

スパコンの概要と CMCのスパコンの紹介

大阪大学サイバーメディアセンター 講師

木戸 善之

2018/9/13

目次

1. スパコンの略歴
2. 計算機の概要
3. 並列計算
4. CMCのスパコン

計算機ってなんだ？

- 計算機
 - 計算に用いる機械（デジタル大辞泉）
 - 計算のための機械、器具のこと。コンピュータや電卓を指すことが多い（Wikipedia）
 - 人が不得意な、正確な演算やルーチンワークを肩代わりするための道具



計算機にも様々な種類が

- パーソナルコンピュータ
 - 主に個人で使用するために作られたコンピューター。パソコン、PC
- 汎用機(メインフレーム)
 - 企業の基幹業務に利用される大規模なコンピューター
- スーパーコンピュータ
 - 高度な数値計算(量子物理、流体解析、ケモ・バイオインフォマティクス、天文学...etc)のためのコンピューター
- 数値だけでなく画像、文書など様々な入力に対し処理を行い出力する装置



計算機は気安く触れられるものではなかった

- 世界初の計算機

- Atanasoff-Berry Computer (ABC), 1937年, Iowa, USA
- ENIAC, 1946年, Pennsylvania, USA
- Zuse Z3, 1941年, Germany

- 世界初の汎用機

- UNIVAC I, 1950年, USA

1939年 – 1945年
第二次世界大戦！

アポロ11号月面着陸, 1969年

木戸の誕生日, 1975年

Grid Computing,
2003年, I. Foster

1930

1940

1950

1960

1970

1980

1990

2000

2010

CMCの前身、大型
計算機センター発
足, 1962年

TCP/IP採用, 1983年
, USA

WWW/HTTP, 1990年
, CERN, EU

Webサービス, 2000年
, W3C



弾道計算のシミュレーション

真空中の放物運動

- ただの放物運動

– 初期角度: θ_0 初期速度: v_0 重力加速度: g
時間: t

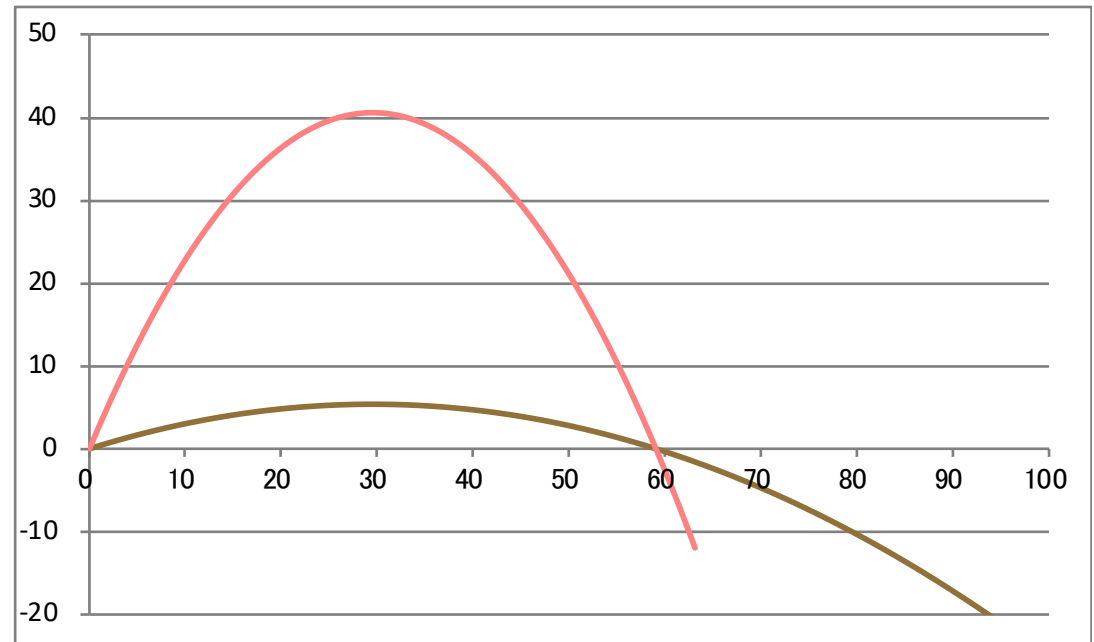
t 秒後の速度

$$v_x = v_0 \cos \theta_0$$
$$v_y = v_0 \sin \theta_0 - gt$$



t 秒後の座標

$$x = v_0 \cos \theta_0 t$$
$$y = v_0 \sin \theta_0 t - \frac{1}{2} gt^2$$



弾道計算のシミュレーション 空気抵抗を入れてみよう

- 速度に比例する空気抵抗を持つ放物運動
 - 初期角度: θ_0 初期速度: v_0 重力加速度: g 時間: t
 - 空気抵抗: k 物体の質量: m

t 秒後の速度

$$v_x = v_0 e^{-\frac{k}{m}t} \cos \theta_0$$

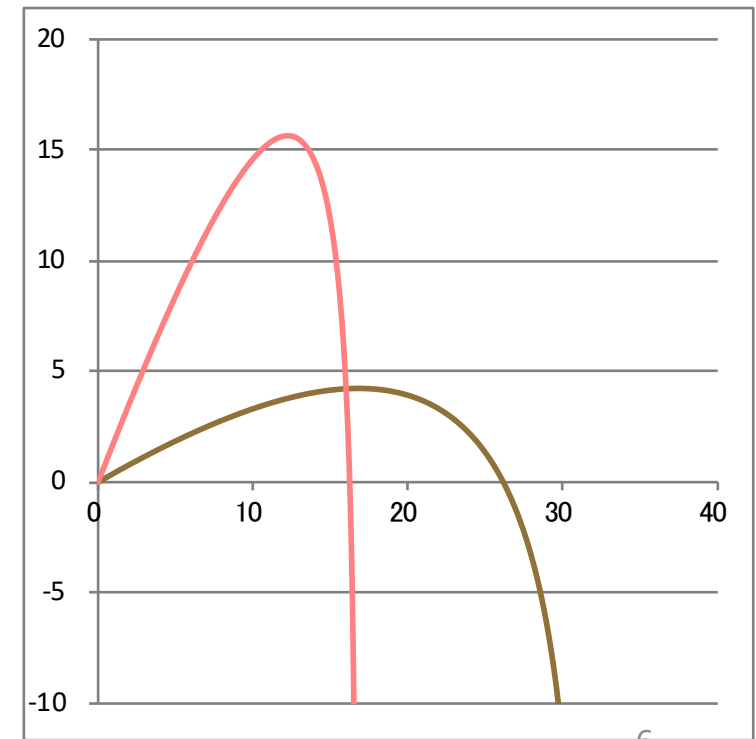
$$v_y = (v_0 \sin \theta_0 + \frac{m}{k} g) e^{-\frac{k}{m}t} - \frac{m}{k} g$$



$$x = \frac{mv_0}{k} \cos \theta_0 (1 - e^{-\frac{k}{m}t})$$

t 秒後の座標

$$y = \frac{m}{k} \left\{ (v_0 \sin \theta_0 - \frac{m}{k} g) (1 - e^{-\frac{k}{m}t}) - gt \right\}$$



弾道計算シミュレーション 様々な要因と応用

- 発射された物体を正確に標的に当てるため
- 最初は軍事目的



- ゴルフ、野球などスポーツ科学
- 人工衛星、スペースシャトルの打ち上げ
- フライトシミュレータ
- 3Dゲームなどのレンダリング物理演算
- 分子動力学によるポリマー合成、ドラッグデザイン
- 天気予報

■ 要素

- 初速
- 仰角
- 空気抵抗（湿度、気温、気圧により密度が変わり空気抵抗が変化）
- 弾丸の前面投影面積、表面の摩擦係数、質量
- 風力、風向き
- コリオリの力、重力加速度
- 物体の自転（スピン）による揚力

シミュレーション
現実世界での現象を単純化、簡略化した数理モデル（又は模型）を用いて検証を行う模擬実験

世界初の〇〇計算機

- 世界初の計算機

- ENIAC, 1946年, Pennsylvania, USA

- 世界初の汎用機

- UNIVAC I, 1950年, USA

商業・業務利用に特化
可用性を追求

科学技術計算に特化
計算性能を追求

- 世界初の電卓

- Anita Mark8, 1963年, UK

- 世界初？のスーパーコンピュータ

- CDC 6600, 1964年, Lawrence
Livermore National Lab., USA.

- 世界初のパーソナルコンピュータ

- Altair 8800, 1975年, USA

3 MFLOPS

FLOP = Floating point number Operations Per Second

計算機の速さって？

- FLOPS (Floating-point Operations Per Second)
 - 一秒間に浮動小数点演算を何回できるか？
 - 京：約10Peta FLOPS (10^{16} 回)

Y：ヨタ

Z：ゼタ

E：エクサ

P：ペタ

T：テラ

G：ギガ

M：メガ

k：キロ

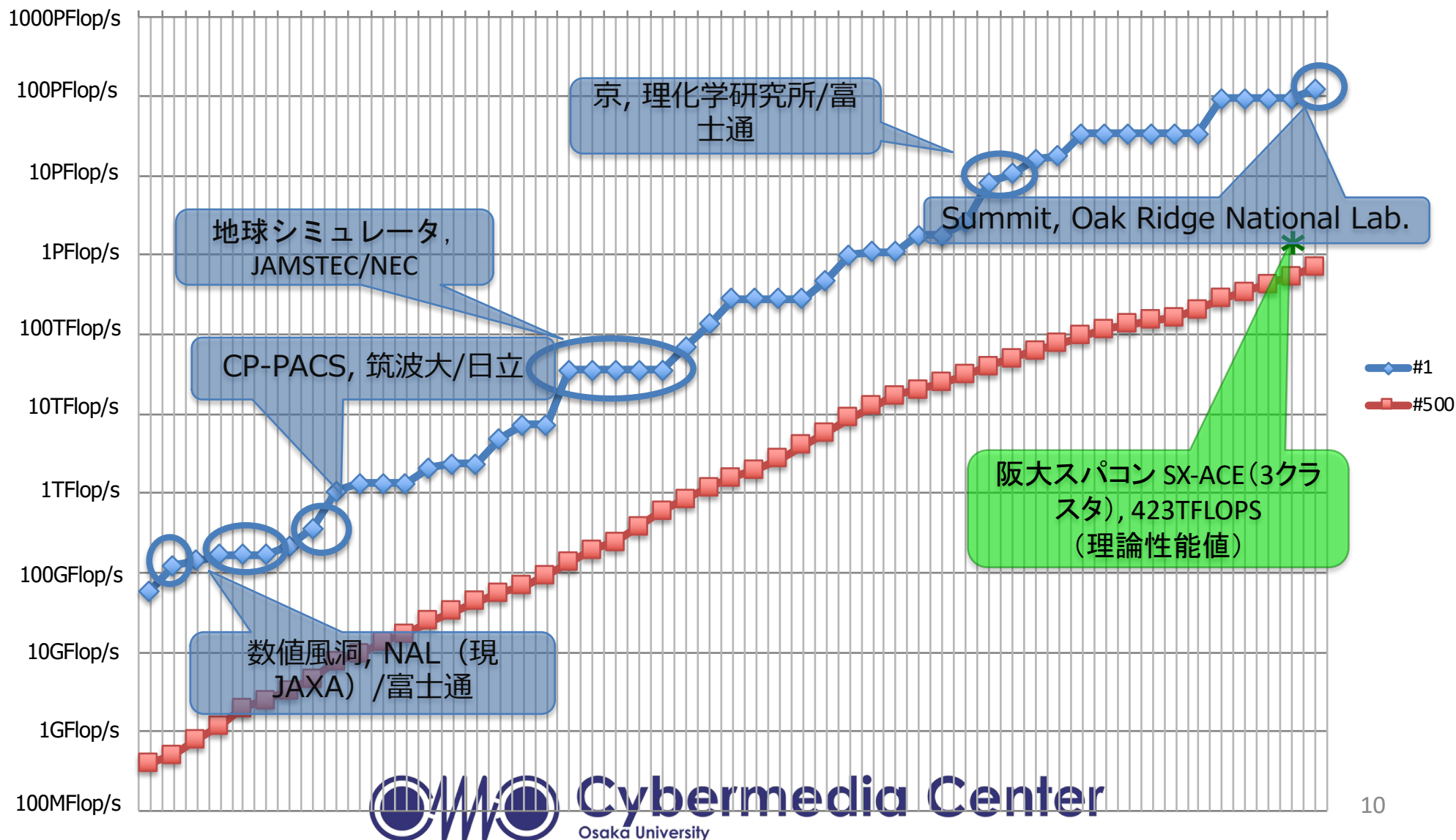
10,000,000,000,000,000

世界初のスパコン
CDC 6600

3,000,000

Top500 Jun. 2018

Jun,
1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018



日本のスパコン

名称・愛称	設置者	メーカー	性能	Top 500ランク
AI Bridging Cloud Infrastructure (ABCI)	産総研	Fujitsu	19.9 PFlops	5
Oakforest-PACS	東大・筑波大	Fujitsu	13. 6 PFlops	12
K Computer	理化学研究所	Fujitsu	10.5 PFlops	16
TSUBAME3.0	東工大	HPE	8.1 PFlops	19
-	気象庁	Cray	5.7 PFlops	25, 26

参考性能

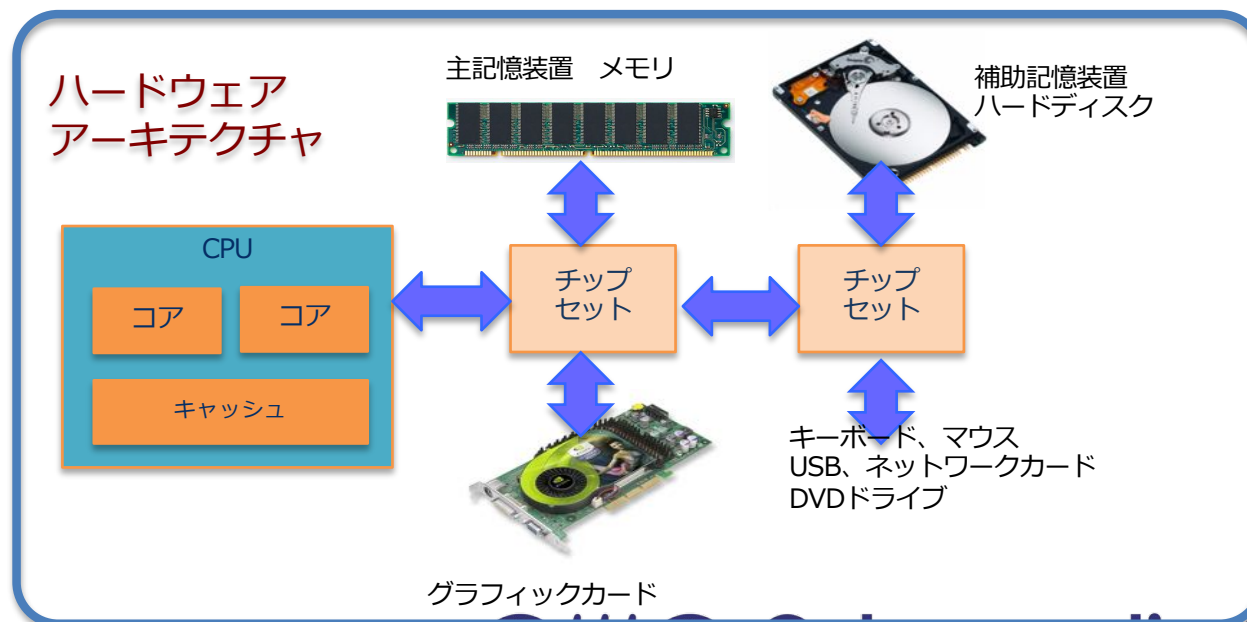
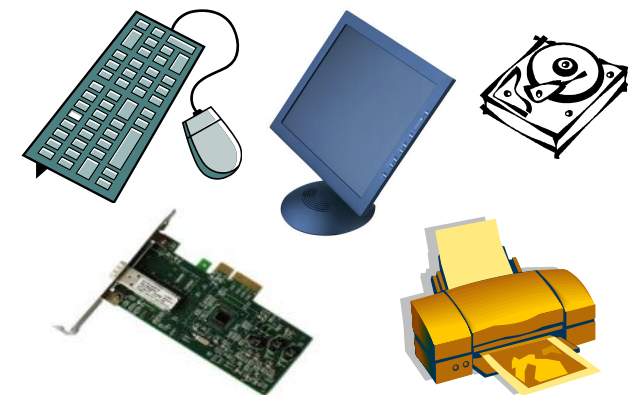
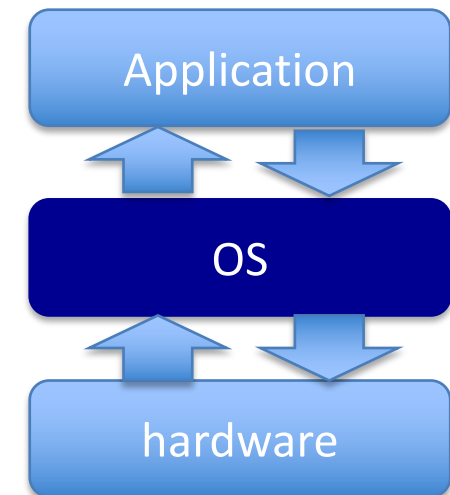
OCTOPUS	阪大	NEC	理論性能 1.4 Pflops	だいたい150位 から200位と 同くらいの性能
---------	----	-----	--------------------	--------------------------------

目次

1. スパコンの略歴
2. 計算機の概要
3. 並列計算
4. CMCのスパコン

計算機のアーキテクチャ

- 中央処理演算装置: CPU (プロセッサ)
 - 計算を行う頭脳
 - 命令により演算を行う
 - ベクタ部 (SIMD) とスカラ部を
あわせもつ
- 主記憶装置: メモリ
 - 揮発性が高く電源を落とすと
内容は破棄
- 補助記憶装置: ハードディスク
 - 不揮発性で電源を落としても内
容を保持
- グラフィックカード
 - 出力装置につなぐデバイス
 - GPGPU: 画像処理専用の補助演
算装置
- 入力装置: キーボード、マウス



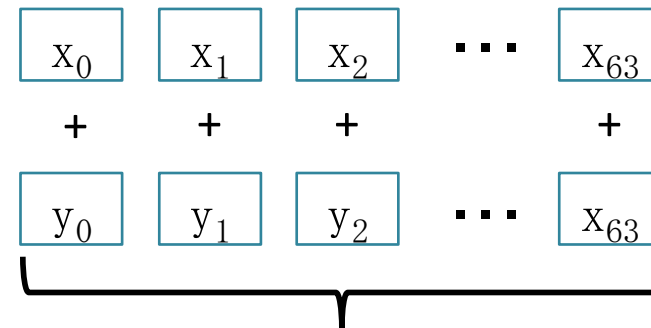
ベクタとスカラ

スパコン／プロセッサの種類

スカラ

- 計算機の命令を1つずつ実行
- 逐次的に命令を実行
- 高速化: パイプライン処理, スーパースカラ
- 代表システム: 京、Tsubame、etc.
- 得意な計算: 遺伝子相同性検索

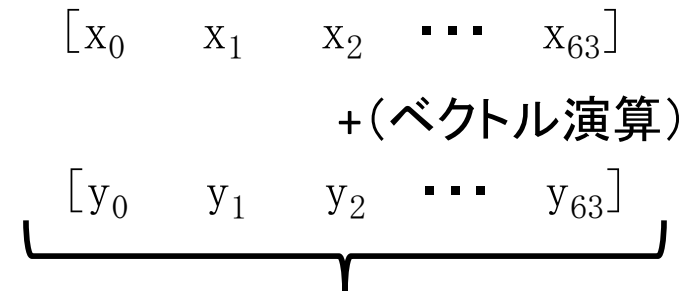
行列和の計算



64 スカラ命令

ベクタ

- 複数の命令を一つにまとめて実行
- 同じ命令(演算)に対し異なるデータ(項)で実行する場合、1つにまとめて実行することができる
- 代表システム: 阪大SX-ACE、地球シミュレータ
- 得意な計算: 気候シミュレーション、流体解析



1 ベクトル命令

計算機におけるOS

- ARPANETからいろいろなネットワークが接続されて超巨大なネットワーク → インターネットに変貌. The Internet

- ネットワークのモチベーション
 - データの共有
 - 大型計算機を複数のユーザで共有

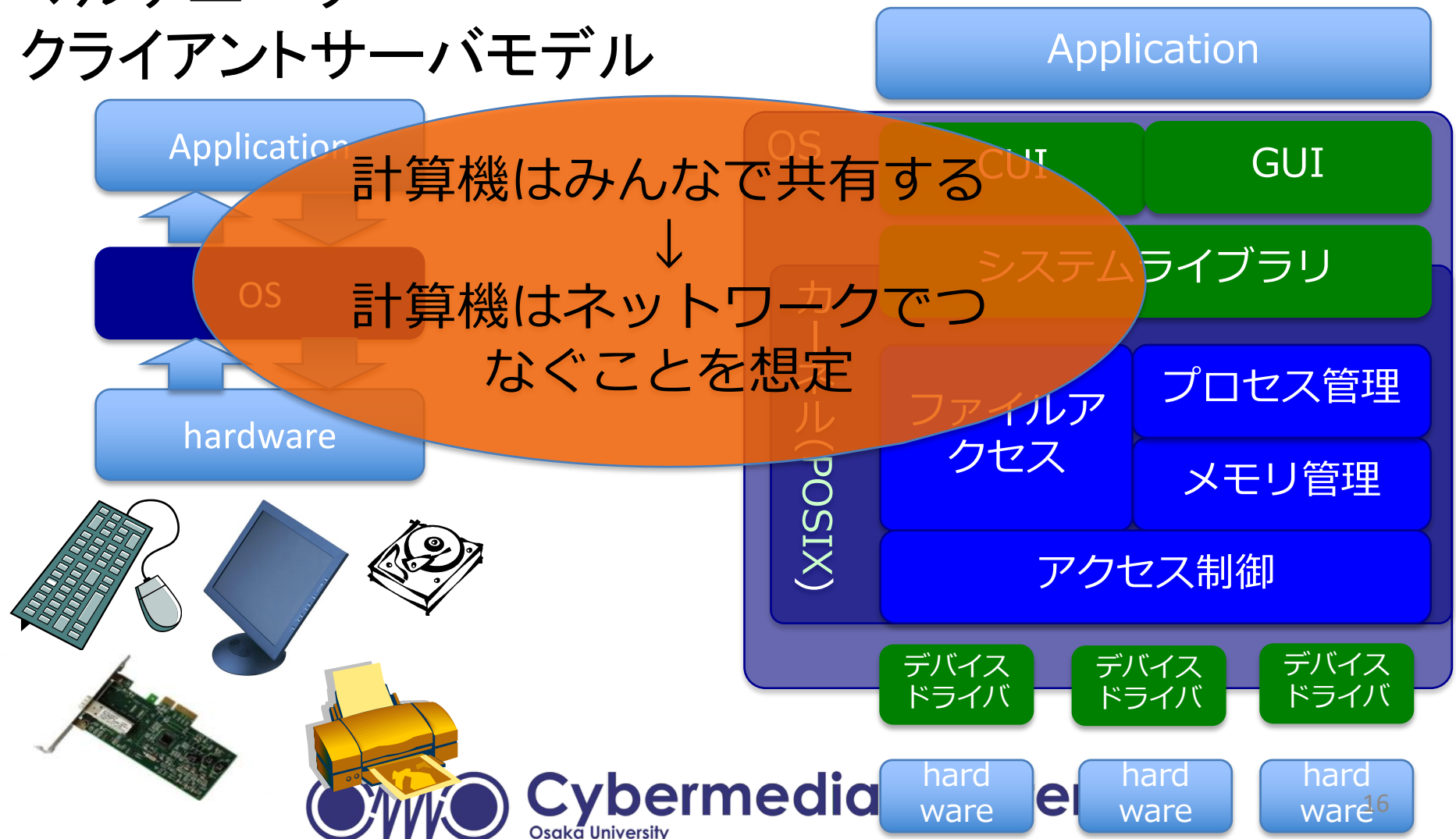


UNIXの登場

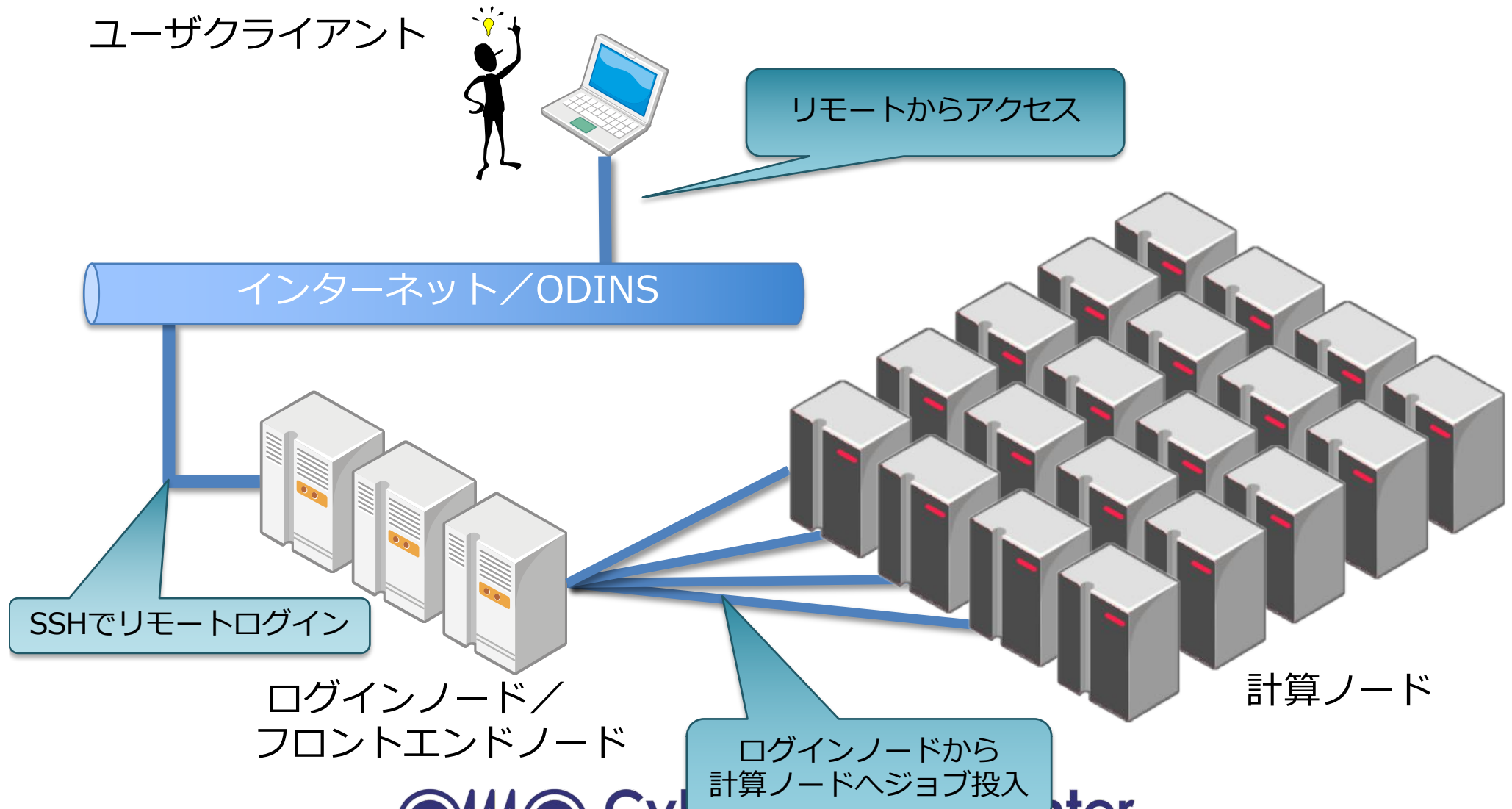
- Unics, 1969年 AT&T, USA
- Version 7 Unix, 1979年, AT&T, USA
- Linux, 1991年, GNU, USA
- Windows 3.1 1992年, Microsoft, USA
- Windows NT 3.5 1994年, Microsoft, USA
- Windows 95, 1995年, Microsoft, USA
- Mac OS X, 2001年, Apple, USA
- Windows XP, 2001年, Microsoft, USA
- Android, 2007年, Google, USA
- iOS, 2008年, Apple, USA

OS: UNIXのアーキテクチャ

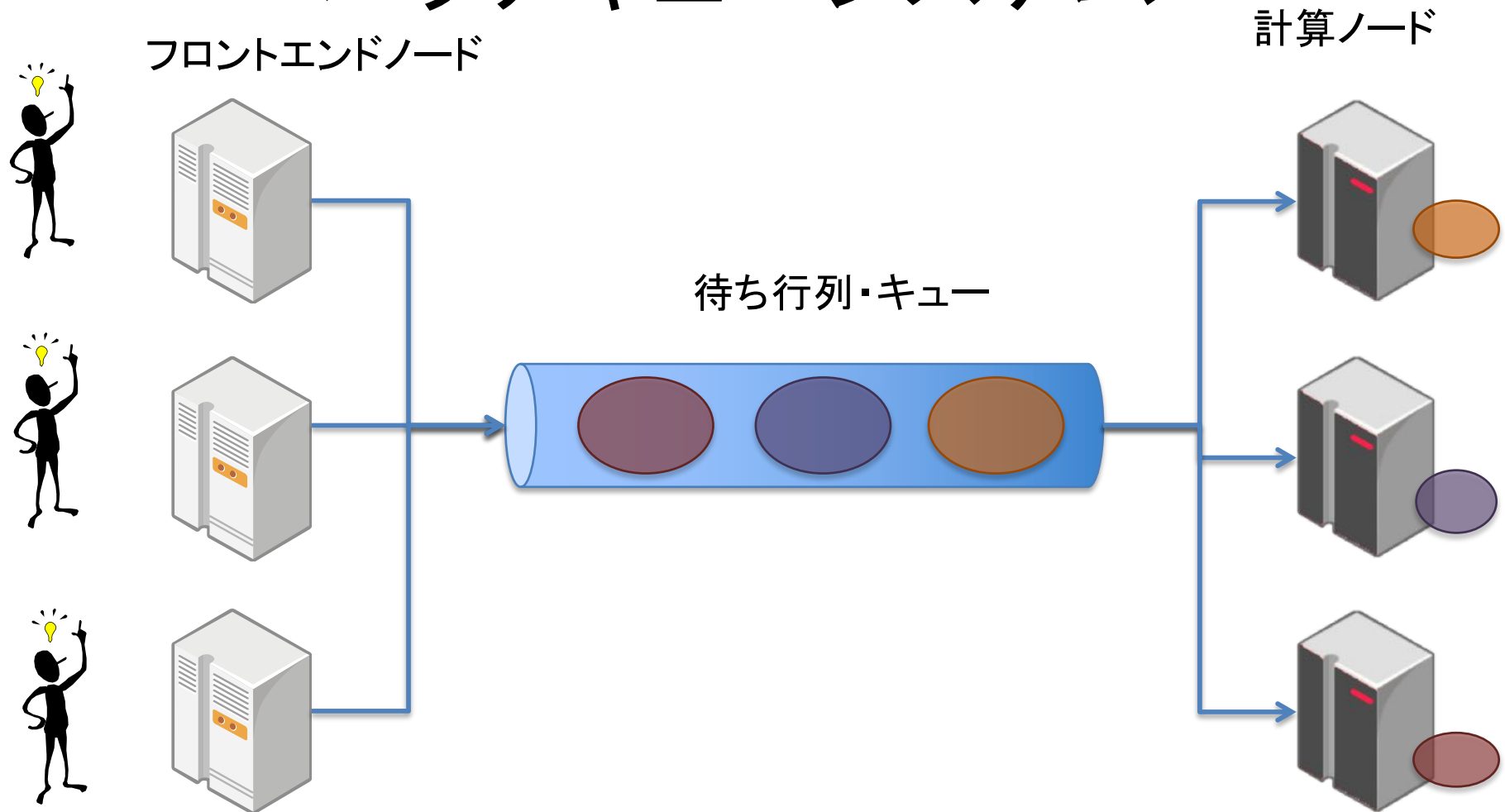
- マルチタスク(マルチプロセス)
- マルチユーザ
- クライアントサーバモデル



コンピュータ・クラスタ



ジョブ投入 バッチキューシステム



qsubでジョブ投入

計算機環境の指定

```
#!/bin/csh
#PBS -q ACE
#PBS -l elapstim_req=1:00:00,memsz_job=60GB
#PBS -v F_RSVTASK=4
setenv F_PROGINF DETAIL
cd $PBS_O_WORKDIR
./a.out
```

ジョブキューに登録

```
$ qsub a_batch.sh
Request 88156.cmc submitted to queue: ACE.
$
```

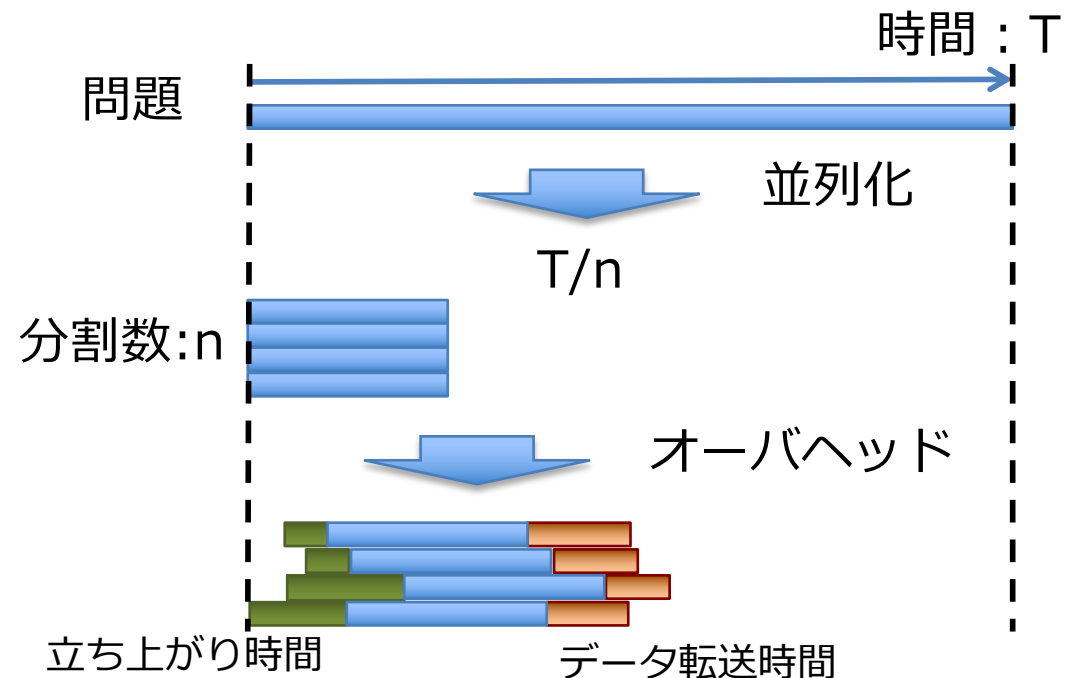
目次

1. スパコンの略歴
2. 計算機の概要
3. 並列計算
4. CMCのスパコン

並列プログラミングとは

- 逐次処理の問題、プログラム(実行時間: T)を n に分割し、 n 台のプロセッサ(or 計算機)で T/n 時間にする
- プロセッサに割り当てられるタスクは独立
- 並列化できる問題はデータに依存性のない問題のみに限られる

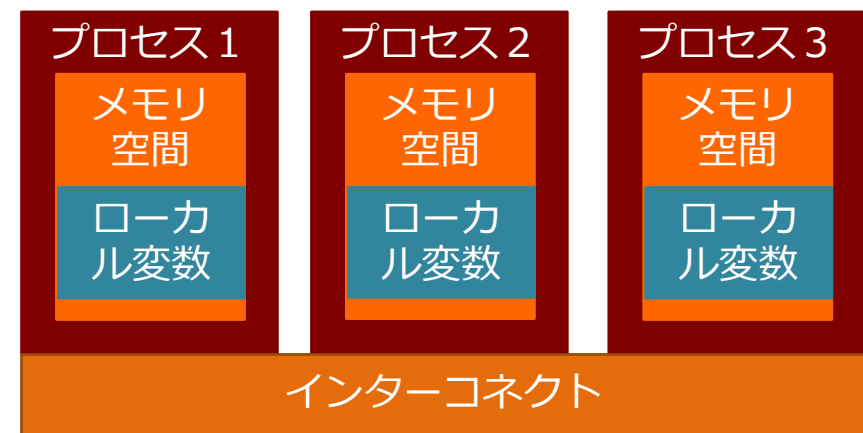
- 通信によるオーバーヘッド
 - 立ち上がり時間
 - データ転送時間



プロセス並列とスレッド並列

- プロセス並列
 - メモリ空間は独立
 - 並列タスク間でのデータ通信は必要に応じて
 - 例: Message Passing Interface (MPI)
- スレッド並列
 - メモリ空間を共有
 - データ通信の必要性なし
 - 例: OpenMP

プロセス並列



スレッド並列



Message Passing Interface (MPI)

- メッセージパッシング／プロセス間通信の規格
- 分散メモリ型並列計算機での並列実行に向く
- 大規模計算が可能に
- スケーラビリティ、性能は高
- 主要な実装例
 - MPICH
 - LAM
 - OpenMPI (C言語, C++, Fortran)
- 略語
 - Processor Element: プロセスの単位. MPIプロセスを指す.
 - Rank: PEの識別番号のこと. MPI_Comm_rank関数で設定されるランクID

MPIでHello, World.

```
#include <mpi.h>
#include <stdio.h>
#include <string.h>

int main(int argc, char **argv) {
    char msg[20];
    int myrank, num, i;
    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank( MPI_COMM_WORLD, &myrank);
    MPI_Comm_size( MPI_COMM_WORLD, &num);
    if ( myrank == 0 ) {
        strcpy(msg, "Hello, World.");
        for (i = 1; i < num; i++) {
            MPI_Send(msg, strlen(msg) + 1, MPI_CHAR, i, 99, MPI_COMM_WORLD);
        }
        printf("rank %d send: %s\n", myrank, msg);
    } else {
        MPI_Recv(msg, 20, MPI_CHAR, 0, 99, MPI_COMM_WORLD, &status);
        printf("rank %d receive: %s\n", myrank, msg);
    }
    MPI_Barrier(MPI_COMM_WORLD);
    MPI_Finalize();
    return 0;
}
```

Rank 0 は送信側

その他のRankは受信側

```
$ mpicc hello_mpi.c -o m.out
$ mpirun -n 4 ./m.out
rank 0 send: Hello, World.
rank 1 receive: Hello, World.
rank 2 receive: Hello, World.
rank 3 receive: Hello, World.
```

OpenMPによるスレッド並列

- MPI
 - プログラムが並列化を意識してコードを書く必要がある
- OpenMP
 - 1ノードの中で閉じた並列処理
 - コンパイラが自動的に並列化 (並列化効率にはコンパイラに依存)
 - ソースコード中にOpenMPディレクティブを挿入し、OpenMP環境下では有効になり、それ以外では無効になる
 - 並列化と非並列では同じソースコードとなる

```
#ifdef _OPENMP
    //OpenMPを使う
#else
    //OpenMPを使わない
#endif
```

```
#pragma omp parallel for
    for (i = 0; i < 1000; i++) {
        // 並列処理させたいコード
    }
```

OpenMPでHello, World.

```
#include <omp.h>
#include <stdio.h>
```

```
int main(int argc, char **argv) {
    #pragma omp parallel num_threads(4)
    {
        printf("Thread %d, Hello, world\n", omp_get_thread_num());
    }
    return 0;
}
```

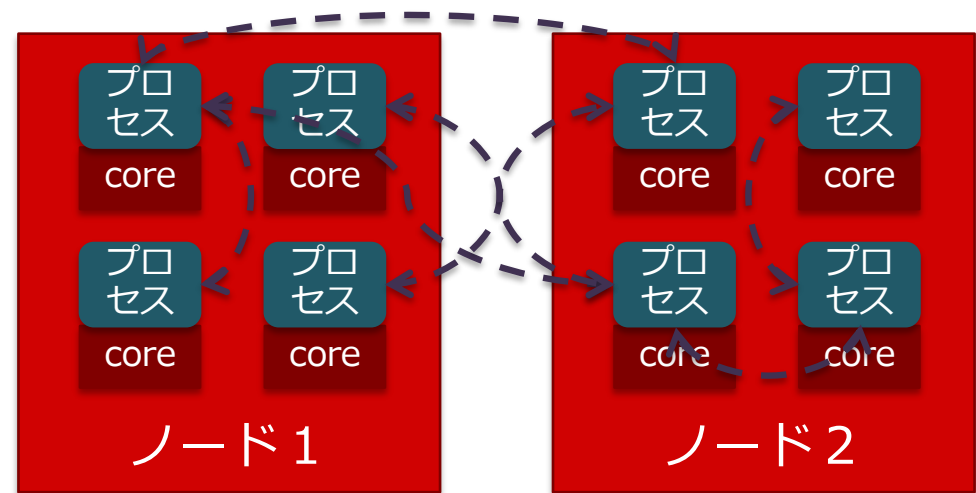
4つのスレッド並列

```
$ gcc -fopenmp hello_omp.c -o o.out
$ ./o.out
Thread 3, Hello, world
Thread 0, Hello, world
Thread 1, Hello, world
Thread 2, Hello, world
```

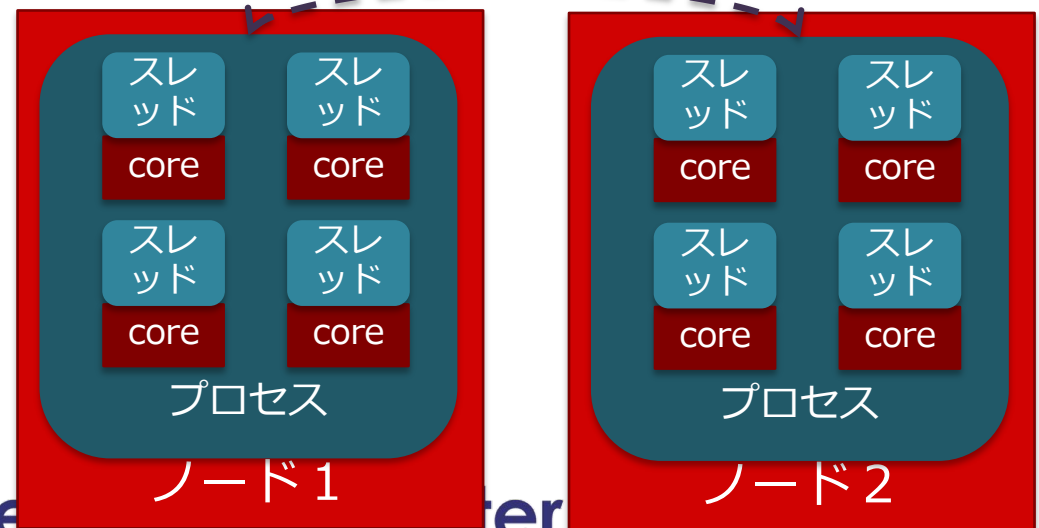
ハイブリッド並列

- フラット並列モデル
 - プロセス並列のみ
 - コアごとにプロセス
 - 通信がプロセス間ごとに発生
- ハイブリッド並列モデル
 - プロセス並列(ノード間) + スレッド並列(ノード内)
 - コア数が増えてもプロセス数は増加せず
 - 通信が混雑しない

フラット並列



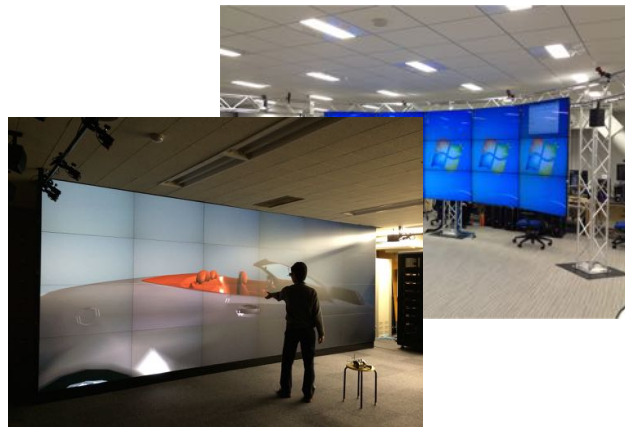
ハイブリッド並列



CMC大規模計算機システムサービス

スカラープロセッサ

新システム“OCTOPUS” 大規模可視化対応PCクラスター (VCC)
2018年4月運用開始

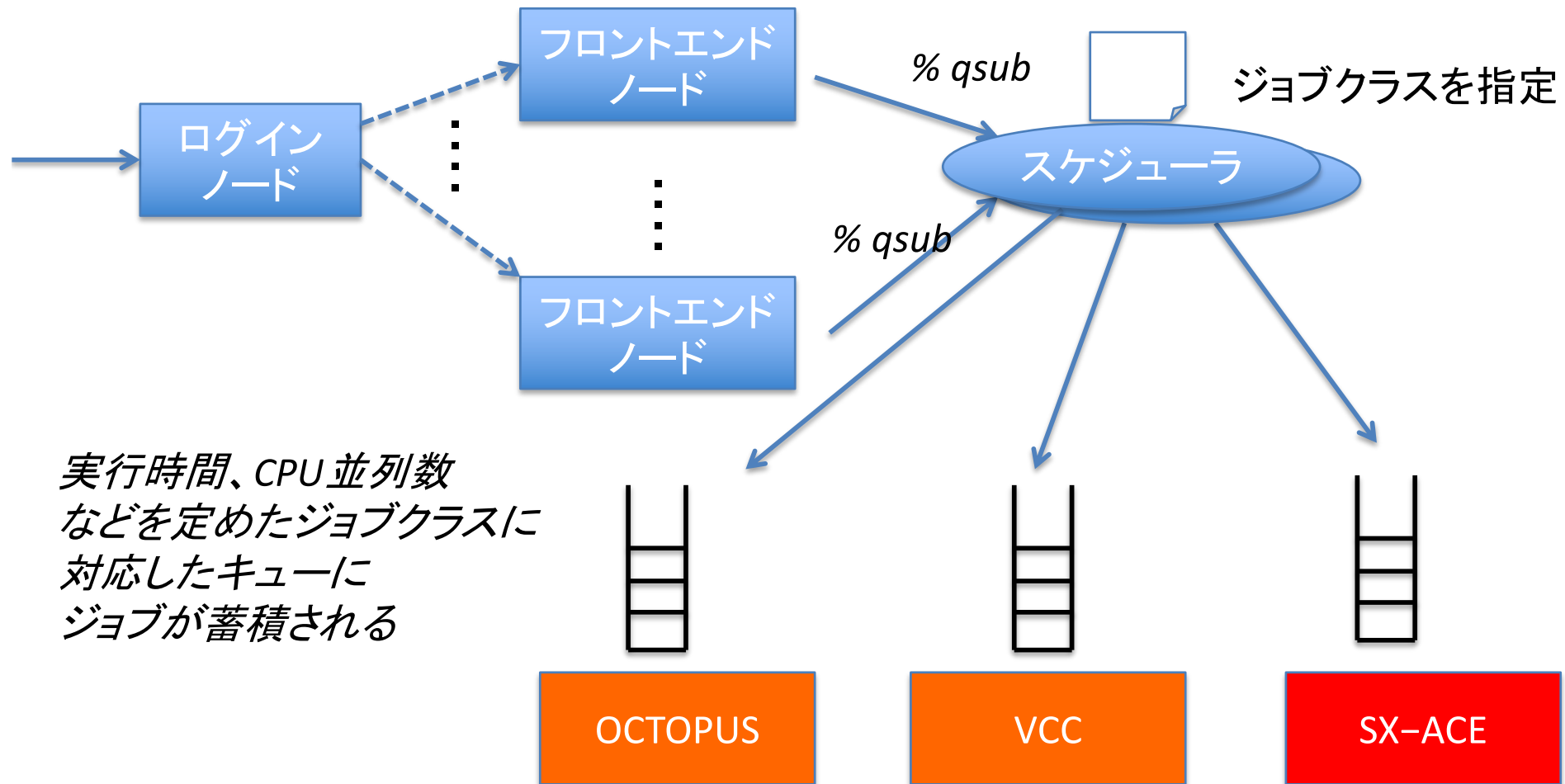


ベクタープロセッサ

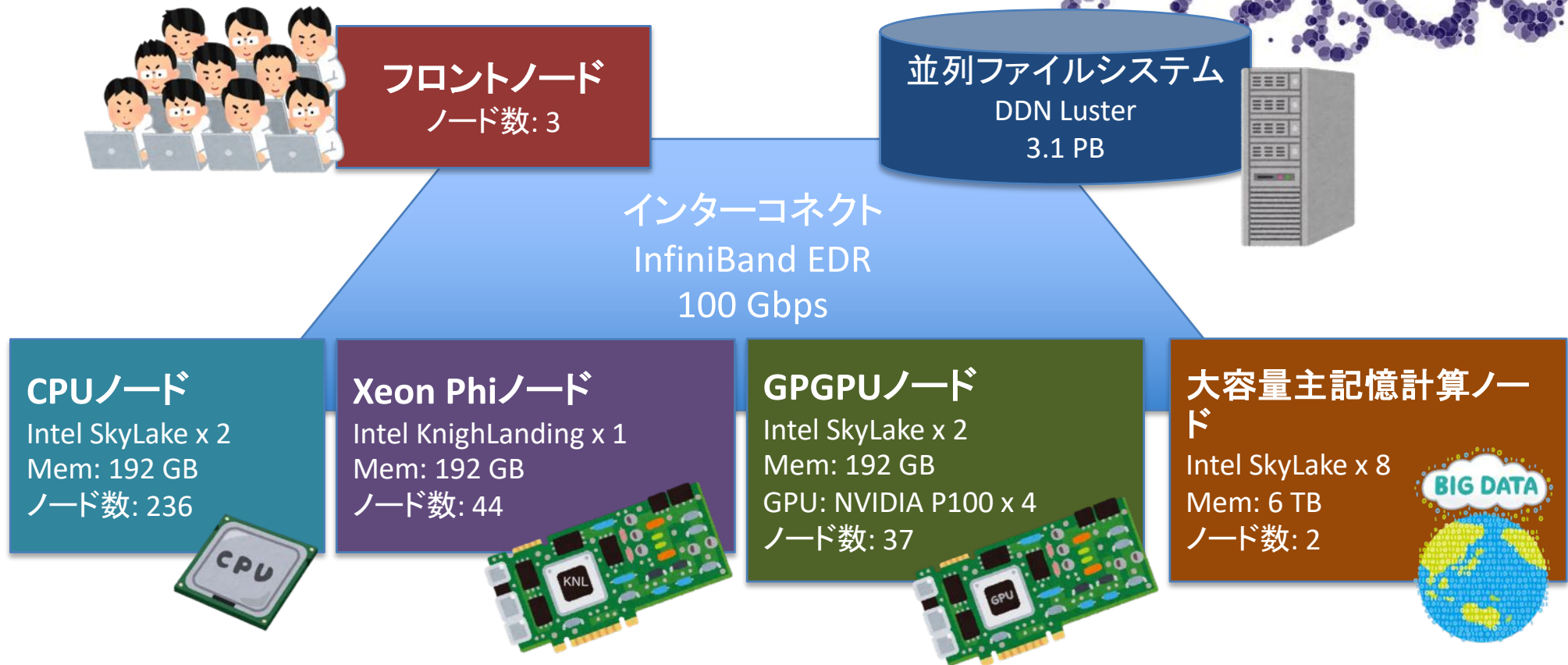
スーパーコンピュータ
“SX-ACE”



CMC大規模計算機システムの利用方法



OCTOPUS



<http://www.hpc.cmc.osaka-u.ac.jp/octopus/>

講習会の予定

- 9/10 スパコンに通じる並列プログラミングの基礎
- 9/13 スーパーコンピュータ概要とスーパーコンピュータ利用入門
- 9/19 SX-ACE高速化技法の基礎
- 9/20 並列コンピュータ 高速化技法の基礎
- 9/26 SX-ACE 並列プログラミング入門(MPI)
- 10/10 AVS可視化処理入門
- 10/11 AVS可視化処理応用／特別相談会
- http://www.hpc.cmc.osaka-u.ac.jp/lecture_event/lecture/