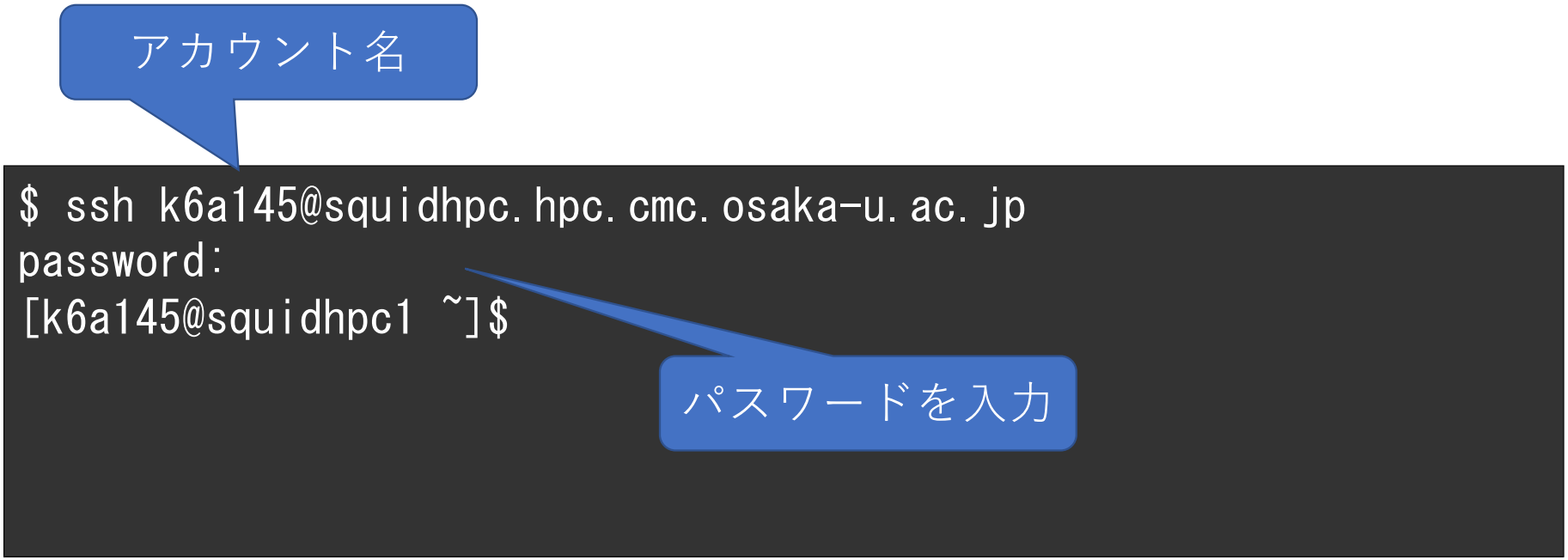
Rハンズオンセミナー
SQUIDでRを使おう

大阪大学サイバーメディアセンター
木戸　善之
mailto:kido@cmc.osaka-u.ac.jp
2021/11/12

セミナーを始める前に

- ターミナルソフトを開いて，ログインしてみてください



必要なファイルをワークディレクトリにコピー

```
[k6aXXX@squidhpc1 ~]$ cp -fr /system/lecture/seminar/R/*
/sqfs/work/kosyuXXX/k6aXXX/

[k6aXXX@squidhpc1 ~]$ cp /system/lecture/seminar/R/bashrc.sample ~
```

ユーザごとに異なります
kosyuXXXのXXXはアカウント末尾3桁

.bashrcの編集

- 環境変数の設定が必要

```
[k6a145@squidhpc3 ~]$ vi .bashrc
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi

# User specific environment
PATH="$HOME/.local/bin:$HOME/bin:$PATH"
export PATH

# Uncomment the following line if you don't like systemctl's
# export SYSTEMD_PAGER=

# User specific aliases and functions

export R_LIBS=/sqfs/work/kosyu145/k6a145/R_lib
export ORTED=/system/apps/rhel8/cpu/OpenFOAM/intel2020u4/v2012/ThirdParty-v2012/openmpi-4.0.3/orte/orted

module load BaseCPU/2021
module load BaseR/2021
```

ユーザごとに異なります
ご注意ください

.bashrcの適用

- 編集したbashrcを適用する
- R_LIBのディレクトリを作る

```
[k6aXXX@squidhpc1 ~]$ source ~/.bashrc
Loading BaseCPU/2021
  Loading requirement: intel/2020update4
Loading BaseR/2021
  Loading requirement: R/4.0.3
[k6a145@squidhpda3 k6a145]$ mkdir -p /sqfs/work/kosyu145/k6a145/R_lib
```

日本のスパコン

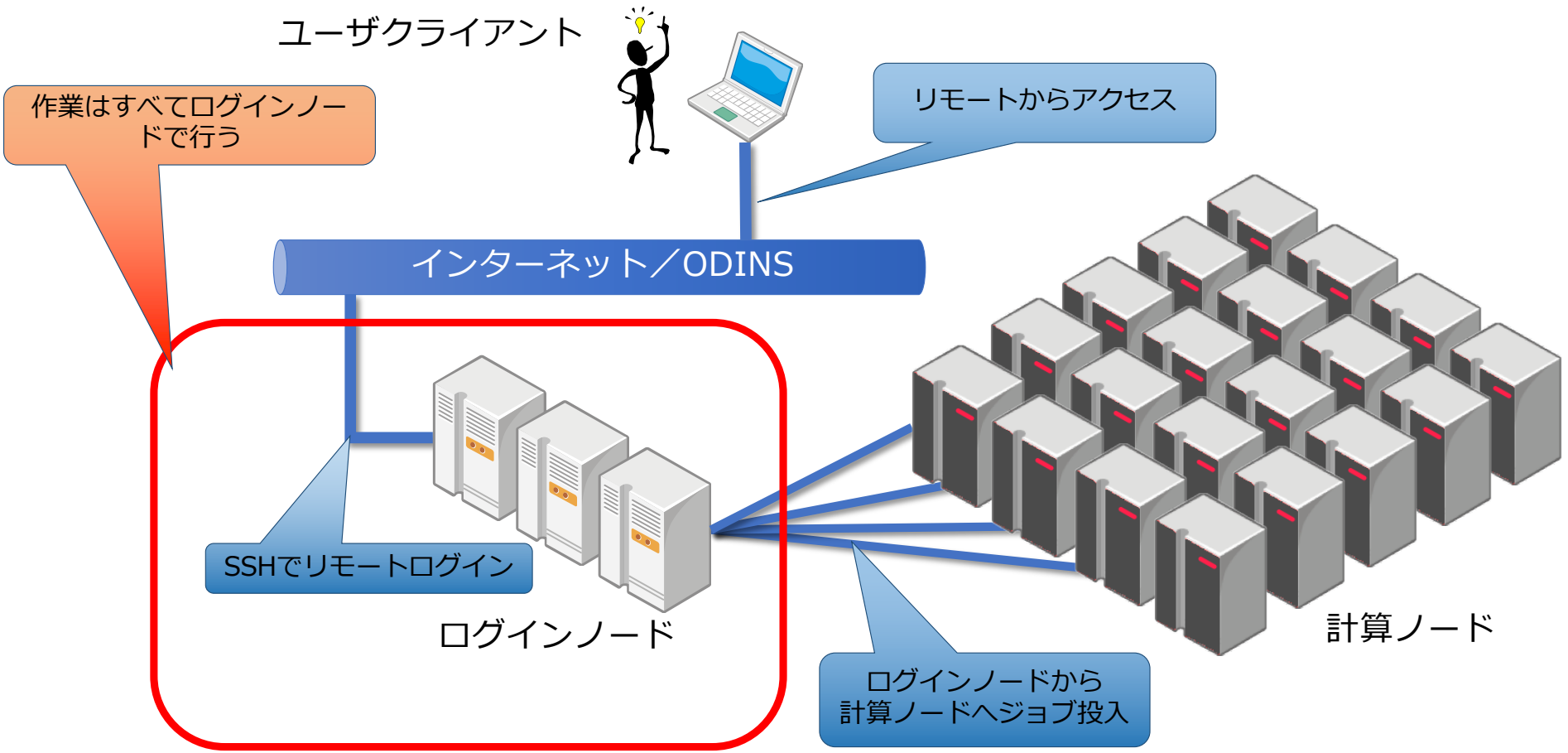
名称・愛称	設置者	メーカ	実行性能	Top 500ランク
富岳	理研	Fujitsu	442.0 PFlops	1
AI Bridging Cloud Infrastructure (ABCI) 2.0	産総研	Fujitsu	22.2 PFlops	12
TOKI-SORA	JAXA	Fujitsu	16.5 PFlops	25
Oakforest-PACS	東大・筑波大	Fujitsu	13.6 PFlops	33
Earth Simulator -SX-Aurora TSUBASA	JAMSTEC	NEC	9.9 PFlops	39
TSUBAME3.0	東工大	HPE	8.1 PFlops	50
Plasma Simulatar	NIFS	NEC	7.8 PFlops	54
不老	名古屋	Fujitsu	6.6 PFlops	62
-	気象庁	HPE	6.1 PFlops	66
SQUID	**阪大**	**NEC**	**6.1 PFlops**	**67**

SQUIDの性能

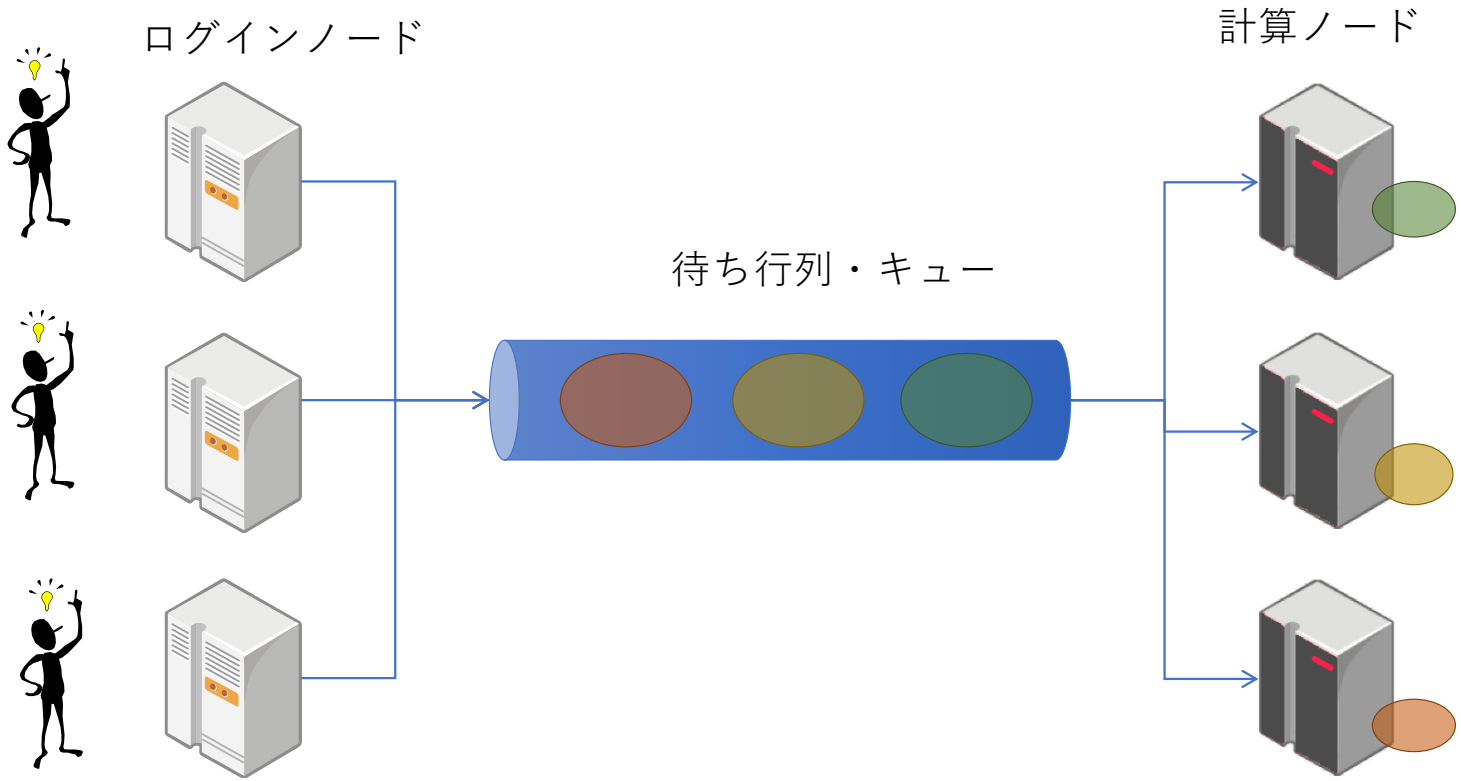
総演算性能	16.591 PFLOPS	
ノード構成	**汎用CPUノード 1,520ノード(8.871 PFLOPS)**	**Intel Xeon Platinum 8368 (38コア) ×2基 メモリ：256 GB**
	GPUノード 42ノード(6.797 PFLOPS)	Intel Xeon Platinum 8368 (38コア) ×2基 メモリ：512 GB GPU：NVIDIA Tesla A100 ×8基
	ベクトルノード 36ノード(0.922 PFLOPS)	AMD EPYC 7402P (24コア) ×1基 メモリ：128 GB ベクトルエンジン：NEC SX-Aurora TSUBASA Type20A ×8基

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

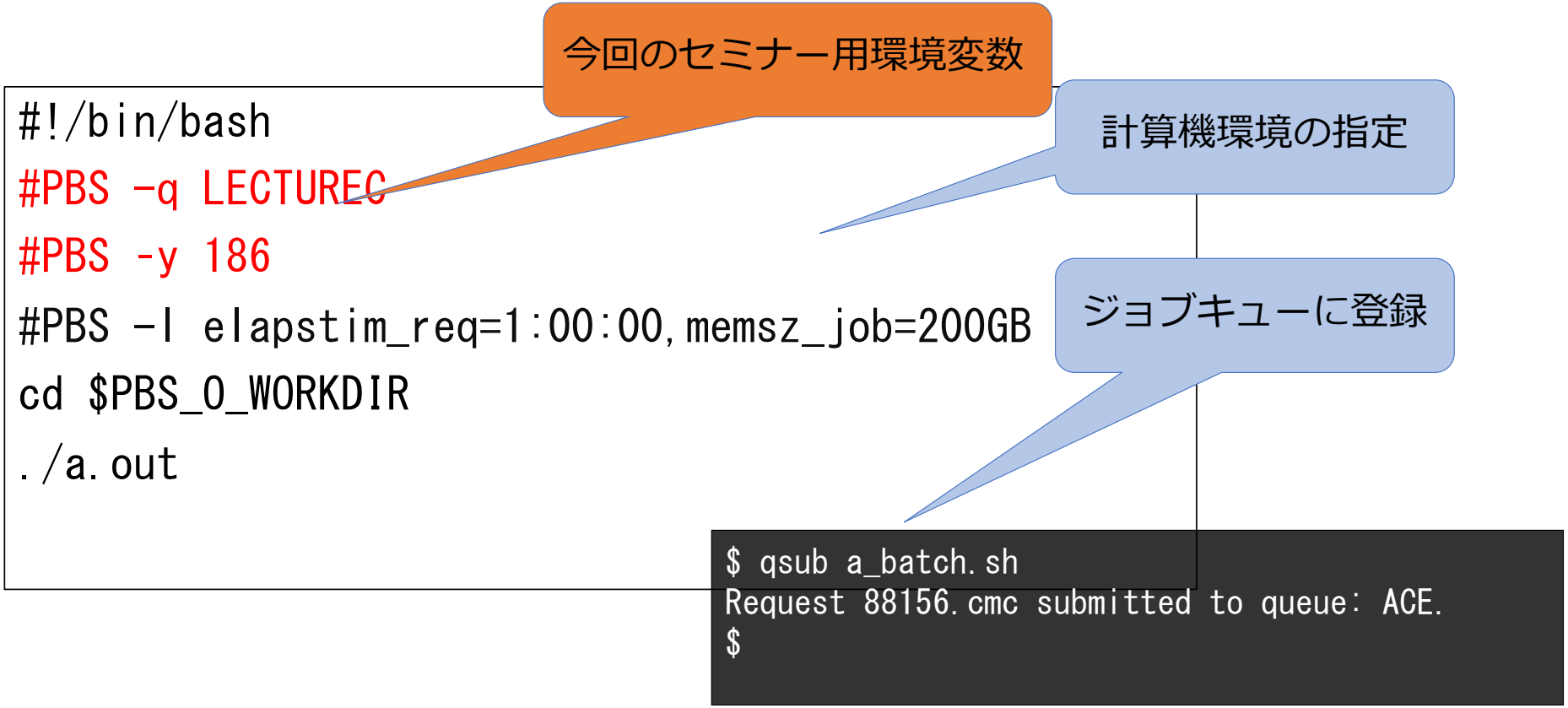
供用コンピュータ・クラスタ



ジョブ投入
バッチキューシステム



qsubでジョブ投入


今回のセミナー用環境変数
計算機環境の指定
ジョブキューに登録
#!/bin/bash
#PBS -q LECTUREC
#PBS -y 186
#PBS -l elapstim_req=1:00:00,memsz_job=200GB
cd $PBS_O_WORKDIR
./a.out
$ qsub a_batch.sh
Request 88156.cmc submitted to queue: ACE.
$

SQUID固有のディレクトリの説明

- ホームディレクトリ：/sqfs/home/k6aXXX
  - フロントエンド，計算ノードからアクセス可能.
- ワークディレクトリ：/sqfs/work/kosyuXXX/k6aXXX
  - フロントエンド，計算ノードからアクセス可能．計算ノードからのファイルアクセスはこちらを推奨

セミナー用ファイル

/system/lecture/seminar/R
+- airline
| +- airlinemid.csv : 航空データ (3.9GB)
| +- airlinenew.csv : オリジナル航空データ (68GB)
|
+- bashrc.sample : .bashrcのサンプル
+- Rmpi.install.cmd : Rmpiをインストールするコマンドサンプル
|
+- R4_FF
| +- airlineBM.R : bigmemory用サンプルスクリプト
| +- airlineFF.R : ff用サンプルスクリプト
| +- airline.R : 通常読み込み用サンプルスクリプト
| +- r4BM.sh : bigmemory用ジョブスクリプト
| +- r4FF.sh : ff用ジョブスクリプト
| +- r4NO.R : 通常読み込み用ジョブスクリプト
+- R4_Rmpi
   +- hello.R : Rmpiを用いたHello Worldスクリプト
   +- r4hello.sh : Hello World用ジョブスクリプト

Rをフロントエンドで動かしてパッケージをインストールしてみよう

```
[k6aXXX@squidhpc1 ~]$ R

R version 4.0.3 (2020-10-10) -- "Bunny-Wunnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

...

> install.packages("rbenchmark", repos="http://cran.ism.ac.jp")
> install.packages("ff", repos="http://cran.ism.ac.jp")
...
> install.packages("Rcpp", repos="http://cran.ism.ac.jp")
> install.packages("bigmemory", repos="http://cran.ism.ac.jp")
...
```

Rmpiをインストールしてみよう

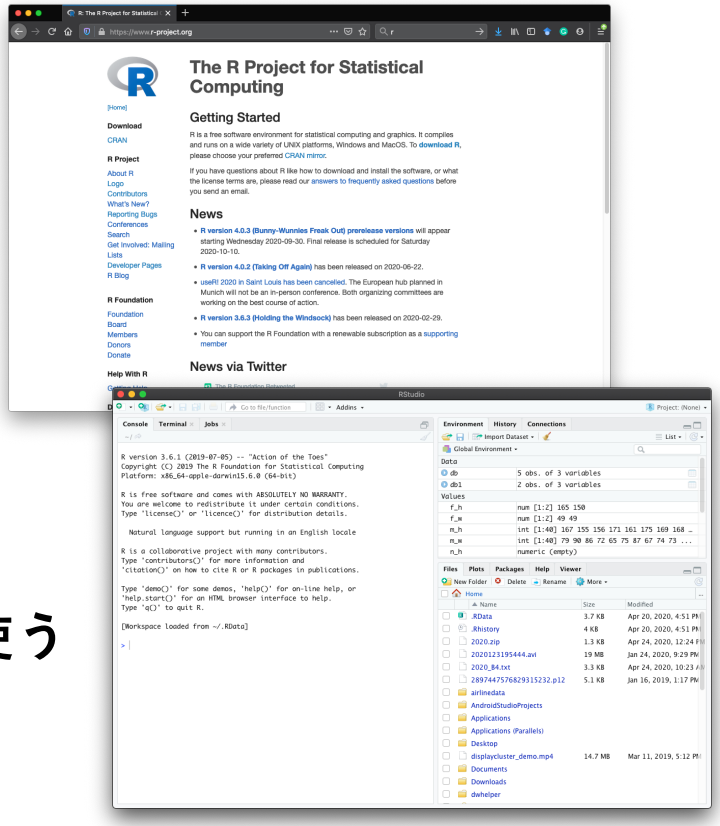
- Rでインストール
- 他のライブラリと違いオプションが長い．．．

```
[w6a009@squidhpc3 ~]$ R

R version 4.0.3 (2020-10-10) -- "Bunny-Wunnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

...

> install.packages("Rmpi", repos="http://cran.ism.ac.jp", configure.args = paste("--with-
mpi=/system/apps/rhel8/cpu/intel/intel2020u4/2020update4/compilers_and_libraries_2020.4.304/linux/mp
i/intel64 --with-Rmpi-type=OPENMPI"))

...
```

Rの特徴（え？このタイミング？）

- 統計のソフトウェアスイート
- 対話環境，IDEなどがあって使いやすい
- スクリプト実行もできちゃう
- フリーソフト（Sとは違う）

- 欠点
  - **オブジェクトは値渡しなのでメモリを大量に使う**
  - **並列環境が苦手（できなくはない**

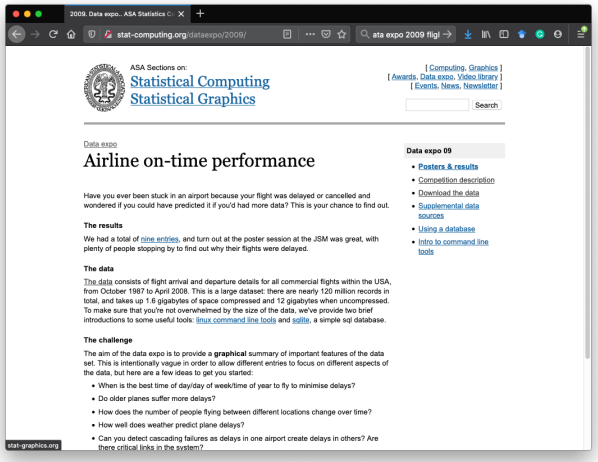


大容量ファイルを扱うパッケージ

- ff : ディスク上の大容量データのメモリ効率の高い格納と高速アクセス機能
  - 大容量のファイルを開いた場合，Rのメモリ領域に展開せずに，カラム数に応じた一時ファイルを作る
  - メインメモリ以上のデータを扱うことができる
  - データのアクセスごとに一時ファイルを参照しに行く（ただし参照方法よってはオンメモリにする場合もある
- bigmemory : 共有メモリとマップファイルを用いた大規模マトリックス管理パッケージ
  - 大容量のファイルを開いた場合，backing fileという一時ファイルを作る
  - あとdescriptor fileというマップファイルを作る
  - メインメモリ以上のデータを扱うことができる
  - ただファイルの読み込みはread.table()と大して変わらない

今回使うデータ

- フリーで利用できる定番，航空データを利用
  - 1987年-2015年の全米のフライトデータ
  - http://stat-computing.org/dataexpo/2009/
  - 全データで68GB
  - これを使おうと思ったんだけど，セミナーの時間内に終わらないので，短くしたものを用意しました．
    - 全データから10,000,000行抜き出したデータ3.9GB



普通に読み込んでみる

アカウント名に変更

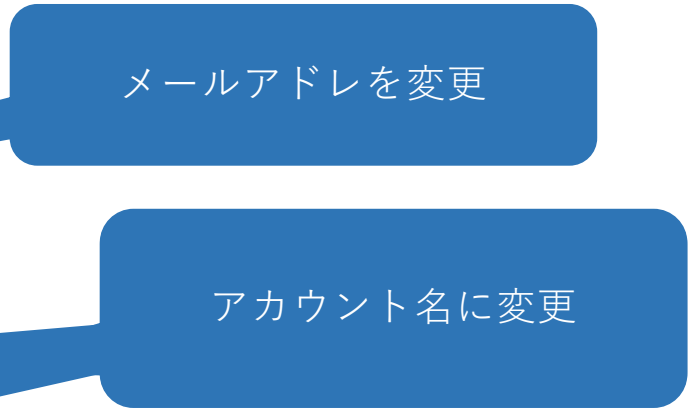
airline.R

```
library(rbenchmark)
start_time <- Sys.time()
air <- read.table("/sqfs/work/kosyuXXX/k6aXXX/airline/airlinemid.csv", header=T, sep=",")
Sys.time() - start_time

test.fx <- function() {
  print(mean(air$Distance))
}
rbenchmark::benchmark(test.fx())
end_time <- Sys.time()
end_time - start_time
```

ジョブスクリプト

r4NO.sh

```
#!/bin/bash
#PBS -q LECTUREC
#PBS -y 186
#PBS -l elapstim_req=0:10:00,cpunum_job=24
#PBS -M kido@cmc.osaka-u.ac.jp
#PBS -m b
#PBS -b 1
#PBS -v R_LIBS=/sqfs/work/kosyuXXX/k6aXXX/R_lib
cd $PBS_O_WORKDIR
Rscript /sqfs/work/kosyuXXX/k6aXXX/R4_FF/airline.R
```




```
[k6a145@squidhpc4 R4_Rmpi]$ qsub r4hello.sh
Request 579113.sqd submitted to queue: LECTUREC.
[k6a145@squidhpc4 R4_Rmpi]$ qstat
RequestID       ReqName  UserName Queue     Pri STT S   Memory      CPU   Elapse R H M Jobs
--------------- -------- -------- -------- ---- --- - -------- -------- -------- - - - ----
579113.sqd      r4hello. k6a145   LECTUREC    0 RUN -   27.35M     0.14        2 Y Y Y    2
```

bigmemoryを使ってみる

airlineBM.R

```
library(bigmemory)
library(rbenchmark)
start_time <- Sys.time()
air <- read.big.matrix("/sqfs/work/kosyuXXX/k6aXXX/airline/airlinemid.csv", header=T,
backingfile="air.bm", type="integer", descriptorfile="air.bm.desc",
backingpath="/sqfs/work/kosyuXXX/k6aXXX/airline/")
Sys.time() - start_time


test.fx <- function() {
  print(mean(air[,"Distance"]))
}
rbenchmark::benchmark(test.fx())
end_time <- Sys.time()
end_time - start_time
```

ffを使ってみる

airlineFF.R

アカウント名に変更

```
library(ff)

library(rbenchmark)

start_time <- Sys.time()

air <- read.csv.ffdf(file="/sqfs/work/kosyuXXX/k6aXXX/airline/airlinemid.csv", header=TRUE,
colClasses=c(Year="integer",Quarter="integer",Month="integer",DayofMonth="integer",DayOfWeek="integer",FlightDate="factor",UniqueCarrier="factor",AirlineID="factor",Carrier="factor",TailNum="factor",FlightNum="factor",OriginAirportID="factor",OrjginAirportSeqID="factor",OriginCityMarketID="factor",Origin="factor",OriginCityName="factor",OriginState="factor",OriginStateFips="factor",OriginStateName="factor",OriginWac="factor",DestAirportID="factor",DestAirportSeqID="factor",DestCityMarketID="factor",Dest="factor",DestCityName="factor",DestState="factor",DestStateFips="factor",DestStateName="factor",DestWac="factor",CRSDepTime="factor",DepTime="factor",DepDelay="factor",DepDelayMinutes="factor",DepDel15="factor",DepartureDelayGroups="factor",DepTimeBlk="factor",TaxiOut="factor",WheelsOff="factor",WheelsOn="factor",TaxiIn="factor",CRSArrTime="factor",ArrTime="factor",ArrDelay="factor",ArrDelayMinutes="factor",ArrDel15="factor",ArrivalDelayGroups="factor",ArrTimeBlk="factor",Cancelled="factor",CancellationCode="factor",Diverted="factor",CRSElapsedTime="factor",ActualElapsedTime="factor",AirTime="factor",Flights="factor",Distance="double",DistanceGroup="factor",CarrierDelay="factor",WeatherDelay="factor",NASDelay="factor",SecurityDelay="factor",LateAircraftDelay="factor",FirstDepTime="factor",TotalAddGTime="factor",LongestAddGTime="factor",DivAirportLandings="factor",DivReachedDest="factor",DivActualElapsedTime="factor",DivArrDelay="factor",DivDistance="factor",Div1Airport="factor",Div1AirportID="factor",Div1AirportSeqID="factor",Div1WheelsOn="factor",Div1TotalGTime="factor",Div1LongestGTime="factor",Div1WheelsOff="factor",Div1TailNum="factor",Div2Airport="factor",Div2AirportID="factor",Div2AirportSeqID="factor",Div2WheelsOn="factor",Div2TotalGTime="factor",Div2LongestGTime="factor",Div2WheelsOff="factor",Div2TailNum="factor",Div3Airport="factor",Div3AirportID="factor",Div3AirportSeqID="factor",Div3WheelsOn="factor",Div3TotalGTime="factor",Div3LongestGTime="factor",Div3WheelsOff="factor",Div3TailNum="factor",Div4Airport="factor",Div4AirportID="factor",Div4AirportSeqID="factor",Div4WheelsOn="factor",Div4TotalGTime="factor",Div4LongestGTime="factor",Div4WheelsOff="factor",Div4TailNum="factor",Div5Airport="factor",Div5AirportID="factor",Div5AirportSeqID="factor",Div5WheelsOn="factor",Div5TotalGTime="factor",Div5LongestGTime="factor",Div5WheelsOff="factor",Div5TailNum="factor"))

Sys.time() - start_time


test.fx <- function() {

  print(mean(air[,"Distance"]))

}

rbenchmark::benchmark(test.fx())

end_time <- Sys.time()

end_time - start_time
```

比較

- read.table
  - ファイルオープン： 3.054893 mins
  - データアクセス（１列の読み）：1.922 mins
- bigmemory
  - ファイルオープン： 7.149671 mins
  - データアクセス（１列の読み）： 1.741 mins
- ff
  - ファイルオープン： 4.171675 mins
  - データアクセス（１列の読み）：6.906 mins

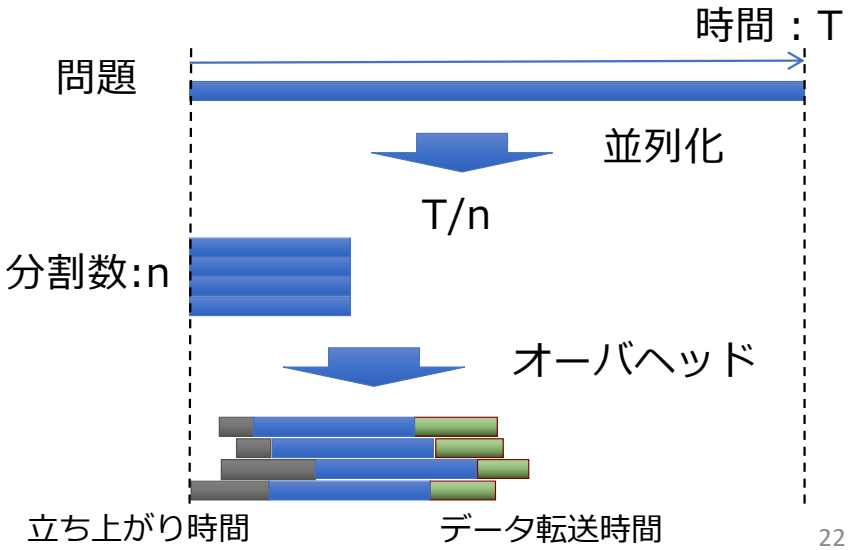
- airlinemid.csv
- 3.9 GB
- 行：10,000,000

一桁GBであればSQUIDでは普通に使える
より大規模になるとffがいいかも

並列プログラミングとは

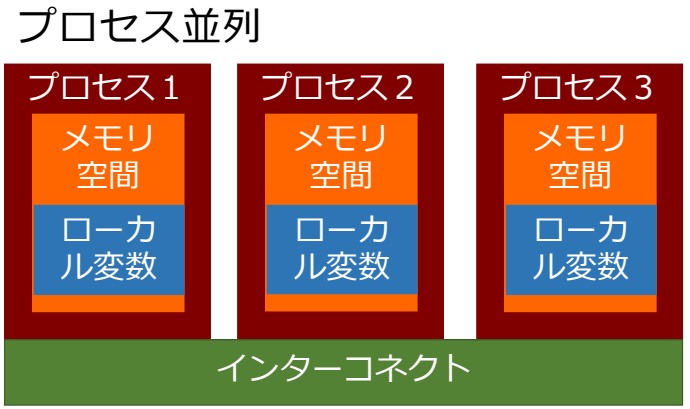
- 逐次処理の問題、プログラム（実行時間：T）をnに分割し、 n 台のプロセッサ（or 計算機）でT/n時間にする
- プロセッサに割り当てられるタスクは独立
- 並列化できる問題はデータに依存性のない問題のみに限られる

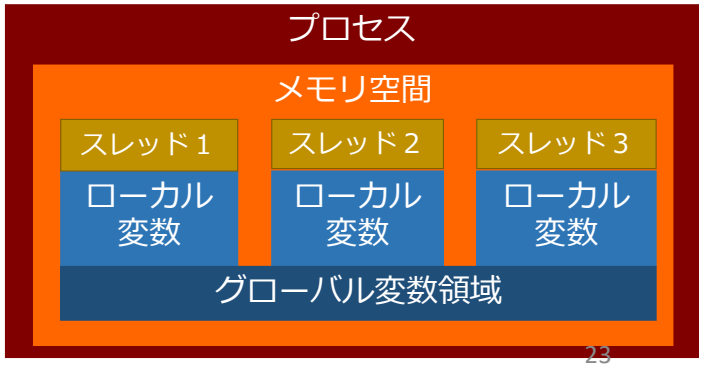
- 通信によるオーバヘッド
  - 立ち上がり時間
  - データ転送時間



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)
  - RだとRmpi
- 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





Rで並列処理 in SQUID

hello.R

```
library(Rmpi)

id <- mpi.comm.rank(comm = 0)
np <- mpi.comm.size(comm = 0)
hostname <- mpi.get.processor.name()

msg <- sprintf("Hello world from process %03d of %03d, on host %s¥n",id, np, hostname)
cat(msg)

mpi.finalize()
```

MPIを使う場合のジョブスクリプト

r4hello.sh

```
#!/bin/bash
#PBS -q LECTUREC
#PBS -y 186
#PBS --group=kosyu145
#PBS -l elapstim_req=0:10:00
#PBS -l cpunum_job=38
#PBS -l memsz_job=200GB
#PBS -M kido@cmc.osaka-u.ac.jp
#PBS -m b
#PBS -b 2
#PBS -T intmpi
#PBS -v R_LIBS=/sqfs/work/kosyu145/k6a145/R_lib
module load BaseCPU/2021
module load BaseR/2021
cd $PBS_O_WORKDIR

mpirun $NQSV_MPIOPTS -np 76 Rscript /sqfs/work/kosyu145/k6a145/R4_Rmpi/hello.R
```

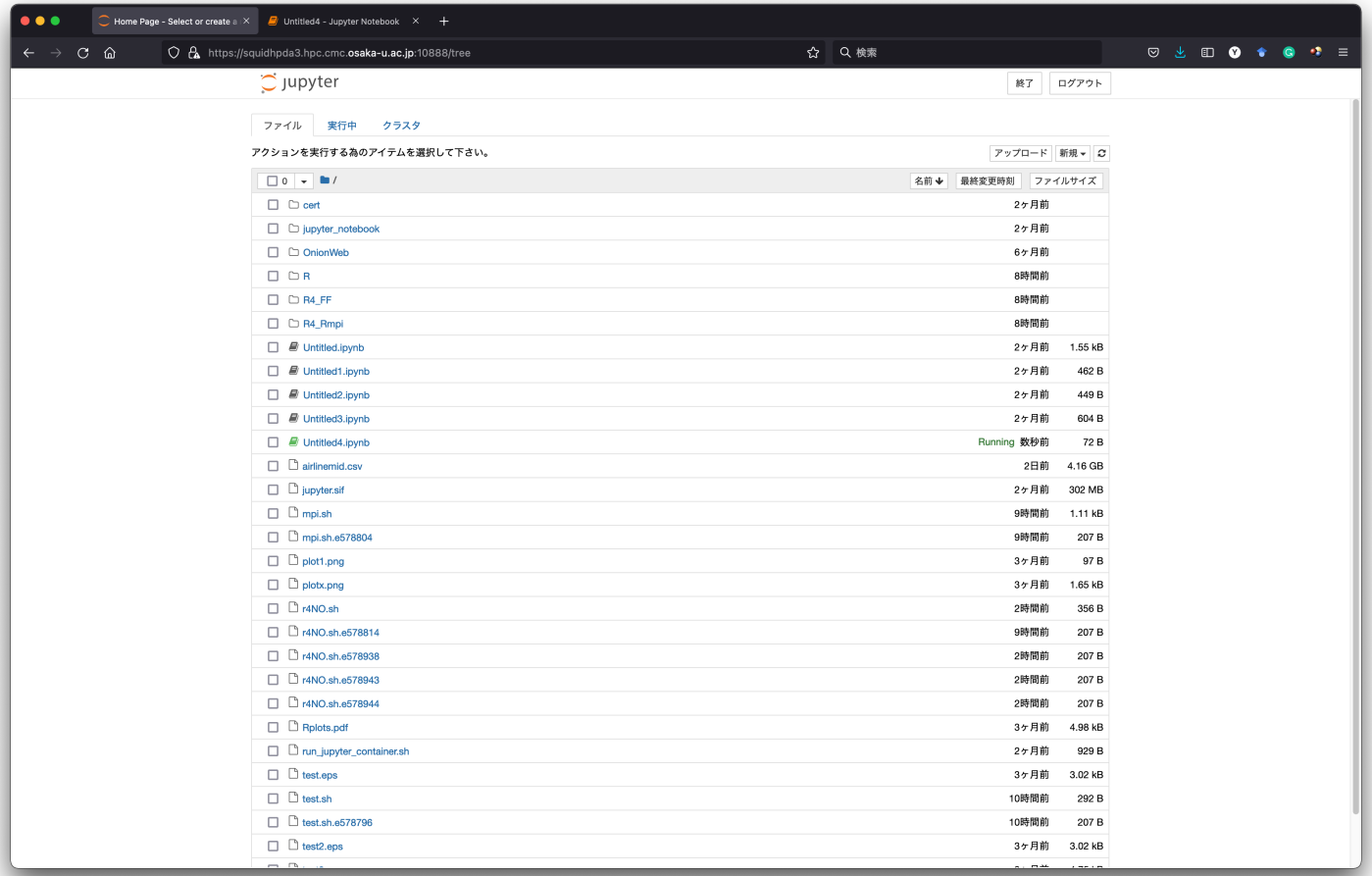
ノードあたりの使用コア数

ノード数

総並列数

おまけ

- squidhpda（高性能データ解析基盤）でjupyter notebookが使える
- ですがRのカーネルが．．．



まとめ

- SQUIDでRは使える
- 大規模データもあつかえる
  - ff, bigmemoryパッケージを使えばより大規模なデータを使える
- 並列処理
  - 今の所バルクジョブは可能
- 講習会終了後
  - LECTURECキューはなくなるので，SQUIDキュー，DBGキューに置き換えて使用可能（1週間