティカル・ライブラリ(VSL)乱数ジェネレータ

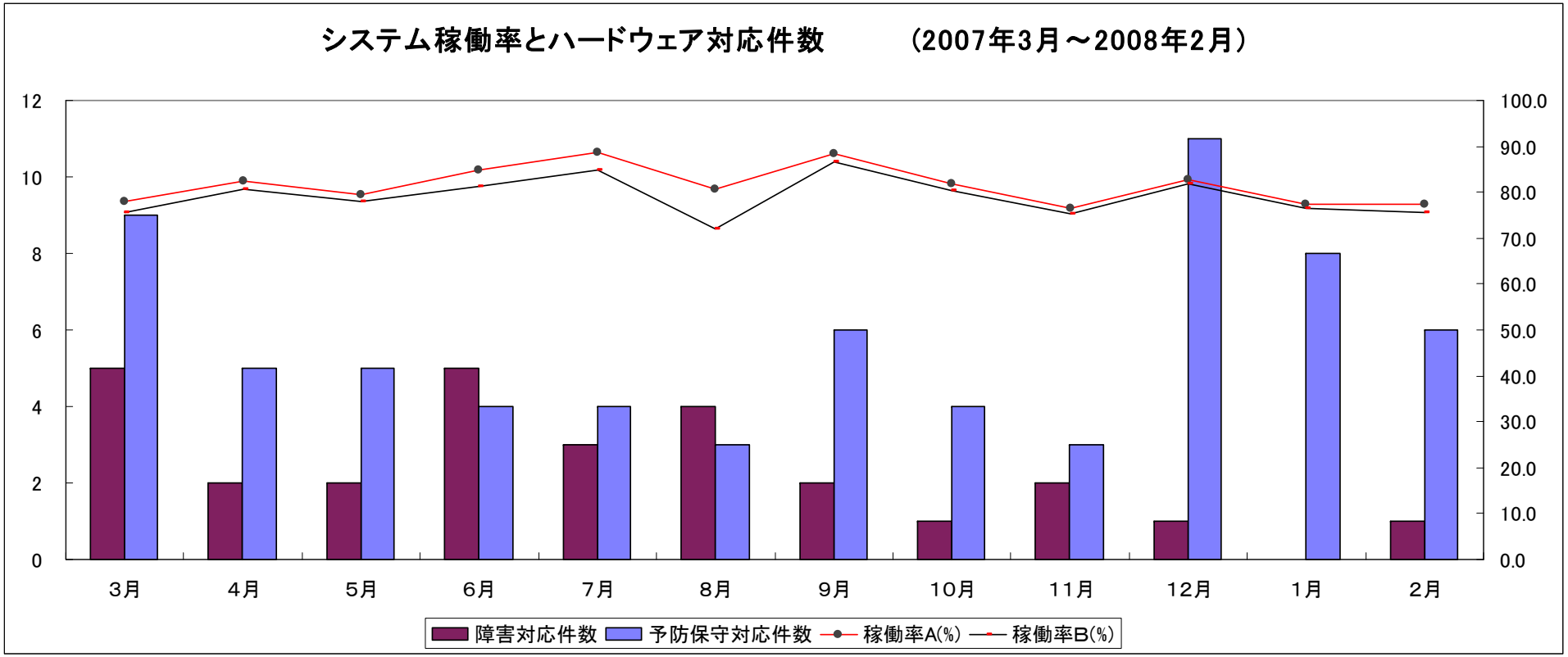
システム運用と保守

- 学際共同利用プログラムを半年単位で実施
  - 各プロジェクトグループにパーティション（キュー）を割り当て
  - プロジェクトの進行状況と budget（割り当てCPU時間）に応じて戦略的に運用する場合がある
    - 大規模パーティション運用
    - 特に特定時期に計算を加速する必要がある場合
- 共用キュー上のジョブに対し「グループ間公平スケジューリング」が行われる
- 毎月１回のシステム定期保守
  - 原則として毎月最終月曜日
  - システム予防保守による安定稼動

運用開始からのシステム稼動状況



H19年度の稼働率と故障状況



まとめ

- PACS-CSはバンド幅重視の超並列クラスタ
- 大規模計算科学シミュレーションが対象
  「PACS-CSだからこそ実行可能なジョブ」を！
- 大容量ファイルサーバを持つが全ノードから自由に参照できるわけではないため注意が必要
- ジョブスクリプトをベースとしたバッチシステム
- 各プロジェクトに対するキュー割り当て（共用キューもあり）を行い、場合によって戦略的にスケジュール