第1回 PBSゼミ

ゼミ担当者 ： 金美和，松本義秀，中尾昌広，輪湖純也
指導院生 ： 下坂久司
開催日 ： 2002 年 11 月 15 日

ゼミ内容: 近年のコンピュータによる多くの試みは，非常に多くのコンピュータ資源を必要としており，PC クラスタを用いた並列処理などが行われている．PC クラスタにおいて，一部のノードに付加が集中し過ぎていたり，逆にほとんど使用されていないノードが存在したりすれば，効率的にクラスタを利用することができない．そのため，複数のユーザが効率的にクラスタを利用するためには，ジョブの管理を行うためのスケジューラが必要になる．本ゼミでは，スケジューラの１つとして PBS を紹介し，そのインストール方法や基本的な使い方について学ぶ．

1 はじめに

クラスタは通常，複数のユーザが同時に利用する．あるユーザが，他のユーザのジョブを意識せずにジョブを実行してしまうと，Fig. 1 のように，１つのノードで複数のジョブが実行される可能性がある．ジョブの競合が起こったノードでは，プロセッサを効率よく利用できなくなってしまう．特に並列ジョブでは，１つのプロセスの遅れが全プロセスに影響してしまう．そこで，どのノードが空いているかを管理し，ユーザが投入するジョブを適切なノードで実行するために自動的にスケジューリングを行うためのジョブ管理システム (Job Management System：以下 JMS) が必要となる．

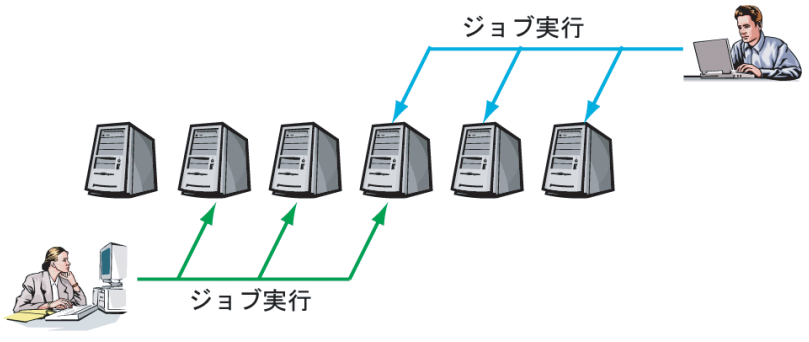

Fig. 1 ジョブの競合

2 様々な JMS

現在 JMS には，市販のもの，フリーのものを併せて 20 種類以上が存在している．ここではその中でも代表的なものをいくつか紹介する．

1. DQS：Distributed Queuing System

   Florida State University によって開発されたフリーの JMS．商用バージョンは Condine と呼ばれ，ドイツの GENIAS GmbH が提供している．ここで紹介するものの中では唯一，Debian 用のパッケージが公式に用意されている．

2. LSF:Load Sharing Facility

   カナダの Platform Computing Corporation によって提供されている商用パッケージであり，20000 以上のライセンスが購入されている．

3. Condor

   University of Wisconsin によって開発されたフリーのパッケージである．信頼性のあるチェックポイントシステムやプロセス移住をサポートした最初のシステムである．リモートからの I/O にも対応している．ただしオーバーヘッドが大きいために，動作は遅い．

4. NQS / Generic NQS / The Connect:Queue

   使い勝手がシンプルである．また NQS は JMS のデファクトスタンダードである．また多くの JMS が NQS との互換性 (NQS コマンドのサポート) を持っている．Sterling Software によって 1980 年代初期には The Connect:Queue と呼ばれる商用パッケージも開発された．これをフリーなパッケージとして実装したのが Generic NQS である．Generic NQS はイギリスの University of Sheffield によって運営されている．

3 PBS とは

PBS(Portable Batch System) とは，Veridian Systems 社によって NASA 向けに開発されたフリーの JMS であり，ネットワークで接続されたスーパーコンピュータや大規模なクラスタを含むマルチプラットフォームな UNIX 環境で動作する．Fig. 2 は PBS のロゴマークである．

Fig. 2　PBS ロゴ

4 PBS の問題点

国内で PC クラスタを提供している HIT 社では，自社の HP で PBS の問題点について言及している．それによると，PBS はポータブルなバッチキューイングシステムを目指して開発されたものであり，複雑な条件下での自動運用までは期待できないとしている．また，その理由として以下のようなことを挙げている．

- 複数の利用者がスクラッチディスクスペース (作業スペース) を取り合ってジョブが落ちる．
- １つのノードにジョブが 3 つも同時に割り当てられてしまうが，他のノードが空いている場合がある．
- 同時に 10 万件程のジョブを投入すると PBS から応答が帰ってこない．

さらに，PBS では困難な課題でも，前述の LSF を導入すると容易に解決できる事も多い．その例として，

- 複数のユーザのジョブが特定の時期に集中する場合のジョブの競合の解消．
- 複数のグループでクラスタを購入した場合のフェアなジョブ管理．
- フリーの JMS の多重度の設定や資源管理機能の制約．
- ヘビーユーザが投入する多量のキューをバイパスする仕組み．
- パラレル計算ノードとシリアル計算ノードの CPU 配分をジョブによって任意に変更．
- 利用者に応じてジョブの優先実行順位を変更．
- クラスタ全体のリソース状態 (CPU，memory，ジョブの実行状況など) を把握．
- クラスタの一部に異常が発生した場合のスムーズな発見．

このように，PBS は様々な問題点も存在している．しかし，HIT 社の HP でも，目安として 8 CPU までの少数ノードで少人数による使用であるなら，フリーの PBS でも十分に実用的ではあると述べており，ベオウルフクラスタにおいては，JMS に商用の LSF を導入するよりもその分のコストでノードを増やすのも悪くはないとしている．

5 PBS の構成

PBS は，4 つの主要なコンポーネントから成る．それらは，コマンド，Job Server，Job Executor，Job Scheduler と呼ばれ，各コンポーネントの主な仕事は，次のようになる．

- コマンド　:

  PBS は，コマンドラインと GUI の両方を提供している．これらは，ジョブの実行，モニタリング，修正，削除などに使われる．コマンドは，さらに次の 3 つに分類することができる．

  1. ユーザ・コマンド : qsub，qstat，qdel，qreturn　等
  2. オペレータ・コマンド : qenable，qdisable，qrun，qstart，qstop，qterm
  3. アドミニストレータ・コマンド : qmgr，pbsnodes

  （注）オペレータ・コマンドとアドミニストレータ・コマンドは，アクセス権限が異なる．

- Job Server　:

  Job Server（以下サーバ）は，PBS において中核となるものである．サーバは，**pbs_server** コマンドにより起動される．すべてのコマンドとその他のデーモンとのコミュニケーションは，IP ネットワークを経由してこのサーバと行われる．

  サーバの主な機能は，バッチジョブの生成・受理，ジョブの修正，システム障害に対するジョブの保護，ジョブの実行（実際には Job Executor にジョブの実行を要求）のような基本的なバッチサービスを提供することである．

  ひとつのサーバが，1 つまたは複数の**queue（以下キュー）**を管理する．キューは，ゼロもしくは複数のバッチジョブを一時的に収容する．キューという名前にもかかわらず，キューの中のジョブは，FIFO（First In First Out）により支配される必要はない．キューへのアクセスは，キューを管理しているサーバに限られているため，すべてのクライアントは，キューやキュー内にあるジョブについての情報をサーバへのバッチ要求を通して得る．キューは，次の 2 つのタイプに分類される．

  1. ルーティング・キュー : ジョブがこのルーティング・キューにある時，このジョブは新しいキューに配置される候補となる．各ルーティング・キューは，ジョブが配置されるべきキューのリストを持っている．この新しいキューは，同じサーバ内の違うキューかまたは違うサーバのキューである．また，サーバはジョブを実行することができるノードを知らなければならない．この情報は，サーバ内の PBS_HOME/server_priv で宣言されている．
  2. エグゼキューション・キュー : ジョブがこのエグゼキューション・キューにある時，このジョブは実行の候補となる．

- Job Executor　:

  Job Executor は，実際にジョブを実行するためのデーモンである．このデーモン (**pbs_mom**) は，すべての実行中のジョブの生みの親となるため**Mom** と呼ばれる．Mom は，サーバからジョブのコピーを受け取った時，そのジョブを実行する．また，Mom は実行ジョブの出力についても責任を負っている．Mom は，ジョブを実行できるノード毎に動いている必要がある．

- Job Scheduler　:

  Job Scheduler（以下スケジューラ）は，どのジョブをどこで，またいつ実行させるかといったポリシーの管理を含むデーモンである．なぜならば各サイトは，何が良くて，効率的であるかという独自の考え方を持っている．そのため PBS は，各サイトが独自のスケジューラを生成することを許している．デフォルトは，FIFO スケジューラとなっている．FIFO スケジューラは，fifo オーダに加えて，いくつかの異なるジョブのソート機能を持っており，PBS_HOME/sched_priv/sched_config ファイルを編集することによって，どのジョブをいつ実行するかといったスケジューリングポリシーを変更できる．

  スケジューラは，**pbs_sched** コマンドにより起動される．実行時には，スケジューラは Mom からシステムのリソースに関する状態を獲得し，サーバからジョブのアベイラビリティについて情報を獲得する．サーバとのインタフェースはコマンドと同じ API である．実際，スケジューラはサーバのバッチ・マネージャとして現れる．

以下の Fig. 3 は，マルチホスト環境における PBS の各コンポーネントの連携図である．

1. クライアントはイベントをサーバに与え，スケジューリングの開始を要求する
2. サーバはスケジューリング・コマンドをスケジューラへ送る
3. スケジューラはリソース情報を Mom に要求する
4. Mom は，要求された情報をスケジューラに返す
5. スケジューラは，ジョブに関する情報をサーバに要求する
6. サーバはジョブの状態をスケジューラに送り，ポリシーに従ってジョブを実行するための決定を行う
7. スケジューラは，サーバに実行要求を行う
8. サーバはジョブを Mom に送り，Mom はジョブを実行する

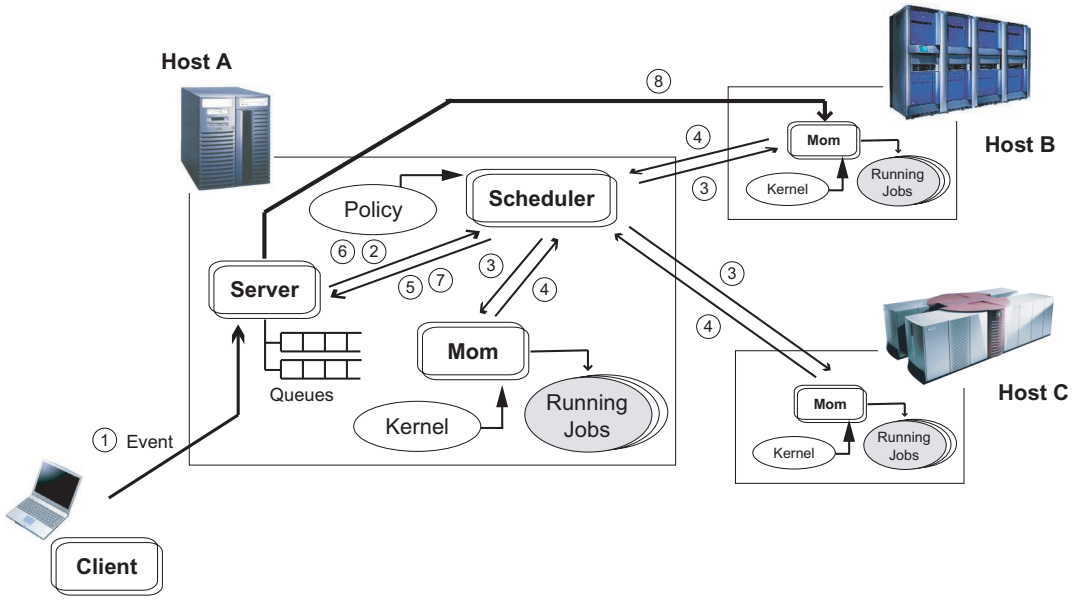

Fig. 3　PBS の構成

6　インストール方法

6.1　PBS の入手方法

`http://pbs.mrj.com/download.html` にてダウンロードを行う．その際にユーザ登録をする必要がある．ユーザ登録終了後，2 日以内に確認メールが送られ PBS のダウンロードが可能になる．

6.2　インストール

まずダウンロードした PBS のアーカイブを適当なディレクトリで展開する．

```
tar xvfz OpenPBS_X_X_XX.tar.gz
```

同じディレクトリ内に解凍された PBS のディレクトリに入り，コンパイルとインストールを行う．

```
cd OpenPBS_X_X_XX
./configre [option]
make
make install
```

なお./configre にオプションをつけることによって，PBS のインストール場所や機能を追加・削除などを行うことが

できる．以下に主要なオプションを示す．

- `--prefix=`場所 PBS をインストールするディレクトリを指定 (Default:/usr/local)
- `--enable-docs` PBS のドキュメントのインストール (Default:disable)
- `--enable-server` PBS サーバのインストール (Default:enabled) [1]

6.3 ノードの設定

6.3.1 サーバの設定

PBS サーバとなるマシン上で，`{PREFIX}/server_priv/nodes` というファイルを作成し，ジョブを実行させたいホスト名を記述する．以下に例を示す．

```
Queen
Slave
```

もし，ノードがタイムシェアリングノードであれば，行末に:ts を付加する．

```
Queen:ts
Slave:ts
```

次に pbs_server，pbs_sched，pbs_mom のデーモンを実行する．なお，pbs_server を最初に実行させるときは”-t”オプションをつけて実行させる[2]．2 回目以降は必要ない．

```
# pbs_server -t create
# pbs_sched
# pbs_mom
```

次に`{PREFIX}`/mom_priv/config というファイルを作成し，ジョブを実行させたいホスト名を以下のように記述する．

```
$clienthost Queen
$clienthost Slave
```

6.3.2 スレーブの設定

スレーブとなるマシン上で pbs_mom のデーモンを実行する．

```
# pbs_mom
```

次に`{PREFIX}`/mom_priv/config というファイルを作成し，ジョブを実行させたいホスト名を記述する[3]．

```
$clienthost Queen
$clienthost Slave
```

6.4 キューの設定

キューとサーバの設定は qmgr というコマンドで行う．またこの設定はサーバだけで行う．qmgr のコマンドの書式は以下の通りである．

```
command server [names] [attr OP value[,attr OP value,...]]
command queue [names] [attr OP value[,attr OP value,...]]
command node [names] [attr OP value[,attr OP value,...]]
```

command には以下の変数が入る．

- active　動作させたいオブジェクトをセットする．

[1] スレーブには server が必要ないので disable にしておいた方がよい
[2] インストールする場所を変えた場合，bin などに PATH を通したものとしてこれ以降は説明を行う．
[3] サーバの設定と同じ

- create　新しいオブジェクト（キュー，ノード）を作成する．
- delete　既存のオブジェクト（キュー，ノード）を消す．
- set　オブジェクトで定義されている属性を変更する．
- unset　定義されているオブジェクトの属性を外す．
- list　現在の属性と関連付けられたもの一覧表を作る．
- print　すべてのキューとサーバの属性をプリントする．

OP には以下の変数が入る．command には以下の変数が入る．

- ＝　右辺の属性をセットする．
- ＋＝　右辺の属性を追加する．
- －＝　右辺の属性を削除する．

次にその他の設定を示す．

- queue_type = Execution か Routing を選択する
- enabled = True としたならば，キューにジョブが入る．False としたならばジョブを受け付けない
- started = True としたならばジョブが実行されるかサーバによってジョブが配分される
- route_destination(Routing キューのみ)　他のサーバのキューにジョブを送ることができる．(例：Qmgr:set queue route_destinations=para,overthere@anoter.com)
- resources_max　計算資源に制限を加えることができる (例：Qmgr:set queue resource_max.cput=2:00:00 で 2 時間を越える CPU 時間がキューの中で与えられないようにできる)．以下に主要なオプションを示す．
  - cput: ジョブの中のすべてのプロセスに使われる CPU 時間
  - pcput: ジョブの中の一つのプロセスに使われる CPU 時間
  - mem: ジョブに使われる最大メモリ総量
  - pmem: ジョブの中のすべてのプロセスに使われる最大メモリ総量
  - vmem: ジョブに使われる最大仮想メモリ総量
  - pvmem: ジョブの中のすべてのプロセスに使われる最大仮想メモリ総量
  - nodect: ユーザが使うことのできるノード数
  - walltime: 待ち時間
- resources_min　資源限界の最小値を定義する
- resources_default　資源範囲を指定しなかったキューに入力する仕事に対するデフォルトの値を定義する

設定例を以下に示す[4]．

```
Queen:/root# qmgr
#キューの作製
Qmgr:create queue para
Qmgr:set queue para queue_type = Execution
Qmgr:set queue para enabled = True
Qmgr:set queue started = True
Qmgr:set server scheduling = True
Qmgr:set server default_queue = para
#サーバへのアクセス制限
Qmgr:set server acl_hosts = *.isl.doshisha.ac.jp
Qmgr:set server acl_host_enable = True
#キューに関するリソースの設定
Qmgr:set queue para resourves_min.nodect=1
Qmgr:set queue para resourves_max.nodect=2
#終了
Qmgr:quit
```

[4]Qmgr:はプロンプトとして現れる

7 PBS の実行方法

PBS を利用する主な内容は以下の三つに分けられる.

- qsub : ジョブを投入する.
- qstat : キュー，ジョブの状態を表示する.
- qdel : ジョブを削除する.

それぞれの実行方法について説明していく.

7.1 ジョブの投入

ジョブを投入するには qsub コマンドを用いる．実行方法には直接コマンドを入力する標準入力方法と，スクリプトを用いる方法がある.

7.1.1 標準入力

qsub コマンドの ” -l ”オプションで使用するリソースをする．Fig. 4 にその例を示す.

```
kim@Queen:~/pbs$ qsub -l ncpus=1,nodes=1 -q para -N test[Enter]  …1
cd $PBS\_O\_WORKDIR [Enter]              …2
./a.out [Enter]                        …3
[Ctrl]+[d]                             …4
3.Queen.work.isl.doshisha.ac.jp        …5
kim@Queen:~/pbs$
```

Fig. 4　qsub コマンドの直接入力

1. CPU，ノード数，キュー, ジョブ名を指定する.
2. ジョブはホームディレクトリで実行を開始するため，実行ファイルのあるディレクトリに cd で移動する.
3. 実行ファイルを指定する.
4. 入力終了は [Ctrl]+[d].
5. JobID が割り振られる.

7.1.2 シェルスクリプトによる入力

上記と同様の内容をシェルスクリプトを用いて記述した例を Fig. 6 に示す.

```
kim@Queen:~$ less pai.sh
#!/bin/sh
#PBS -l ncpus=1
#PBS -l nodes=1
#PBS -q para
#PBS -N test
cd $PBS_O_WORKDIR
./a.out
kim@Queen:~$ qsub pai.sh
3.Queen.work.isl.doshisha.ac.jp
kim@Queen:~$
```

Fig. 5　シェルスクリプトの記述

7.1.3 qsub コマンド

qsub の主なオプションコマンドは以下のとおりである.

＜ qsub のオプション＞

オプション	指定名	説明
-N	jobName	ジョブの名前
-q	queueName	キューの指定
-o	outFile	標準出力をファイルに保存
-e	errorFile	エラーの出力をファイルに保存
-j	oe	エラーを標準出力にまとめて保存する
-l	ncpus	1 つのジョブで必要とする CPU 数
	mem	1 つのジョブで必要とする総メモリ量
	walltime	1 つのジョブで必要となる計算時間
	nodes	1 つのジョブで必要とする node 数
-m	a	ジョブがエラー終了した場合にメールを送る
	b	ジョブのスタート時点でメールを送る
	e	ジョブの終了時点でメールを送る
	abe	3 つを指定する場合は続けて書く

＜実行例＞

```
kim@Queen:~$ ls
OpenPBS_2_3_16.tar.gz  a.out   ccc.sh.e5  cpi.c  pai.sh
a                      ccc.sh  ccc.sh.o5  cpi.o  upt.sh
kim@Queen:~$ less pai.sh
#!/bin/sh
#PBS -A nakao
#PBS -N test
#PBS -j oe
#PBS -l ncpus=2
mpirun -np 2 a.out
exit 0

kim@Queen:~$ qsub pai.sh
17.Queen.work.isl.doshisha.ac.jp
kim@Queen:~$ ls
OpenPBS_2_3_16.tar.gz  a.out   ccc.sh.e5  cpi.c  pai.sh    upt.sh
a                      ccc.sh  ccc.sh.o5  cpi.o  test.o17
kim@Queen:~$ less test.o17
Process 0 on Queen.work.isl.doshisha.ac.jp
Process 1 on Slave.work.isl.doshisha.ac.jp
pi is approximately 3.1416009869231241, Error is 0.0000083333333309
wall clock time = 0.000915
```

Fig. 6　MPI を利用した pbs の実行例

7.2 ジョブ実行状況

ジョブの実行状況を確認するためには qstat コマンドを用いる．主な qstat コマンドには以下のようなものがある．

＜ qstat のオプション＞

オプション	説明
qstat -Q	全てのキューのリミット値を表示
qstat -q	システムの全てのキューを表示
qstat -a	システムの全てのジョブを表示
qstat -s	全てのジョブをステータスコメント付きで表示
qstat -r	実行中の全てのジョブを表示

qstat の各オプションについて説明する．

7.2.1 全てのキューのリミット値を表示 [qstat -Q]

```
kim@Queen:~$ qstat -Q
Queue            Max Tot Ena Str Que Run Hld Wat Trn Ext Type
---------------- --- --- --- --- --- --- --- --- --- --- ----------
para               0   0 yes yes   0   0   0   0   0   0 Execution
```

Fig. 7　qstat -Q の実行結果

項目	説明	項目	説明
Max	キューで同時に実行できる最大のジョブ数	Hld	ホールドされているジョブ数
Tot	キュー待ちのジョブ数	Wat	待ち状態のジョブ数
Ena	キューのステータス (Enable yes or no)	Trn	別のキューまたはサーバーに転送されたジョブ数
Str	キューのステータス (Started yes or no)	Ext	終了状態のジョブ数
Que	キューイングされているジョブ数	Type	キュータイプ (Execution or Route)
Run	実行中のジョブ数		

7.2.2 システムの全てのキューの表示 [qstat -q]

```
kim@Queen:~$ qstat -q

server: Queen

Queue            Memory CPU Time Walltime Node Run Que Lm  State
---------------- ------ -------- -------- ---- --- --- --  -----
para               --      --       --     --    0   0 --   E R
```

Fig. 8　qstat -q の実行結果

項目	説明
Memory	キューで利用できる最大メモリ容量
CPU Time	1 つのジョブで利用可能な最大時間
Walltime	1 つのジョブで利用可能な最大経過時間
Node	1 つのジョブで利用可能な最大 node 数
Run	実行中のジョブ数
Que	実行待ちのジョブ数
Lm	同時に実行可能なジョブ数
Srate E	キューのステータス (Enable or Disable)
Srate R	ジョブのステータス (Running or Stopped(投入不可))

7.2.3 システムの全てのジョブを表示 [qstat -a]

```
[sgiadm@bshead PBS]$ qstat -a
work.isl.doshisha.ac.jp:
                                                            Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID NDS TSK Memory Time  S Time
--------------- -------- -------- ---------- ------ --- --- ------ ----- - -----
3.Queen.work.is kim   para     pbstest005   9917   5   5   20mb 744:0 R   --
4.Queen.work.is kim   para     pbstest005   9832   5   5   20mb 744:0 R   --
```

Fig. 9　qstat -a の実行結果

7.3 ジョブの削除

ジョブの削除には Job ID を指定します.

1. qstat でキューに入っているジョブの"Job ID"を確認する.
2. キャンセルしたいジョブの"Job ID"を qdel を使ってキャンセルする.
3. qstat でキャンセルされているかを確認する.

項目	説明
Job ID	ジョブ ID
Username	ジョブの投入を行った user アカウント名
Queue	ジョブが投入されたキュー名
Jobname	ジョブの投入時に User が指定したジョブ名
SessID	セッション ID
NDS	ジョブでのクエスト node 数
TSK	ジョブでのリクエスト CPU 数 or タスク数
Req'd Memory	ジョブによってリクエストされる memory 数
Req'd Time	ジョブによってリクエストされる時間
S	ジョブのステータス (Running or Queueing)
Elap Time	ジョブが使用した CPU 時間 or 経過時間

```
kim@Queen:~/pbs$
kim@Queen:~/pbs$ qsub test.sh     …1
3.Queen.work.isl.doshisha.ac.jp
kim@Queen:~/pbs$ qstat -a
work.isl.doshisha.ac.jp:
                                                                  Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID NDS TSK Memory Time  S Time
--------------- -------- -------- ---------- ------ --- --- ------ ----- - -----
3.Queen.work.is kim   para     pbstest005   9917   5   5   20mb 744:0 R   --
kim@Queen:~/pbs$ qdel 3          …2
kim@Queen:~/pbs$ qstat -a              …3
kim@Queen:~/pbs$
```

Fig. 10　ジョブの削除

参考文献

1) PC クラスタにおけるジョブ管理

2) クラスタシステムにおけるジョブスケジューラ Condor

3) LSF+SCC

4) 日本 SGI　PBS とその利用法