

生物情報解析システムの使い方

ゲノムインフォマティクストレーニングコース 準備編

生物情報解析システムの紹介

Computer system for biological information analysis



分散処理計算機クラスタ
SGI Rackable server C2112-4RP
Intel Xeon (2.8GHz) 20core/node
96GB/node Memory, 40node, 800core



高速ファイルサーバ
DDN SFA7700
Lustre file system : 480TB

生物情報解析システムの構成

大容量データストレージ
・解析システム (LDAS)

生物情報解析システム (BIAS)

ホスト名 Idas-smp

共有メモリ計算サーバ
4TB mem, 80 cores

大容量
ストレージ
720TB

高速ファイルサーバ
480 TB

FDR Infiniband switch

分散処理計算機クラスタ
800cores 4.8GB mem/core

ホスト名
node01-40

ログインノード
384GB mem

ホスト名 bias4

データベースサーバ
512GB mem

SSD
3TBx1

PC端末、プリンタ
(GeneSpring, MATLAB)

研究支援システム

10GEスイッチ

QDR Infiniband
switch

catl catm cats1 cats2

大容量
ディスク
450TB

大容量
ディスク
192TB

大容量
ディスク
90TB

旧システム (買取機分)

生物情報解析システム

基本的な使い方：ログイン先

bias4.nibb.ac.jp

- 基生研外の方は、ユーザーアカウントの申請が必要
<http://www.nibb.ac.jp/cproom/global/appli/index.html>
- sshで接続する (telnetなどは利用できません)
- ログインノード上での作業は、プログラムの作成やファイル管理などの軽い処理にとどめ、大きな計算処理の実行はジョブ管理システム (SGE) を介して行う
- 正確なマシン名は、bias4-login.nibb.ac.jp だが、ログイン時には -login を省略可

ログイン方法：Macユーザ

- ・ ターミナル 上で

```
$ ssh username@bias4.nibb.ac.jp
```

- ・ と入力してリターン

username@bias4.nibb.ac.jp's password:

- ・ と出たらパスワードを入力してリターン

- 画面には何も出ません！ *****等もなし！

```
$ ssh username@bias4.nibb.ac.jp
```

The authenticity of host 'bias4.nibb.ac.jp (133.48.33.122)' can't be established.

RSA key fingerprint is 7b:94:9a:36:ac:60:ae:a0:14:2a:7c:0f:3c:bc:fe:24.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added 'bias4.nibb.ac.jp' (RSA) to the list of known hosts.

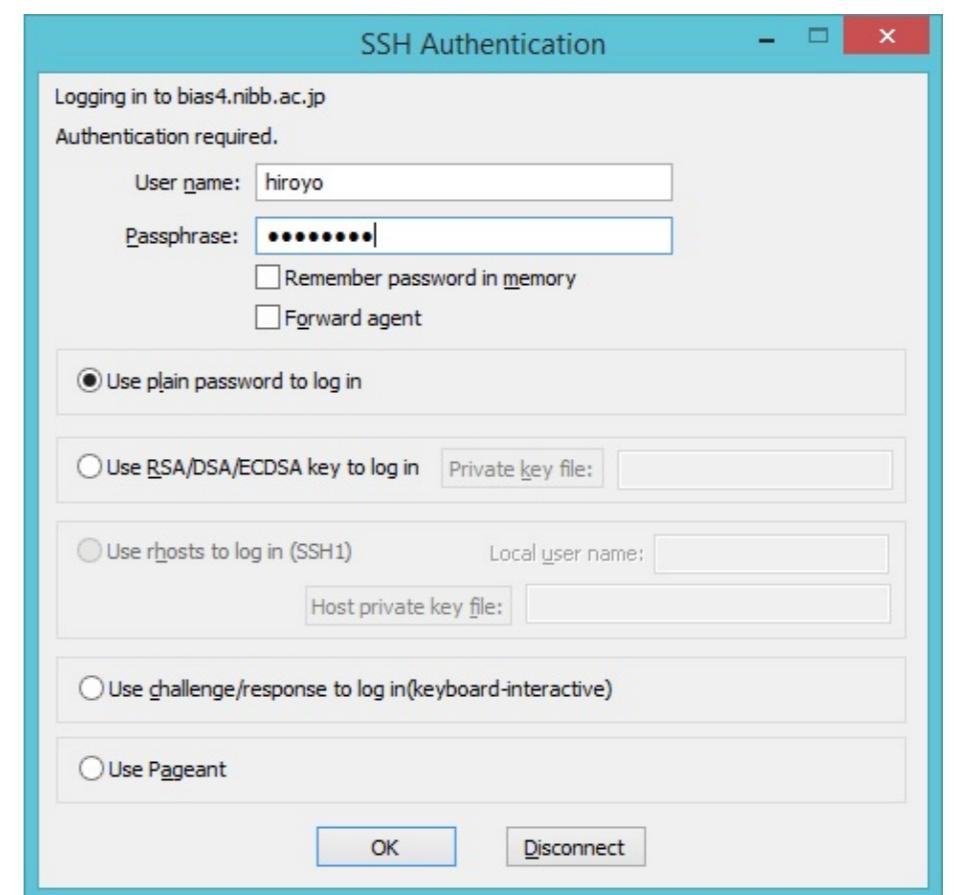
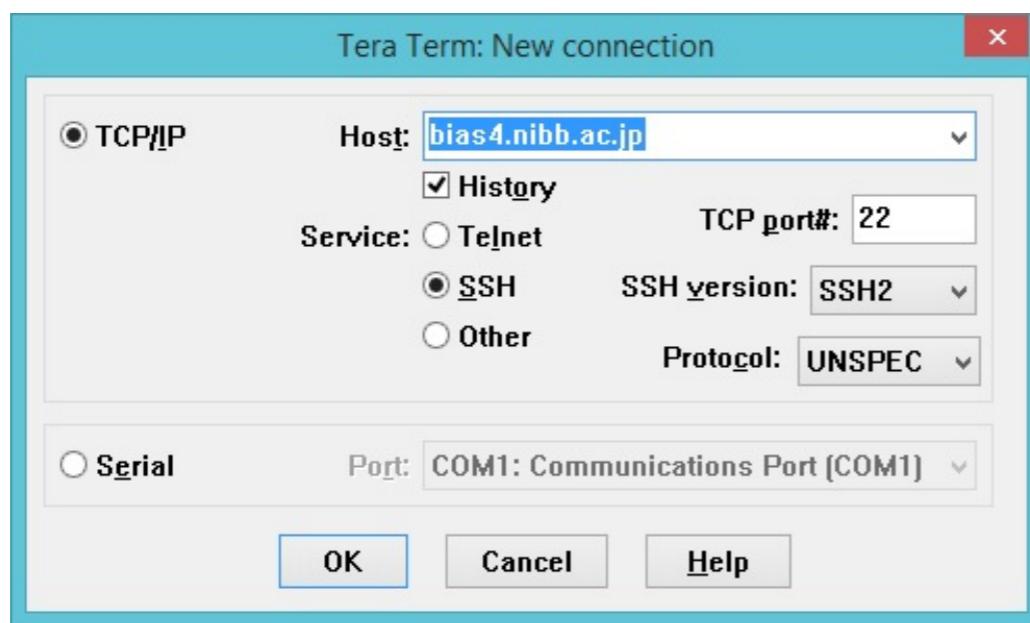
username@bias4.nibb.ac.jp's password:

Last login: Fri Jan 17 15:23:05 2017 from bigfox-01.nibb.ac.jp

[*username*@bias4-login ~]\$

ログイン方法：Windowsユーザ

- TeraTermProをインストール→
<http://ttssh2.sourceforge.jp/>
 - Host: bias4.nibb.ac.jp
 - Service: SSH, TCP port#: 22, SSH version: SSH2, Protocol: UNSPEC
 - User name: ユーザ名
 - Passphrase: パスワード



生物情報解析システム：ログインしてみましょう

```
$ ssh username@bias4.nibb.ac.jp
```

The authenticity of host 'bias4.nibb.ac.jp (133.48.33.122)' can't be established.

RSA key fingerprint is
7b:94:9a:36:ac:60:ae:a0:14:2a:7c:0f:3c:bc:fe:24.

最初の一回だけyes入力

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added 'bias4.nibb.ac.jp' (RSA) to the list of known hosts.

username@bias4.nibb.ac.jp's password: #パスワード入力

Last login: Fri Jan 17 15:23:05 2014 from bigfox-01.nibb.ac.jp

[*username*@bias4-login ~]\$ pwd

[*username*@bias4-login ~]\$ ls

- pwd でログイン後のディレクトリを、
- ls でディレクトリの内容を確認

生物情報解析システムからログアウト

```
[username@bias4-login ~]$ exit  
$
```

- exitコマンドで手元のマシンに戻る

生物情報解析システムにデータをコピー

- scp コマンドで離れたUNIXマシンにデータをコピーできる
- scp (secure copy) 暗号化して送信
- コピーが終わったらbias4にログインしてファイルを確認

```
$ cd ~/data  
$ scp -r 7_bias bias4.nibb.ac.jp:  
username@bias4.nibb.ac.jp's password:  
  
$ ssh username@bias4.nibb.ac.jp  
username@bias4.nibb.ac.jp's password:  
[username@bias4-login ~]$ ls
```

scp コマンド



- 手元のMac（ローカル）から生物情報解析システム（リモート）へ

```
$ scp コピーしたいファイル username@リモートホスト名:コピー先のパス
```



- リモートからローカルへ

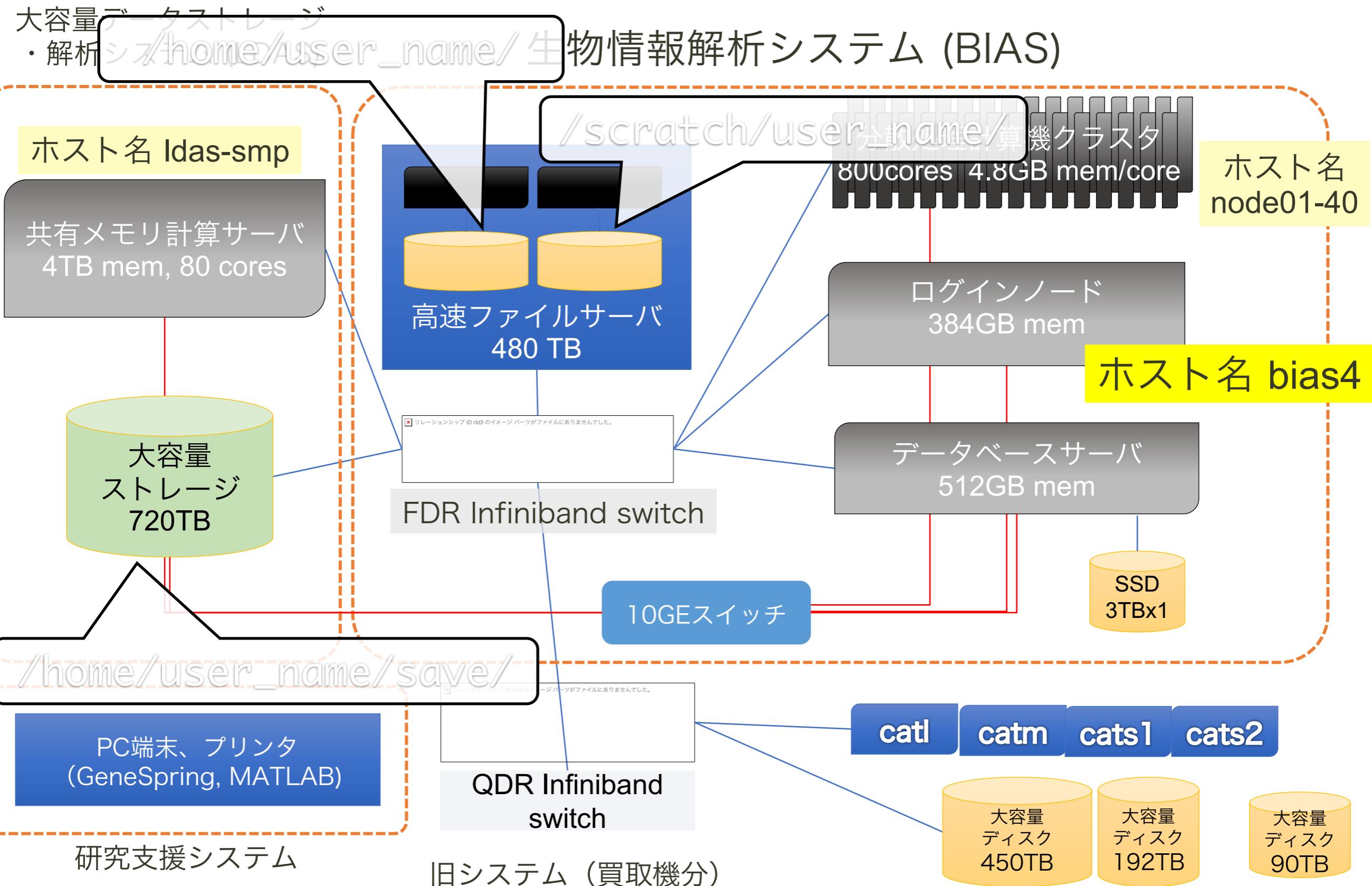
```
$ scp username@リモートホスト名:コピーしたいファイルのパス ローカルのパス
```

- コピー先で「：」以降を省略するとホームディレクトリになる

ディスクの使い分け

- ・ 全てのディスクはどのマシンから同じに見えるようにマウントされています
- ・ ホームディレクトリ：高速ファイルサーバ上 /home/user_name/
 - ✓ 一人あたり容量制限 1.5TB
- ・ save領域：大容量ストレージ上 /home/user_name/save/
 - ✓ 一人あたり容量制限 5TB
- ・ scratch領域：高速ファイルサーバ上 /scratch/user_name/
 - ✓ 容量制限なし
- ・ /scratch 下のデータは一ヶ月で自動消滅します
 - 大事なものは /home 下に戻しましょう

生物情報解析システムの構成



分子生物学 アプリケーション

- ・ ほとんどのアプリケーションは、/bio/bin 内にあり、どのマシンからでも同様に使用可能
- ・ /bio/bin への実行パスもログイン時に通っています。
- ・ 必要なものは当室で隨時インストールします。
- ・ 詳細は生物情報解析システムwiki をご覧ください
- ・ <http://www.nibb.ac.jp/cproom/wiki/index.php>

分子生物学データベース

- BLAST用データベースへのパス
 - /bio/db/blast/db
- フラットファイルへのパス
 - /bio/db/ideas
- このほか、KEGG データベース IlluminaGenomes、interproscan用dbも利用可能です。

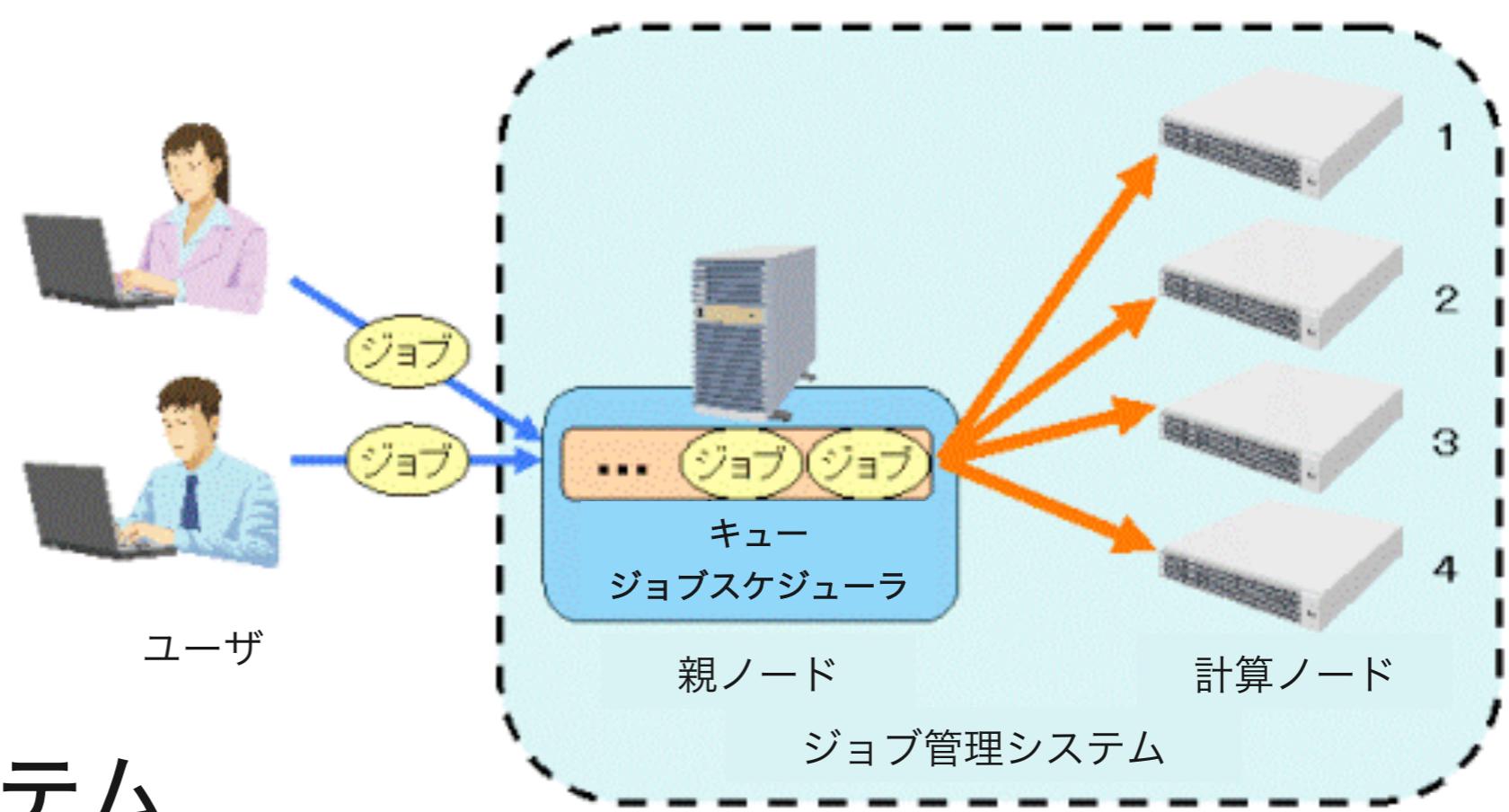
生物情報解析システム

SunGridEngine 利用方法

大型計算機を有効に使うには

複数の人間が同じ計算機群を使いたい...
どの計算機/CPUが空いてるか?
平等に使うには?

- ・ ユーザのジョブを
 - 実行された順番に
 - 空いている計算機に割り振ってくれる



- ・ ジョブ管理システム

ジョブ管理システム

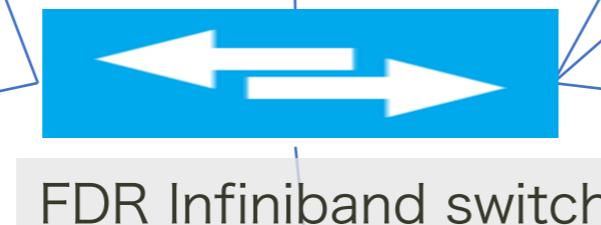
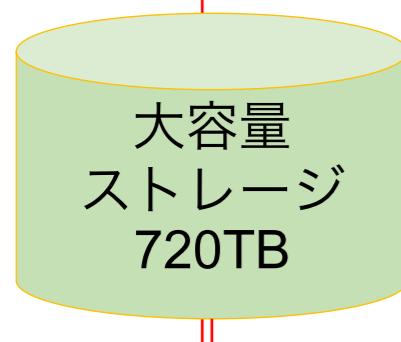
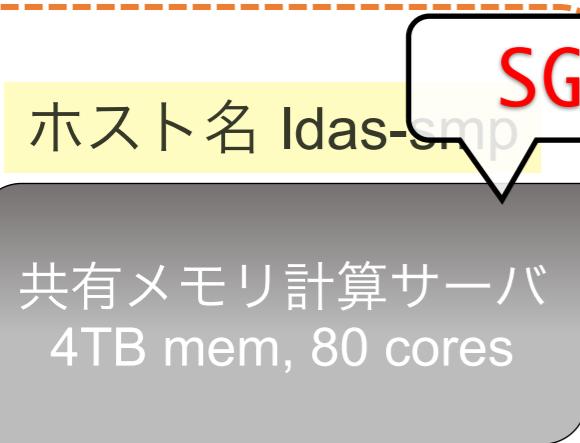
- 親ノードが、複数ある計算機から資源の割り当てを自動で行い、効率を上げる
- ユーザは親ノードにジョブを投げるだけ（親ノードの名前すら知らなくてもよい）
- 生物情報解析システム上でのデータ解析は基本ジョブ管理システムを使うこと
 - bias4.nibb.ac.jp はパワーがないので、皆がbias4上で解析を行うとすぐ倒れます
- SunGridEngine (SGE)

生物情報解析システムの構成

大容量データストレージ
・解析システム (LDAS)

生物情報解析システム (BIAS)

SGE



ホスト名
node01-40

ログインノード
384GB mem

ホスト名 bias4

データベースサーバ
512GB mem



10GEスイッチ

PC端末、プリンタ
(GeneSpring, MATLAB)

研究支援システム

QDR Infiniband
switch

SGE

catl

catm

cats1

cats2



旧システム (買取機分)

作業ディレクトリ

bias4.nibb.ac.jp 上での ~/7_bias

```
$ cd ~/7_bias  
$ ls
```

ジョブ管理システム:SGEの利用

- 実行したいコマンドをシェルスクリプト内に記述し、**qsub**コマンド（後述）を用いてジョブ管理システムに実行させる
- シェルスクリプト ex.sh の中身を確認

```
$ less ex.sh
```

```
#!/bin/sh
#$ -cwd
#$ -q gitc
```

```
bowtie2 -x eco -U ecoli.fastq -S ecoli.sam
```

qsub実行！

- ・シェルスクリプトをジョブ管理システム(SGE)に投入：qsub

```
$ qsub scriptfile
```

- ・ex.sh をqsubコマンドで実行

```
$ qsub ex.sh
```

```
Your job 814953 ("ex.sh") has been submitted
```

- ・投入されたジョブはログインノードから分散処理計算機クラスタ内のノードに送られて実行される
- ・標準出力と標準エラー出力の内容がホームディレクトリにファイルとして作られる (-cwd オプションを付けるとqsubしたディレクトリ)
- ・投入したジョブの状況を見るには qstat コマンドを使う

```
$ qstat
```

qsub (bowtie2) 結果の確認

```
$ ls ecoli.sam
```

結果ファイルの確認

```
$ ls ex.sh.*
```

標準出力・標準エラー出力ファイル

```
ex.sh.e814953 ex.sh.o814953
```

```
$ less ex.sh.e814953
```

標準エラー出力ファイルの中身を確認

```
330118 reads; of these:
```

```
330118 (100.00%) were unpaired; of these:
```

```
3364 (1.02%) aligned 0 times
```

```
229054 (69.39%) aligned exactly 1 time
```

```
97700 (29.60%) aligned >1 times
```

```
98.98% overall alignment rate
```

SGEの主なコマンド

qsub <i>script_file</i>	<i>script_file</i> ジョブを投入
qstat	自分のジョブの状態を表示
qstat -u '*'	全ユーザのジョブ状態を表示
qdel <i>job-ID</i>	<i>job-ID</i> のジョブを削除

```
$ qstat
job-ID  prior  name    user   state submit/start at  queue slots ja-task-ID
-----
814953 0.00000 ex.sh   hiroyo r      01/08/2017 14:14:54 small@node04  1
814954 0.00000 job.sh  hiroyo qw     01/08/2017 14:14:54 small@node05  1
814955 0.00000 job.sh  hiroyo qw     01/08/2017 14:14:54 small@node05  1
```

```
$ qdel 814953
hiroyo has deleted job 814953
```

qsub のオプション

- qsub には様々なオプションがあり、シェルスクリプト内で「#\$」に続けて書いておくことで機能を加えることができる

```
$ less ex.sh
```

```
#!/bin/sh
#$ -q gitc          gitc キューを指定
#$ -cwd             qsub したディレクトリに移動してジョブを実行
bowtie2 -x eco -U ecoli.fastq -S ecoli.sam
```

- オプション -cwd を指定しているので、ファイルのパスを付けていない
- -cwd を指定すると標準出力と標準エラー出力の内容はコマンドを実行したディレクトリに作られる

qsub のオプション

オプション	説明
<code>#\$ -o <i>filename</i></code>	標準出力の結果を指定したファイルに保存
<code>#\$ -e <i>filename</i></code>	標準エラー出力の結果を指定したファイルに保存
<code>#\$ -q <i>queue_name</i></code>	キューを指定してジョブを実行
<code>#\$ -cwd</code>	qsubした時のディレクトリに移動してジョブを実行
<code>#\$ -v 環境変数=値</code>	環境変数をジョブに渡す
<code>#\$ -N <i>job_name</i></code>	ジョブ名を指定する
<code>#\$ -s <i>shell_name</i></code>	ジョブスクリプトを指定したシェルで実行
<code>#\$ -a <i>MMDDhhmm</i></code>	ジョブの開始日時を指定
<code>#\$ -l <i>resource_name</i> 値</code>	ジョブが使うリソース量を指定する
<code>#\$ -pe <i>PE_name</i> プロセス数</code>	並列ジョブを実行する場合の環境と並列数の指定
<code>#\$ -t 開始番号-終了番号</code>	アレイジョブを実行

コマンドラインで qsub

- qsub は標準入力から受け取ったコマンドを実行させることもできる
- **\$ qsub ex2.sh** と全く同じ動きをするコマンド

```
$ echo 'bowtie2 -x ./ecoli_genome -U ./ecoli1.fastq  
-S ./ecoli1.sam' | qsub -q small -cwd
```

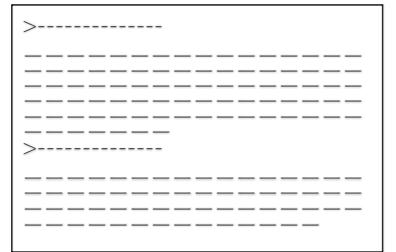
- echo で実行したいコマンドを標準出力に書き出し、パイプで qsub に送っている
- スクリプトで書いていた「#\$」に続けるオプションは、#\$抜きで qsub に続けて指定する。
- スクリプトファイルを用意しなくても qsub 実行できるが、記録の意味でもスクリプトを残しておくことを推奨します。

キュー（ジョブの待ち行列）構成

	分散並列処理型			共有メモリ型			
	分散処理計算機クラスタ			共有メモリ型計算サーバ		cat群	
キュー名	small	medium	large	smps	smpm	smpl	cat
ジョブの特徴	短時間・並列多	中規模	長時間	中メモリ	大メモリ	最大メモリ	denovoアセンブリ用
利用ノード	node01-40	node01-40	node01-40	ldas-smp	ldas-smp	ldas-smp	catl, catm cat1, cat2
最大実行時間 /job	6時間	72時間	no limit	no limit	no limit	no limit	no limit
最大ジョブ数 /キュー	580	200	20	8	4	1	112
最大使用メモリ /ジョブ	4GB	4GB	4GB	500GB	1TB	4TB	512GB/1TB/ 96GB/96GB
利用できるPE	smp, mpi128, mpi256, make	smp, mpi128, mpi256, make	smp, mpi128, make	smp	smp	smp	smp, make

- ・キューを指定しない場合、デフォルトでは「small」で実行される
- ・ユーザあたりジョブ同時実行数は最大 400

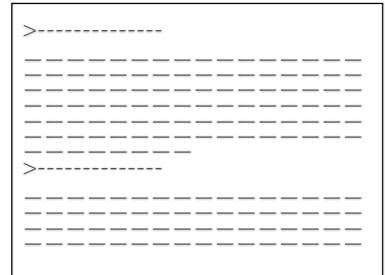
並列化



bowtie2



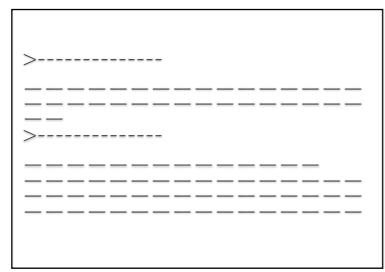
node-01



bowtie2



node-01



bowtie2



node-14



node-01~node40
分散処理計算機クラスタ

- 1台で順次実行していれば時間がかかる計算も、分割して複数台、複数CPUに仕事をさせれば数倍の速度で終わる

アレイジョブ

- 似たようなジョブファイルをいくつも作ったり、qsubを何度も実行するのは面倒
- アレイジョブ機能を使う
- 1つのスクリプトファイルで複数の独立ジョブを実行させる
- #\$ -t 開始番号-終了番号
- 開始 - 終了 でジョブの数を指定
- 変数 \${SGE_TASK_ID} が自動で設定され、この変数をインクリメントしながらスクリプトの内容が回数分実行される
- qsub も1回でよい

アレイジョブ

- test_fastq ディレクトリには、10個の圧縮された fastq.gz ファイルがある
- これらを一度にマッピングできるスクリプトを作成
- bowtie2 は .gz ファイルを直接マッピングできる
- emacs で ex.sh を 改変して作る
- test_fastq 内の 10個のfastq.gz ファイルを読み込み
- 結果は results ディレクトリに書き込み

```
$ ls test_fastq  
$ mkdir results
```

test_fastq ディレクトリの中を確認
結果用ディレクトリを作成

アレイジョブ用スクリプトファイルの作成

```
$ emacs ex.sh
```

emacs で ex.sh を開く

```
#!/bin/sh
#$ -cwd
#$ -q gitc
#$ -t 1-10

bowtie2 -x eco
        -U test_fastq/ecoli.${SGE_TASK_ID}.fastq.gz
        -S results/ecoli.${SGE_TASK_ID}.sam
```

- Ctl-X, Ctl-W で別名 (ex2.sh) を付けて保存

アレイジョブ実行

- 実行とジョブの確認

```
$ qsub ex2.sh
```

```
Your job 6597 ("ex2.sh") has been submitted
```

```
$ qstat
```

job-ID	prior	name	user	state	submit/start at	queue	slots	ja-task-ID
6597	0.5050	ex2	hiroyo	r	01/19/2017 18:04:04	small@node03.local	1	1
6597	0.5050	ex2	hiroyo	r	01/19/2017 18:04:04	small@node06.local	1	2
6597	0.5050	ex2	hiroyo	r	01/19/2017 18:04:04	small@node08.local	1	3
6597	0.5050	ex2	hiroyo	r	01/19/2017 18:04:04	small@node06.local	1	4
6597	0.5050	ex2	hiroyo	r	01/19/2017 18:04:04	small@node06.local	1	5

ジョブID	ジョブ名	ジョブオーナー	実行開始時間	キュー名@実行ホスト名	アレイジョブ番号
優先順位 (全て同じ)		実行 ステータス r : running qw : pending Eqw : failed			

アレイジョブ：結果の確認

- 10個のsamファイルができているはず

```
$ ls results
```