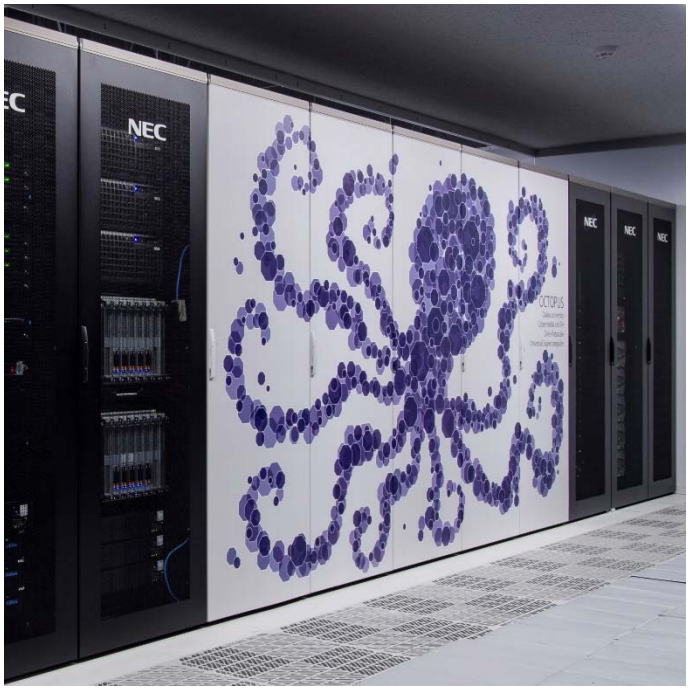
OCTOPUSの利用方法

大阪大学 サイバーメディアセンター
大規模計算機システム担当

OCTOPUSの紹介

4種類のノードと3PBのストレージで構成された
ハイブリッド型スーパーコンピュータ
各ノード間は100Gbpsで通信可能



	汎用CPU ノード	GPU ノード	Xeon Phi ノード	大容量主記憶 搭載ノード	合計
コア数	24	24	64	128	**9624**
演算性能	1.996 TFLOPS	23.196 TFLOPS	2.662 TFLOPS	8.192 TFLOPS	**1.463 PFLOPS**
メモリ	192GB	192GB	192GB	6TB	**72.864TB**
ノード数	236ノード	37ノード	44ノード	2ノード	**319ノード**

OCTOPUS利用の流れ

ユーザー端末

フロントエンド
ノードへの接続

OCTOPUS
フロントエンドノード

プログラム準備

ジョブスクリプト
作成

OCTOPUS
計算ノード

プログラム実行

フロントエンドノードへの接続

SSH （Secure Shell）接続

Windowsの方
ターミナルソフトを使用
（ TeraTerm, Putty等）

Mac/Linuxの方
ターミナルから
sshコマンドを使用

ssh *userid*@octopus.hpc.cmc.osaka-u.ac.jp

接続先

octopus.hpc.cmc.osaka-u.ac.jp

みなさまの端末から接続をお願いします

OCTOPUS利用の流れ

ユーザー端末

フロントエンド
ノードへの接続

OCTOPUS
フロントエンドノード

プログラム準備

ジョブスクリプト
作成

OCTOPUS
計算ノード

プログラム実行

プログラム準備

OCTOPUSでは多様なソフトウェア、
プログラムを実行可能です

主なソフトウェア

Gaussian16, GROMACS, OpenFOAM, LAMMPS, Caffe,Theano,Chainer,TensorFlow,GAMESS, HΦ, MODYLAS, NTChem, OpenMX, SALMON, SMASH, VisIt

主なプログラミング言語

FORTRAN, C, C++, Python, R, Julia

OCTOPUS利用の流れ

ユーザー端末

フロントエンド
ノードへの接続

OCTOPUS
フロントエンドノード

プログラム準備

**ジョブスクリプト
作成**

OCTOPUS
計算ノード

プログラム実行

計算機の利用方法

インタラクティブ利用

コマンド等を通してコンピュータに直接命令し、
リアルタイムで処理を実行

操作として手軽

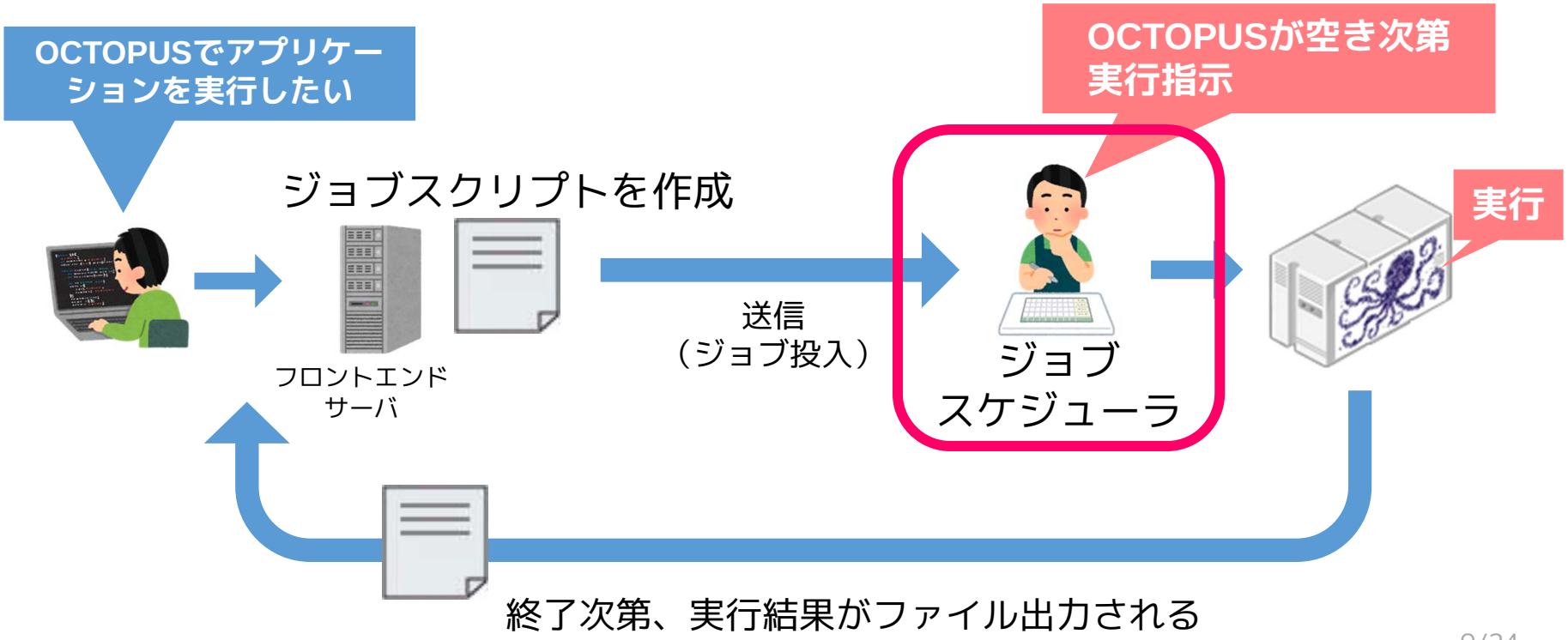
バッチ利用

コンピュータにまとめて処理を命令し実行

処理の命令が終われば、ログアウトしてもOK

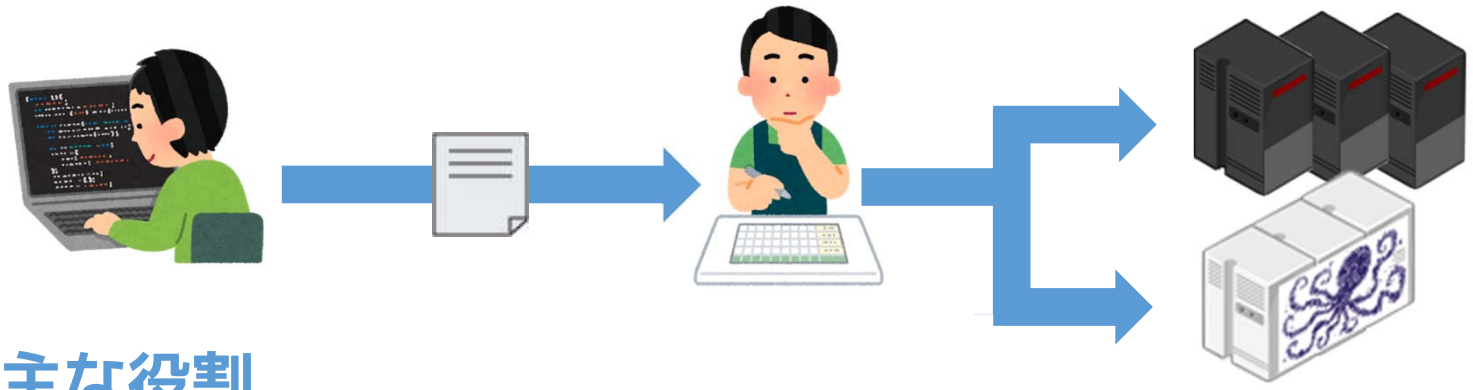
バッチ利用

処理を「ジョブスクリプト」に記述
スクリプトに基づき計算機が処理を実行



ジョブスケジューラとは

あらかじめ管理者によって設定された資源割当ポリシーに従い、ジョブを計算資源に割り当てるソフトウェア



主な役割

計算機システム各ノードのディスク容量、メモリ容量、性能を把握
ノード毎の資源使用率を定期的に監視、管理
ユーザより実行したいジョブ要求を受信し、適切なノードを選定
ジョブ実行に伴う入出力データのファイル転送

ジョブスケジューラとは

当センターでは**バックフィル型**を採用

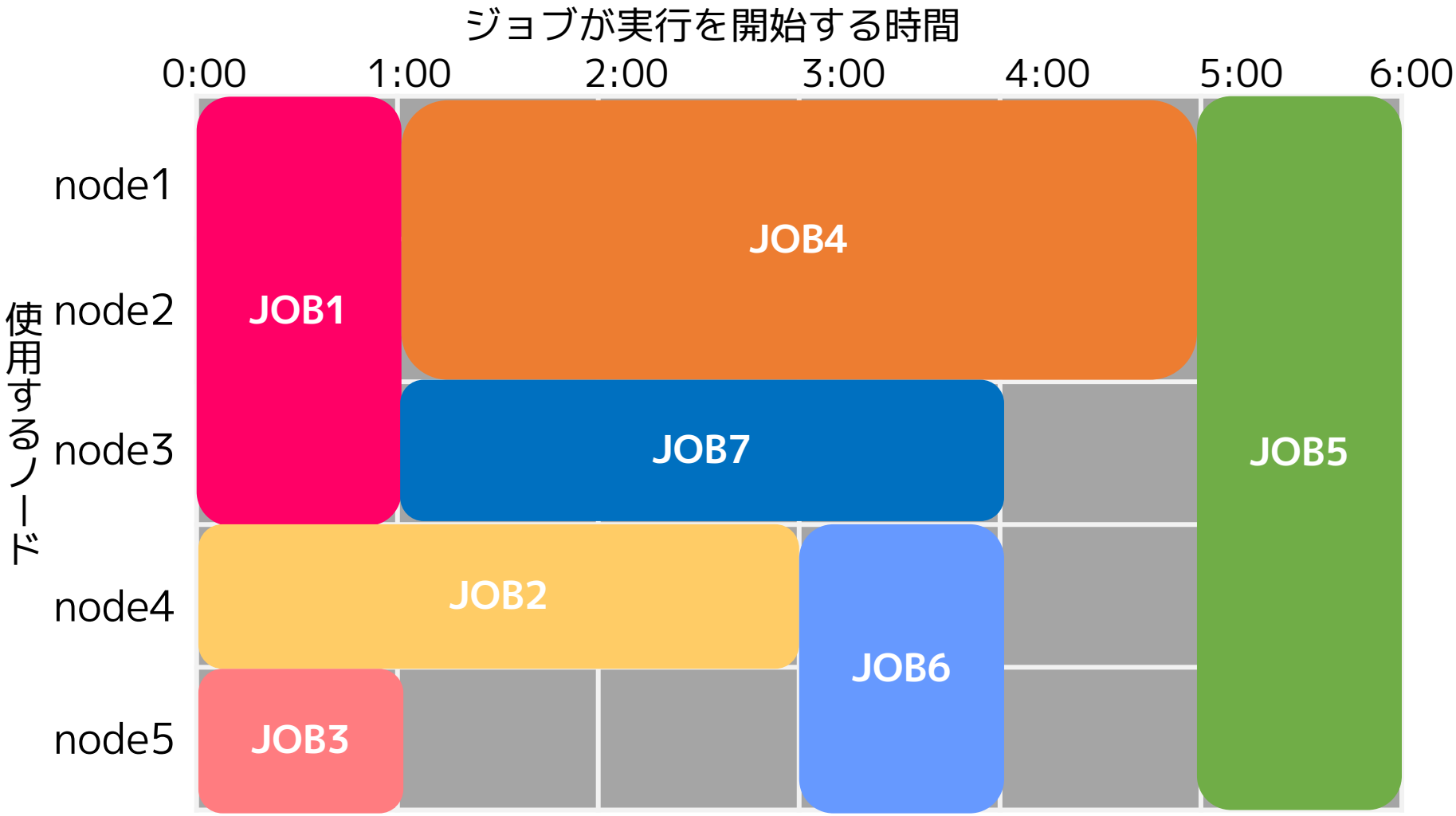
特徴

ジョブの実行開始時間のマップを作成する

マップに載れば、実行開始時間が保障される

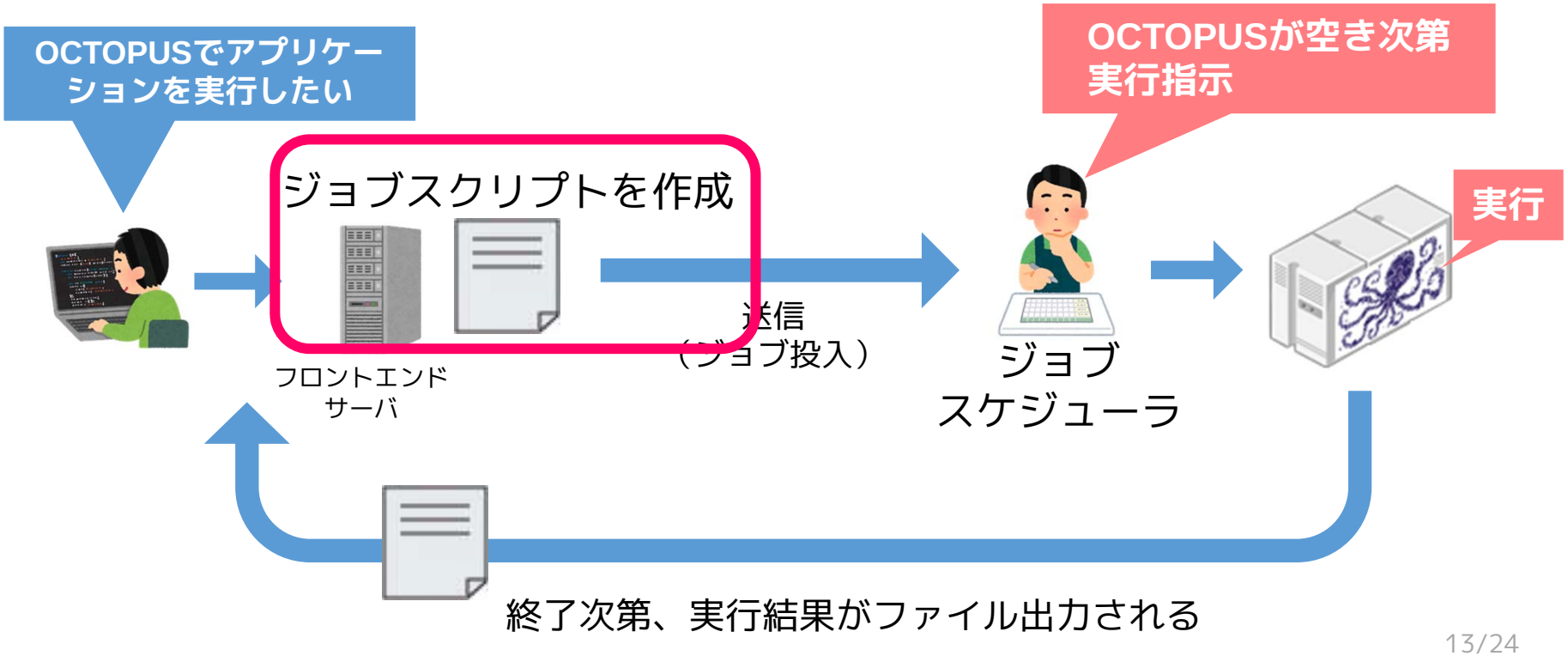
実行中は指定したリソースを占有して割当てる

ジョブスケジューラのイメージ



バッチ利用

処理を「ジョブスクリプト」に記述
スクリプトに基づき計算機が処理を実行



ジョブスクリプト

```
#!/bin/bash

#PBS –q OCTOPUS
#PBS –l elapstim_req=1:00:00

cd $PBS_O_WORKDIR
./a.out
```

OCTOPUSのリソースや環境設定

実行したい処理を記載したシェルスクリプト

ジョブスクリプト

```
#!/bin/bash
#PBS –q OCTOPUS
#PBS –l elapstim_req=1:00:00
cd $PBS_O_WORKDIR
./a.out
```

使用する
リソースや環境

NQSIIオプション(#PBS～)でリソースや環境の設定を行う

オプション	説明
#PBS -q	**ジョブクラスを指定し、計算に使用する計算機やリソースを指定する**
#PBS -l	**使用する資源値**
	elapstim_req ： ジョブの経過時間
	memsz_job ： 1ノードあたりのメモリ量
	cpunum_job ： 1ノード当たりのCPU数
#PBS -v	環境変数の指定(setenvではなくこちらを使うことを推奨する)
#PBS -T	MPI 実行時に指定(IntelMPIの場合、#PBS -T intmpi と指定)
#PBS -b	使用するノード数
#PBS -y	あらかじめ予約した計算ノードを使用する場合に指定(本講習会でも使用) ※普段は使用しません

必須！

ジョブスクリプト

```
#!/bin/bash
#PBS –q OCTOPUS
#PBS –l elapstim_req=1:00:00
cd $PBS_O_WORKDIR
./a.out
```

使用する
リソースや環境

ジョブクラス	利用可能 経過時間	利用可能 CPU数	利用可能 メモリ	同時利用 可能ノード数
OCTOPUS	120時間	3,072Core (24Core×128ノード)	24,576GB (192GB×128ノード)	128ノード
OCTPHI	120時間	2,048Core (64Core×32ノード)	6,144GB (192GB×32ノード)	32ノード
OCTMEM	120時間	256Core (128Core×2ノード)	12TB (6TB×2ノード)	2ノード
LECTURE	イベント用のジョブクラスです 本講習会ではこちらを使用します			

CPUノード
GPUノード

Xeon Phiノード

大容量主記憶
搭載ノード

ジョブスクリプト

```
#!/bin/bash

#PBS –q OCTOPUS
#PBS –l elapstim_req=1:00:00

cd $PBS_O_WORKDIR
./a.out
```

OCTOPUSで
実行する処理

ファイルやディレクトリの実行・操作を記述
記述方法はシェルスクリプト

$PBS_O_WORKDIR ：ジョブ投入時のディレクトリが設定される

ジョブスクリプト

```
#!/bin/bash
#PBS –q OCTOPUS
#PBS –l elapstim_req=1:00:00
cd $PBS_O_WORKDIR
./a.out
```

ジョブクラスの指定

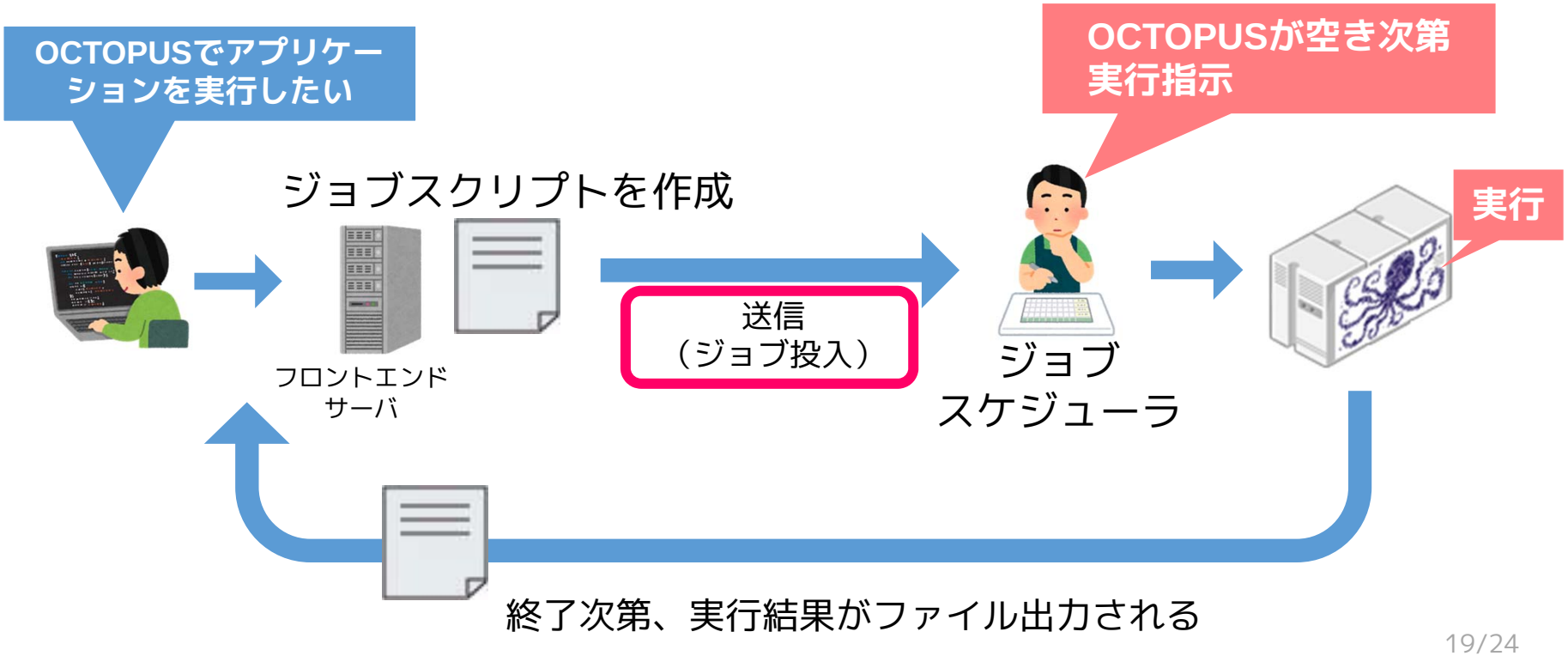
リソースの指定
コンマ後にスペースを入れないよう注意！

ジョブ投入時のディレクトリへ移動

a.outを実行する

バッチ利用

処理を「ジョブスクリプト」に記述
スクリプトに基づき計算機が処理を実行



ジョブの操作方法

ジョブの投入コマンド

$ qsub [ジョブスクリプトファイル]

投入に成功すると
“Request [RequestID] submitted to queue: ジョブクラス名”
と表示され、ジョブごとにRequestIDという通し番号が付与される

ジョブのキャンセルコマンド

$ qdel [RequestID]

キャンセルに成功すると
“Request [RequestID] was deleted”と表示される

投入済みジョブの確認方法

ジョブの状態確認コマンド

```
$ qstat
```

RequestID	ReqName	UserName	Queue	STT	Memory	CPU	Elapse
12345.cmc	nqs-test	a61234	ACE	RUN	8.72G	830.66	208

ジョブの状態

待ち状態では「QUE」
実行が始まると「RUN」 となる。

実行時間

CPU：実際にジョブが消費した時間
　　複数CPU指定の場合は、全CPUを累積表示

Elapse：ジョブが実行されてからの経過時間

ジョブのスケジューリング状況確認コマンド

```
$ sstat
```

RequestID	ReqName	UserName	Queue	Pri	STT	PlannedStartTime
12345.cmc	nqs-test	a61234	ACE	-1.5684/ -1.5684	ASG	2015-06-16 00:01:23

状態監視

実行時刻が決まると「ASG」表示になる。

混雑具合や優先度により、「実行時間の決定」までの待ち時間が異なるが、一旦実行時間が決定されるとその時刻にジョブ実行が始まる。

実行開始時刻

システムメンテナンスやトラブル時は
再スケジュールされることをご了承ください。

実行結果の確認方法

実行結果,エラーは指定しない限り「標準出力」となる

標準出力は**ジョブスクリプト名.oリクエストID**
標準エラー出力は**ジョブスクリプト名.eリクエストID**
というファイル名で自動出力される

catやlessコマンドでファイルの内容を出力し確認

$ cat jobscript.nqs.o12345

意図通りの結果が表示されていれば計算は成功

演習:OCTOPUSでプログラム実行

1. サンプルのジョブスクリプト,プログラムをコピー
   ```
   cp -r /octfs/apl/kosyu/20191016 ~/
   ```

2. ジョブスクリプトを確認
   ```
   cd ~/20191016/
   cat jobscript.nqs
   ```

   LECTUREは講習会用ジョブクラスです
   “＃PBS -y 294”は講習会用の資源予約IDです

3. ジョブスクリプトを投入
   ```
   qsub jobscript.nqs
   ```

4. ジョブの状態確認
   ```
   qstat
   sstat
   ```

   何も表示されていない場合は
   すでに実行終了したor投入に失敗している

5. 実行結果の確認
   ```
   cat jobscript.nqs.o123456 (標準出力)
   cat jobscript.nqs.e123456 (標準エラー出力)
   ```

より高度な利用に向けて

利用の参考になるWebページ

サイバーメディアセンター 大規模計算機システム Webページ
http://www.hpc.cmc.osaka-u.ac.jp/system/manual/

利用方法
http://www.hpc.cmc.osaka-u.ac.jp/system/manual/

FAQ
http://www.hpc.cmc.osaka-u.ac.jp/faq/

お問い合わせ
http://www.hpc.cmc.osaka-u.ac.jp/support/contact/auto_form/

研究成果
http://www.hpc.cmc.osaka-u.ac.jp/researchlist/