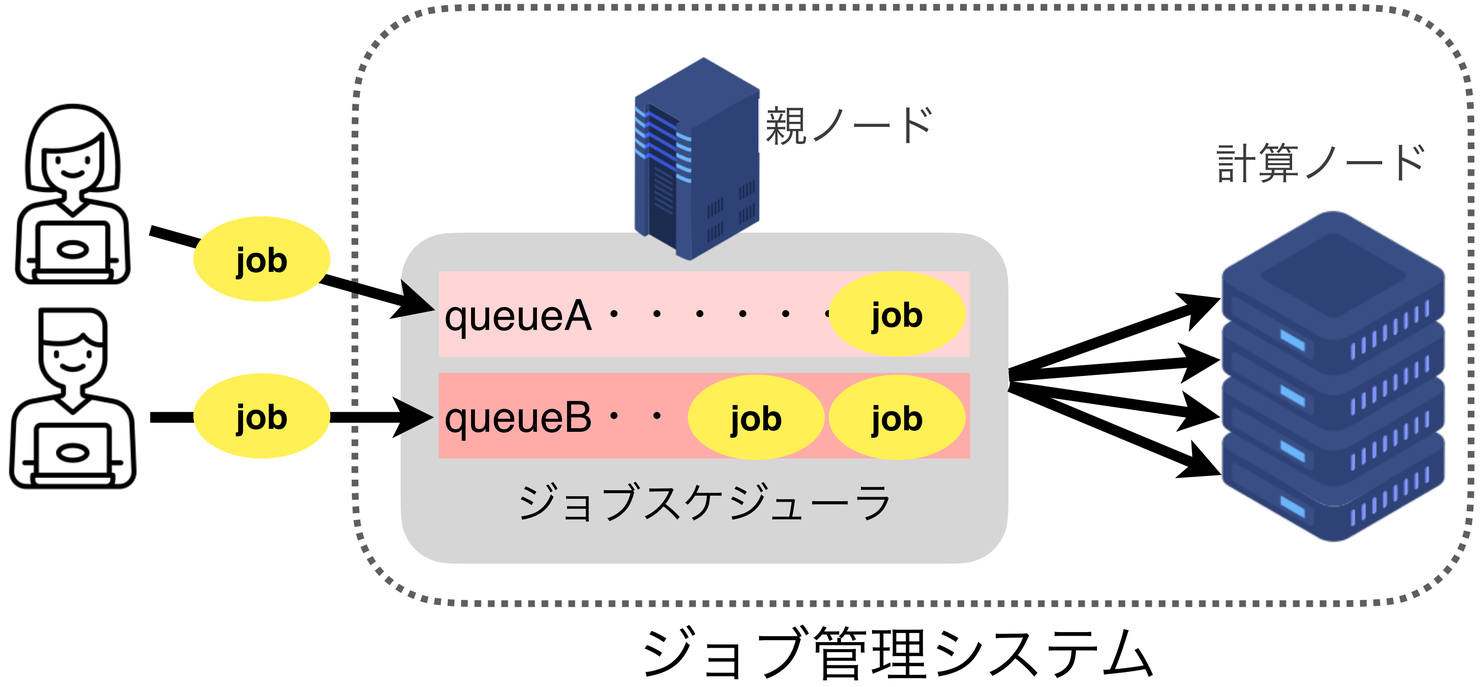
RCCS計算機利用方法
ジョブの実行2

11/Jun/2026 Hiroyo NISHIDE

ジョブ管理システムにおけるリソース指定

- 複数の人間が同じ計算機群を使う
- どのマシン/CPUが空いてるか？ どの計算を優先させるべきか？

- 自動で計算機資源を割り当て効率を上げる



- どれくらい計算機資源（CPU, メモリ）を使いたいかを予約する
- 予約内容はシェルスクリプトの中に書いておく
- 親ノードが内容を読み取り、計算ノードにジョブを割り振る

リソース指定の書き方

- RCCSでは指定されたリソース量によって投入先ノードが変わる
- **#PBS -l** に続けて「：」で区切りながらリソースを指定する

RCCS用スクリプト例

```
#!/bin/bash
#PBS -l select=1:ncpus=12:mpiprocs=1:ompthreads=12
#PBS -l walltime=4:00:00
source /apl/bio/etc/bio.sh
module load diamond
cd ${PBS_O_WORKDIR}
diamond blastp --db spo.dmnd --out sce.tab --outfmt 6 --query sce_prot.fasta
```

- ジョブにおいて使えるメモリ量は、ncpus と jobtype に依存する

リソース指定の書き方

```
#!/bin/bash
#PBS -l select=1:ncpus=12:mpiprocs=1:ompthreads=12
#PBS -l walltime=4:00:00
```

- 必ず指定するリソース
  - ノード数：**select=** 1ノードで実行する
  - ノードあたりのコア数：**ncpus=** 12コアを使う
  - ノードあたりのプロセス数：**mpiprocs=** プロセスは1つ
  - プロセスあたりのスレッド数：**ompthreads=** 12スレッドで実行する
  - 計算時間（時間）：（分）：（秒）：**walltime=** 最大計算時間は4時間

    ncpus = ompthreads x mpiprocs であること

リソース指定の書き方

- 必要に応じて指定するリソース
  - ジョブタイプ ： **jobtype=**
  - 大容量メモリ（256Gb以上）を使いたい場合

7.785Gbx64=504Gbメモリ

```
#!/bin/bash
#PBS -l select=1:ncpus=64:mpiprocs=1:ompthreads=64:jobtype=largemem
```

  - ノードあたりのGPU数：**ngpus=**
  - GPUを使いたい場合

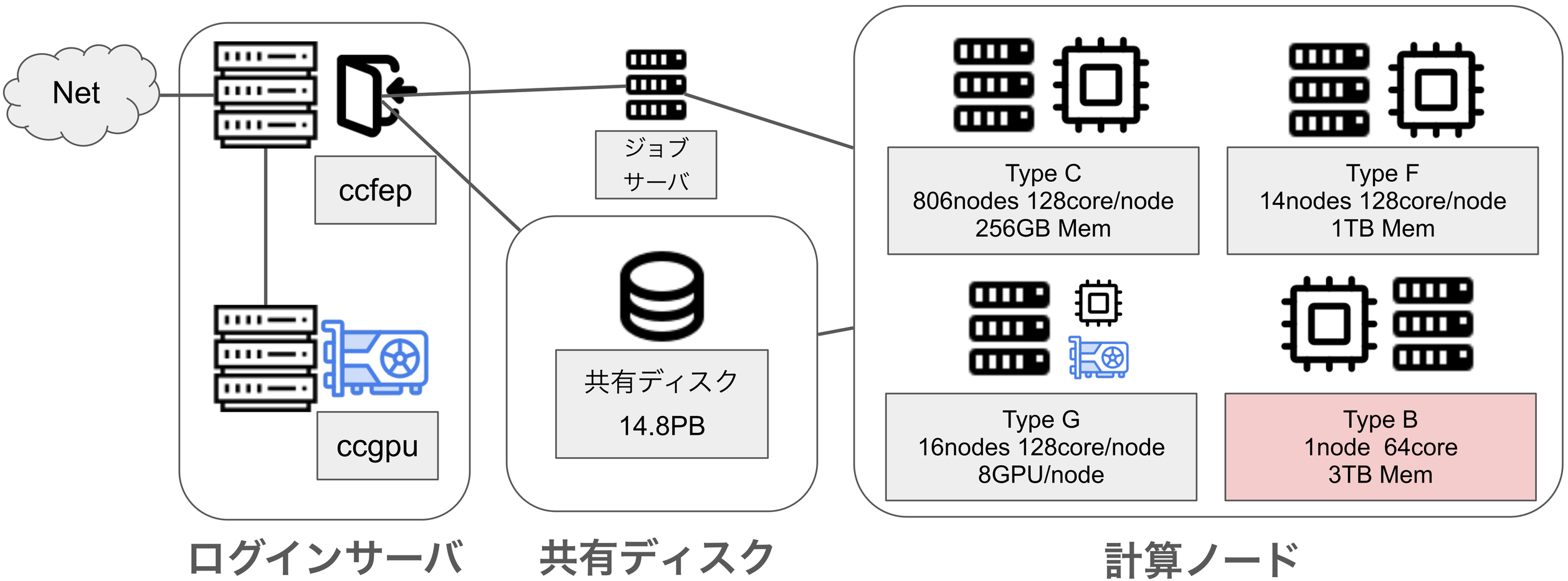
8 GPU, 128コア

```
#!/bin/bash
#PBS -l select=1:ncpus=128:mpiprocs=1:ompthreads=128:ngpus=8
```

RCCS構成図（模式図）

：対話型処理可

：ユーザーログイン不可



参照：https://ccportal.ims.ac.jp/manual/overview

jobtype=　ジョブタイプについて

jobtype	計算ノード	メモリ/コア	利用単位	1ジョブあたりの制限	特徴
largemem (l)	TypeF	7.875 Gb	vnode / ノード	1～14 vnode(s) （64～896コア） （ncpus=64 or 128）	大容量メモリ 1/2ノード(1vnode) 以上占有
vnode (v)	TypeC	1.875 Gb	vnode / ノード	1～50 vnode(s) （64～3,200コア） （ncpus=64 or 128）	1/2ノード(1vnode) 以上占有
core (c)	TypeC	1.875 Gb	コア	1～63 コア （ncpus<64）	64コア以下 総メモリ118Gb以下
gpu (g)	TypeG	1.875 Gb	コア	1～48 GPU, 1～16 コア/GPU (ngpus>0)	GPU利用

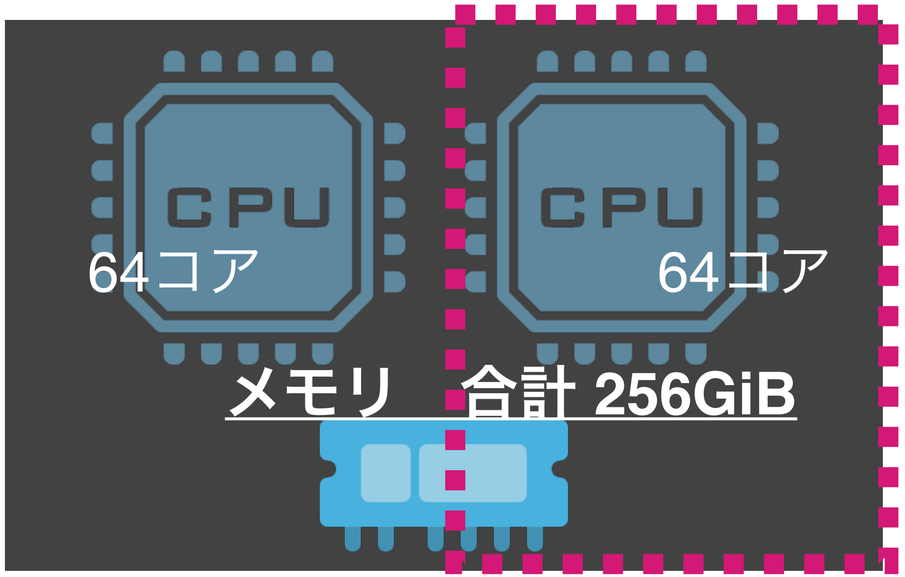
- largemem 以外の jobtype は指定されたリソースより判定可能なため省略可
- **使えるメモリ量は、指定コア数 x 計算ノードタイプのコア当たりのメモリ量**

出典：https://ccportal.ims.ac.jp/manual/queues

ノードタイプとジョブタイプ

TypeC　　806台

jobtype : core / vnode

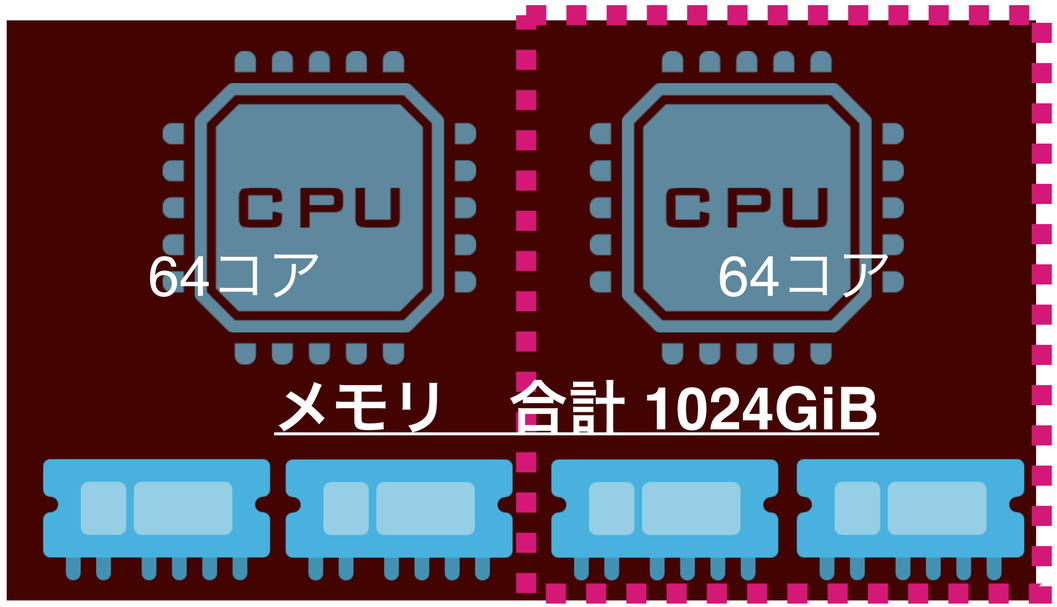


Vノード　（1/2ノード）

jobtype=vnode で使われる単位

TypeF　　14台

jobtype : largemem

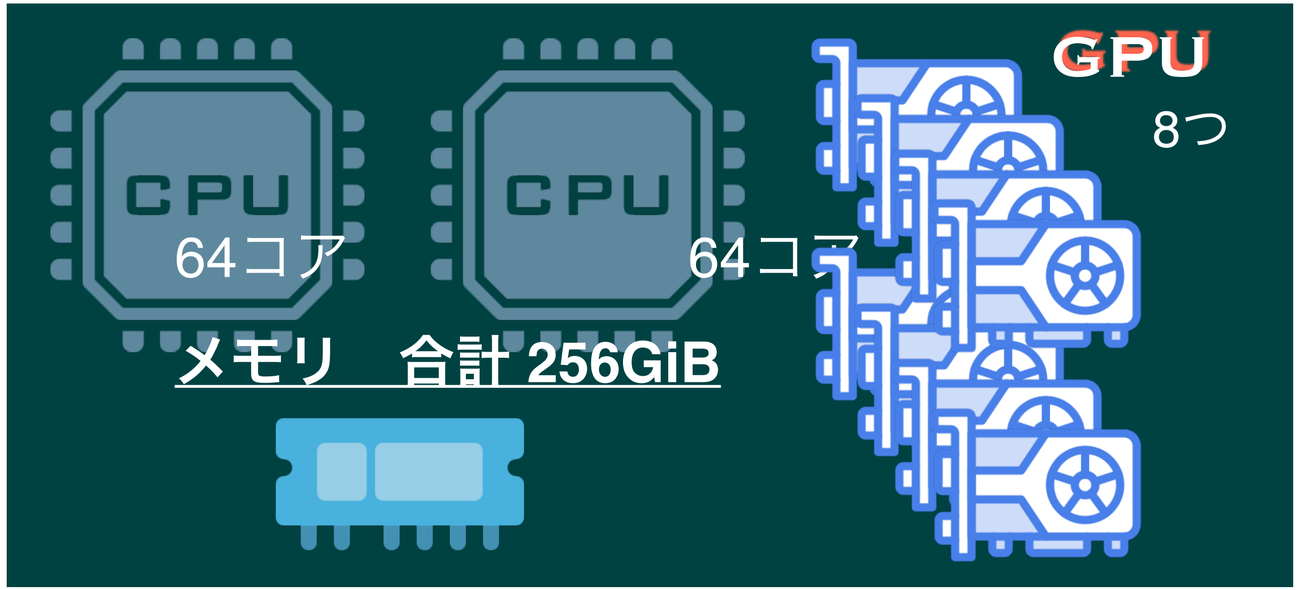


Vノード　（1/2ノード）

ノードタイプとジョブタイプ

TypeG　　16台

jobtype : gpu



1 ノードに

8GPU＋128CPUコア

1GPUあたり16コア

大容量メモリマシンの使い方

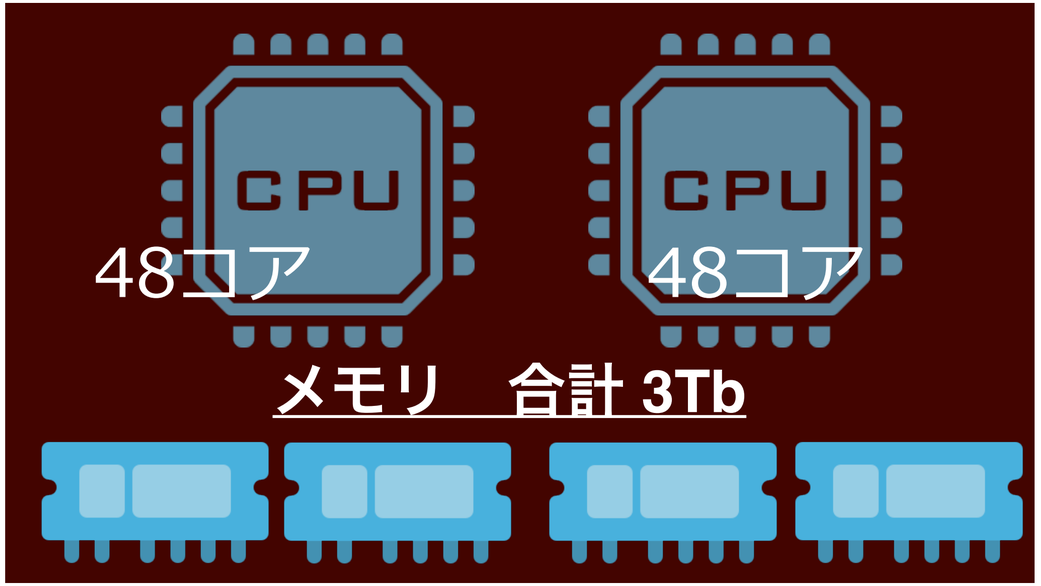
- 最大3Tbのメモリが使える
- 基礎生物学分野で申請したユーザのみ
- 唯一**キュー名**が異なる

（ジョブタイプ指定でなくキュー名を指定することによって利用する）

TypeB　1台

jobtype：nibb (b)

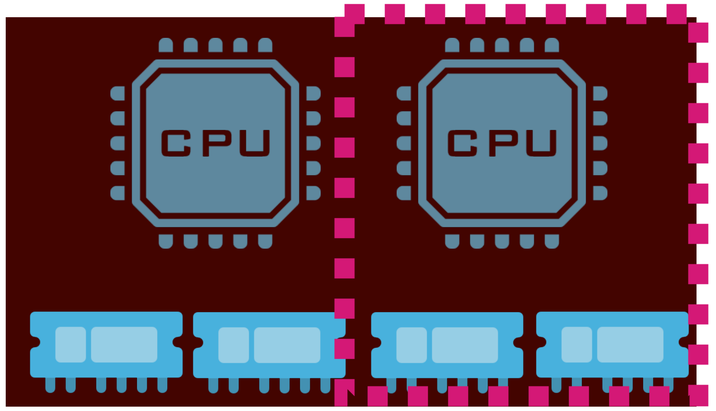
CPU 96コア



大容量メモリマシン用 リソース指定の書き方

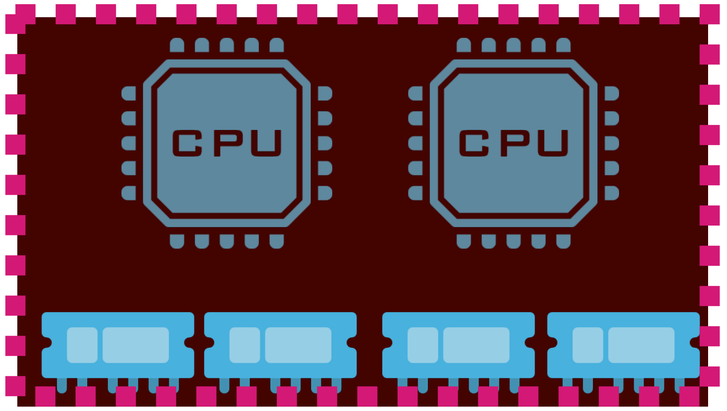
- 1.5 Tb メモリを使う
- システム全体における同時実行数 2

```
#!/bin/bash
#PBS -l select=1:ncpus=48:mpiprocs=1:ompthreads=48
```



- 3 Tb メモリを使う
- システム全体における同時実行数 1

```
#!/bin/bash
#PBS -l select=1:ncpus=96:mpiprocs=1:ompthreads=96
```



大容量メモリマシン用：ジョブ実行

- jsubコマンド： -q に続けてキュー名「HB」を指定して実行

```
$ jsub -q HB my_script.sh
6439875.ccpbs1
```

- jobinfo でジョブの実行状況を確認

```
$ jobinfo
-------------------------------------------------------------------------------
Queue    Job ID Name           Status    CPUs  User/Grp    Elaps     Node/(Reason)
-------------------------------------------------------------------------------
HB(b)     6439875 test.sh        Run        4     ebv/zo0    0:00:20  ccb001
-------------------------------------------------------------------------------
```

CPU点数

jobtype	CPU(ncpus=)	メモリ	CPU キュー係数	GPU キュー係数
largemem	64	504 Gb	60 点 (1vnode) / 1 時間	
	128	1008 Gb	120 点 (2vnode) /1 時間	
vnode	64	120 Gb	45 点 (1vnode) / 1 時間	
	128	240 Gb	90 点 (2vnode) / 1 時間	
core		1.875 Gb/ core	1 点 / (1コア * 1 時間)	
gpu		1.875 Gb/ core	1 点 / (1コア * 1 時間)	60 点 / (1GPU * 1 時間)

queue	CPU (ncpus=)	メモリ	CPU キュー係数
HB	48	1.5Tb	80 点 / 1 時間
HB	96	3Tb	160 点 /1 時間

- 消費点数は実際のジョブ実行時間で計算される。walltimeではない

出典：https://ccportal.ims.ac.jp/manual/queues

参考：

- 分子生物学アプリケーションにおいて、複数プロセス立ち上げや、複数マシンにまたがった計算を行えるものは非常に少ない
  - `select= >1` の指定が有効なアプリケーションはほとんどない
- 複数ノード指定が有効なのは、物理マシンをまたがって並列計算できるアプリケーション (MPI等を使う）のみ

- 指定した`walltime`以内にジョブが終わらないと強制終了
- 指定した`walltime`終了時刻ががメンテナンス開始時刻より後だとメンテナンス復帰後にジョブの実行が開始する

CPU点数計算例

`select=1:ncpus=16:mpiprocs=1:ompthreads=16`

- diamond blastx：nr検索、 クエリDNA配列：96,706件、クエリファイルサイズ183MB
- 1ノード、16コア指定、jobtype: **`core`**
- 所要時間: 73:54:05、typeCノードで実行された
- 消費点数： 1点 x 16コア x 73.9時間 = 1,182点

`select=1:ncpus=64:mpiprocs=1:ompthreads=64:jobtype=largemem`

- InterProScan クエリDNA配列：115,079件、クエリファイルサイズ227MB
- 1ノード、64コア指定（1 vnode)、jobtype: **`largemem`**
- 所要時間： 16:30:22、typeFノードで実行された
- 消費点数： 60点 x 1vnode x 16.5時間 = 990点

アレイジョブ

- オプション　**#PBS -J** 開始番号-終了番号
- ジョブファイル中で変数 **${PBS_ARRAY_INDEX}** に番号がセットされ、インクリメントしながら回数分実行される

```
#!/bin/sh
#PBS -J 1-5
#PBS -l select=1:ncpus=4:mpiprocs=1:ompthreads=4
#PBS -l walltime=4:00:00

diamond blastx --threads ${NCPUS} --db nr --outfmt 6 \
 --query ./my.${PBS_ARRAY_INDEX}.fa --out ./out.${PBS_ARRAY_INDEX}.tab
```

- 上記は4コアジョブが5回サブミットされたのと同じ
- 消費点数は5つのジョブそれぞれがかかった時間に依存する
- あまり細かく分割しないこと（大量サブミットしたユーザのジョブは実行が遅れる）

アレイジョブの見え方

```
$ jobinfo
-------------------------------------------------------------------------------
Queue   Job ID     Name            Status  CPUs  User/Grp     Elaps Node/(Reason)
-------------------------------------------------------------------------------
H(c)   6441620[]  test_array.sh  Array     4  ebv/zo0      0:11:39
H(c)   6441620[1] test_array.sh  Run       4  ebv/zo0      0:10:17 ccc035
H(c)   6441620[2] test_array.sh  Run       4  ebv/zo0      0:10:16 ccc035
H(c)   6441620[3] test_array.sh  Run       4  ebv/zo0      0:10:17 ccc035
H(c)   6441620[4] test_array.sh  X         4  ebv/zo0      0:00:04 ccc039
H(c)   6441620[5] test_array.sh  X         4  ebv/zo0      0:00:04 ccc039
-------------------------------------------------------------------------------
```

一時ファイルのためのディレクトリ

- 原則として /tmp, /var/tmp, /dev/shm の使用は禁止
- jsubジョブ実行では環境変数 ${TMPDIR} が 自動で上記以外に設定されている
  - 計算ノード上の /lwork 領域
  - デフォルト書き込み可能容量　：　11.9Gb/コア
  - 不足する場合は、コア数を増やす
- 大容量の一時ファイルを扱いたい場合： /gwork 領域を指定できる
  - 共有ディスク上、利用量上限なし
  - 設定例： `export TMPDIR=/gwork/users/${USER}/${PBS_JOBID}`
  - /lwork より低速

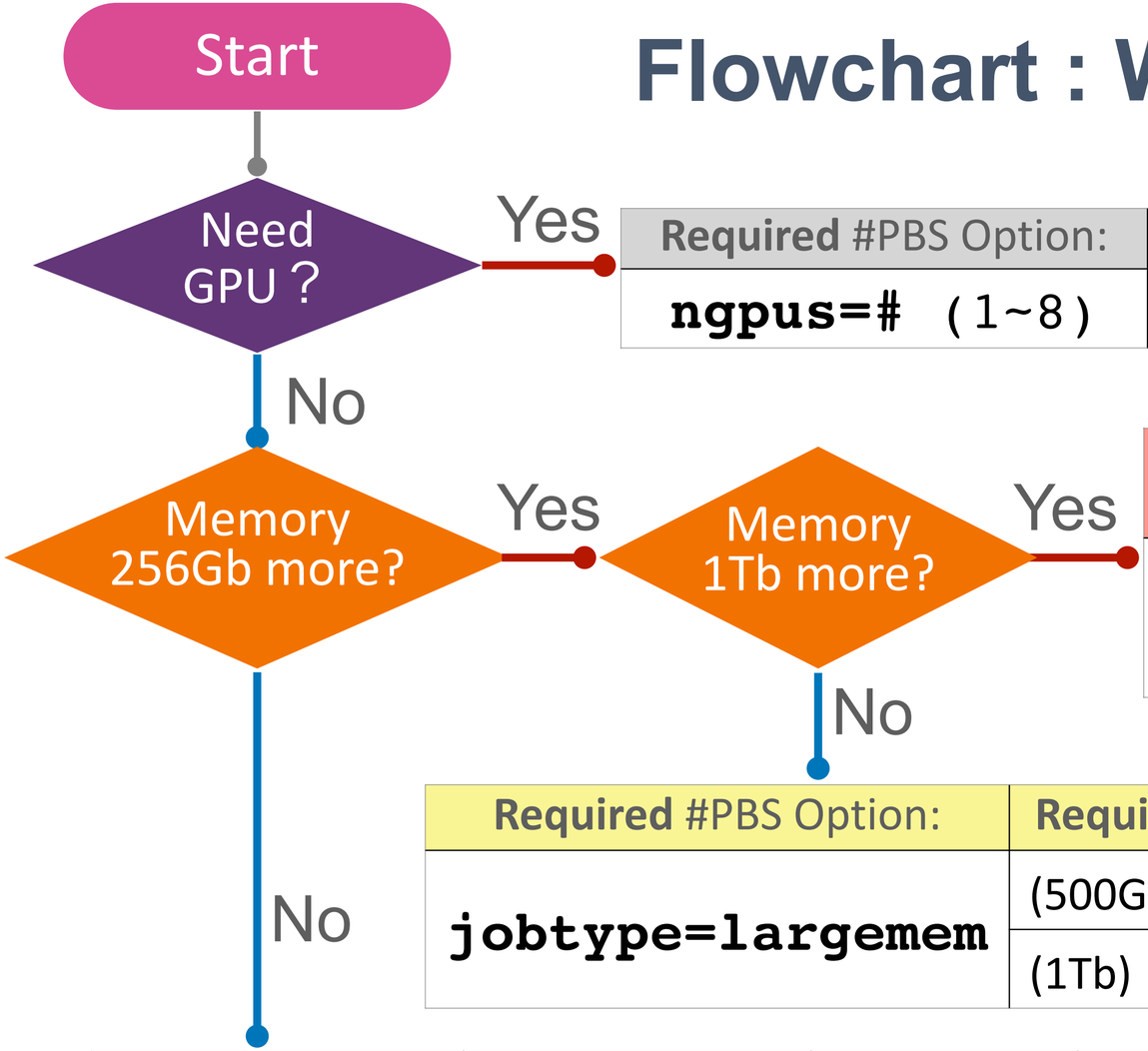
- https://ccportal.ims.ac.jp/manual/storage

全てジョブが終了したら
削除されます

外部からファイルをダウンロードする場合

- jsubジョブ内（計算ノードから）では外部サイトにアクセスできない
- 途中でデータのダウンロードが入るアプリケーションは、先にccfep（ログインノード）上でダウンロードしておくなどの措置が必要
  - BUSCO（FAQに回避方法を記載）, sra-toolkitなど

Flowchart : Which jobtype/queue to use?



Required #PBS Option:	jobtype	CPU score/h	node type
ngpus=# (1~8)	**gpu (g)**	60/1gpu	**G**

Required jsub Option:	**Required** #PBS Option:	jobtype	CPU score/h	node type
jsub -q HB	(1.5Tb) **ncpus=48**	**nibb (b)**	48	**B**
	(3Tb) **ncpus=96**		60	

Required #PBS Option:	**Required** #PBS Option 2:	jobtype	CPU score/h	node type
jobtype=largemem	(500Gb) **ncpus=64**	**largemem (l)**	60/1vnode	**F**
	(1Tb) **ncpus=128**		120/2vnode	

#PBS Option:	Memory	jobtype	CPU score/h	node type
ncpus<64	1.875Gb x ncpus	**core (c)**	1/1core	**C**
ncpus=64		**vnode (v)**	45/1vnode	
ncpus=128			90/2vnode	

情報の在処など

- 計算科学研究センター(RCCS) Webサイト
  https://ccportal.ims.ac.jp/

  - 基礎生物学分野向け情報
    https://ccportal.ims.ac.jp/node/3729

  - 基礎生物学分野向けFAQ
    https://ccportal.ims.ac.jp/node/3776