OPENLAVA 使用介绍

李永义/王均栋
young@caryon.net
leowang@caryon.net
Feb.2017

上海凯远网络科技有限公司

Shanghai Caryon Network Technology Co., Ltd.

- 环境设置及进程的启动
- 提交作业
- 查看作业
- 其他常用命令
- 相关配置文件

环境设置及进程的启动

openlava安装目录：/share/apps/openlava

设置环境变量：

```
source /etc/profile.d/openlava.sh
```

设置开机自启动：

```
chkconfig openlava on

chkconfig --list openlava
openlava        0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

启动、查看、停止openlava进程：

```
service openlava start|status|stop
```

```
$ service openlava status
lim pid: <19766>
res pid: <19768>
sbatchd pid: <19771>
lim mbatchd: <19776>
```

管理节点会显示上述四个进程，计算节点只有三个，没有mbatchd进程

提交作业

方法一：编写lsf提交作业的脚本

提交作业脚本： test.lsf
用户可将此脚本拷贝到自己的工作目录，做简单修改后使用。

提交作业方法：**bsub < test.lsf**
注意：这里有个<符号

脚本内容见下页：

```
#!/bin/bash

#BSUB -J caryontest
#BSUB -q normal
#BSUB -n 32
#BSUB -R span[ptile=16]
#BSUB -o out.%J.txt
#BSUB –m pool1


mpirun -bootstrap lsf /share/apps/platform/helloworld
```

作业名称

队列名称,必须指定

作业使用的**CPU**核数，建议为每节点核数的倍数

每节点分配**20**核，如不写此项，**lsf**将自动分配

输出文件，**%J**代表作业号

指定节点或节点组提交作业，若无必要不建议指定节点，若要使用此项，请将前面的#只保留一个。若需指定多个节点：**#BSUB –m “c0301 c0302”**， 此处**pool1**代表节点组

mpirun的参数

执行程序，若用户有自己编译的程序，请修改为自己的程序路径名称

```
#BSUB -J xhpltest
#BSUB -q normal
#BSUB -n 256
#BSUB -R span[ptile=8]
#BSUB -o out.%J
#BSUB -e error.%J
#BSUB -x

source  /share/apps/intel/ipsxe2015u5/parallel_studio_xe_2015/psxevars.sh  >/dev/null 2>&1
export I_MPI_DAPL_DIRECT_COPY_THRESHOLD=655360
export I_MPI_EAGER_THRESHOLD=128000

##cp HPL_offload.dat  HPL.dat
##mpirun -bootstrap lsf ~/intel64/xhpl_intel64

##env |grep LSB > debug.txt
echo $LSB_HOSTS|sed 's/ /\n/g'|uniq -c|awk '{print $2}' > ./hostlist.$LSB_JOBID
mpirun  -machinefile ./hostlist.$LSB_JOBID  -np $LSB_DJOB_NUMPROC  ~/intel64/xhpl_intel64

##rm -rf ./hostlist.$LSB_JOBID
echo "Tested on host: `hostname`"
```

更“传统”的MPI提交参数

mpirun的参数-machinefile 和 -np

run_xhpl.traditional.lsf

```
#/bin/bash

#BSUB -J fluent1
#BSUB -q normal
#BSUB -n 560
#BSUB -R span[ptile=28]
#BSUB -o out.%J
#BSUB -e error.%J
##BSUB -x

/share/apps/ansys_inc/v172/fluent/bin/fluent -pib -g 3d -t $LSB_DJOB_NUMPROC \
-cnf=$LSB_DJOB_HOSTFILE -i 250.jou 1 > 250.out
```

Fluent提交作业的脚本

```
#!/bin/bash
#BSUB -q normal
#BSUB -o out%J.txt
#BSUB -e error%J.txt
#BSUB -J startest
#BSUB -n 56
#BSUB -R "span[ptile=28]"
##BSUB -R "span[hosts=1]"

/share/apps/CD-adapco/11.06.010-R8/STAR-CCM+11.06.010R8/star/bin/starccm+ \
-rsh ssh -batchsystem lsf -batch iteration.java suo1-8-steady-65.sim
```

StarCCM提交作业的脚本

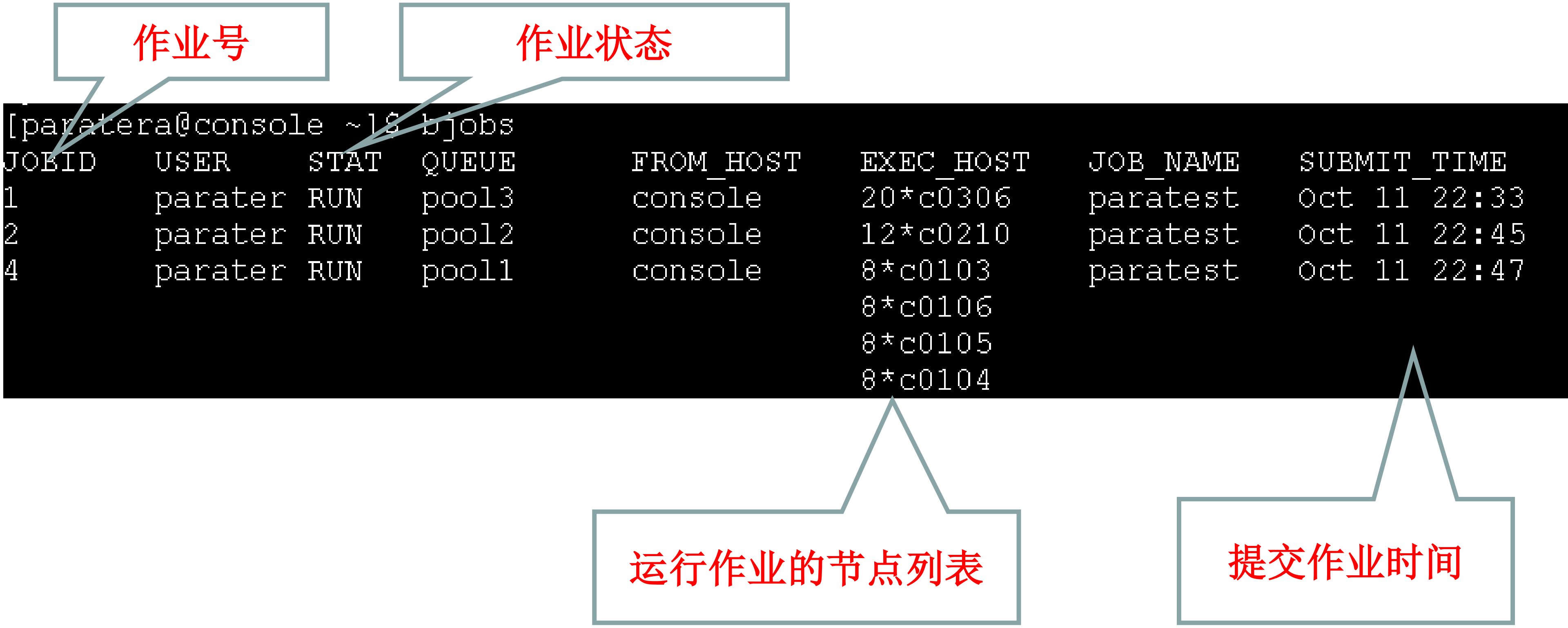
方法二：直接命令行提交

```
#bsub -J paratera -q normal -n 32 -R "span[ptile=16]"
-o out.%J.txt -m pool1 mpirun --bootstrap  lsf
/share/apps/platform/helloworld
```

与上述脚本方式提交效果相同

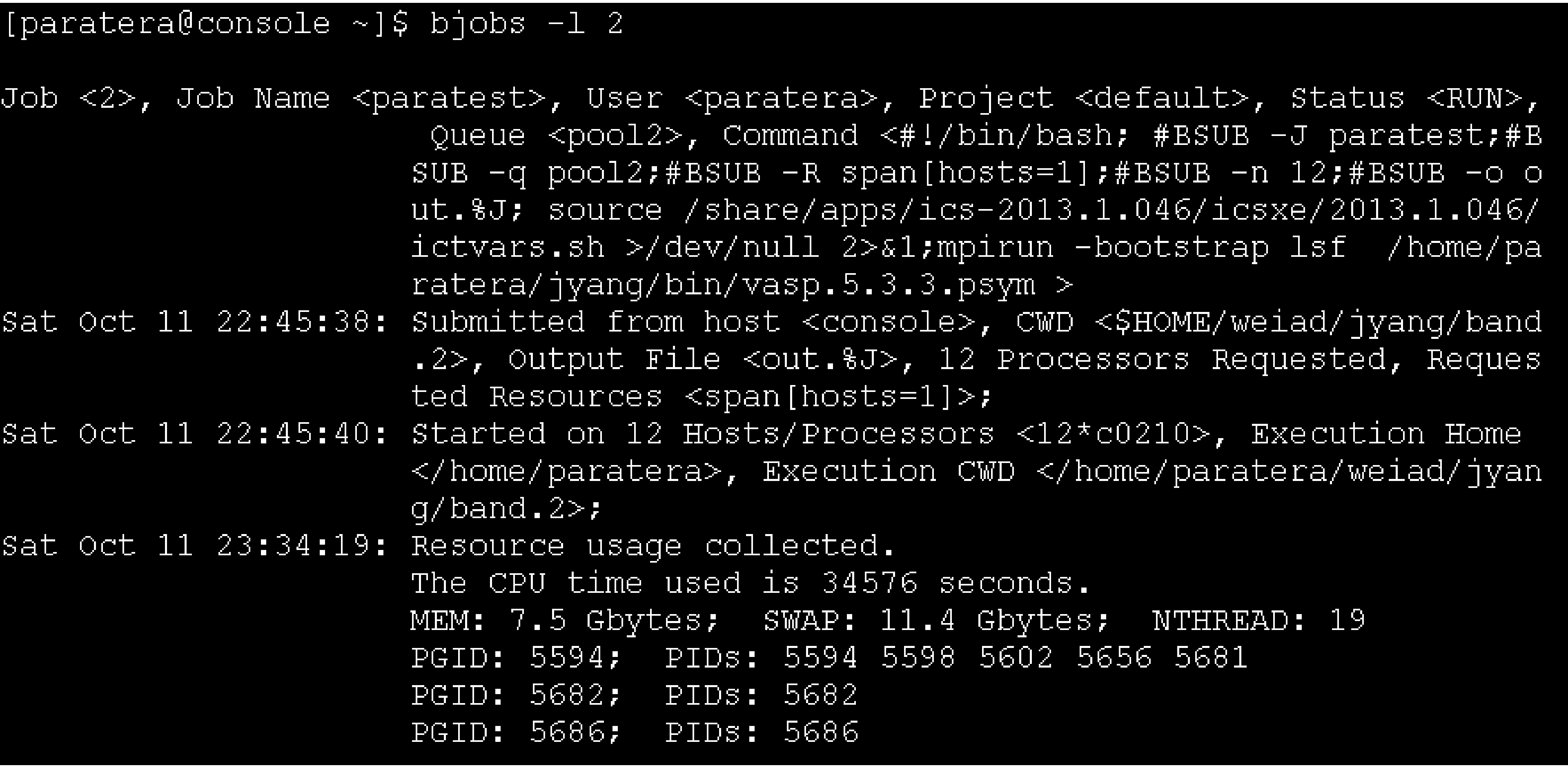
查看作业

查看作业命令：bjobs



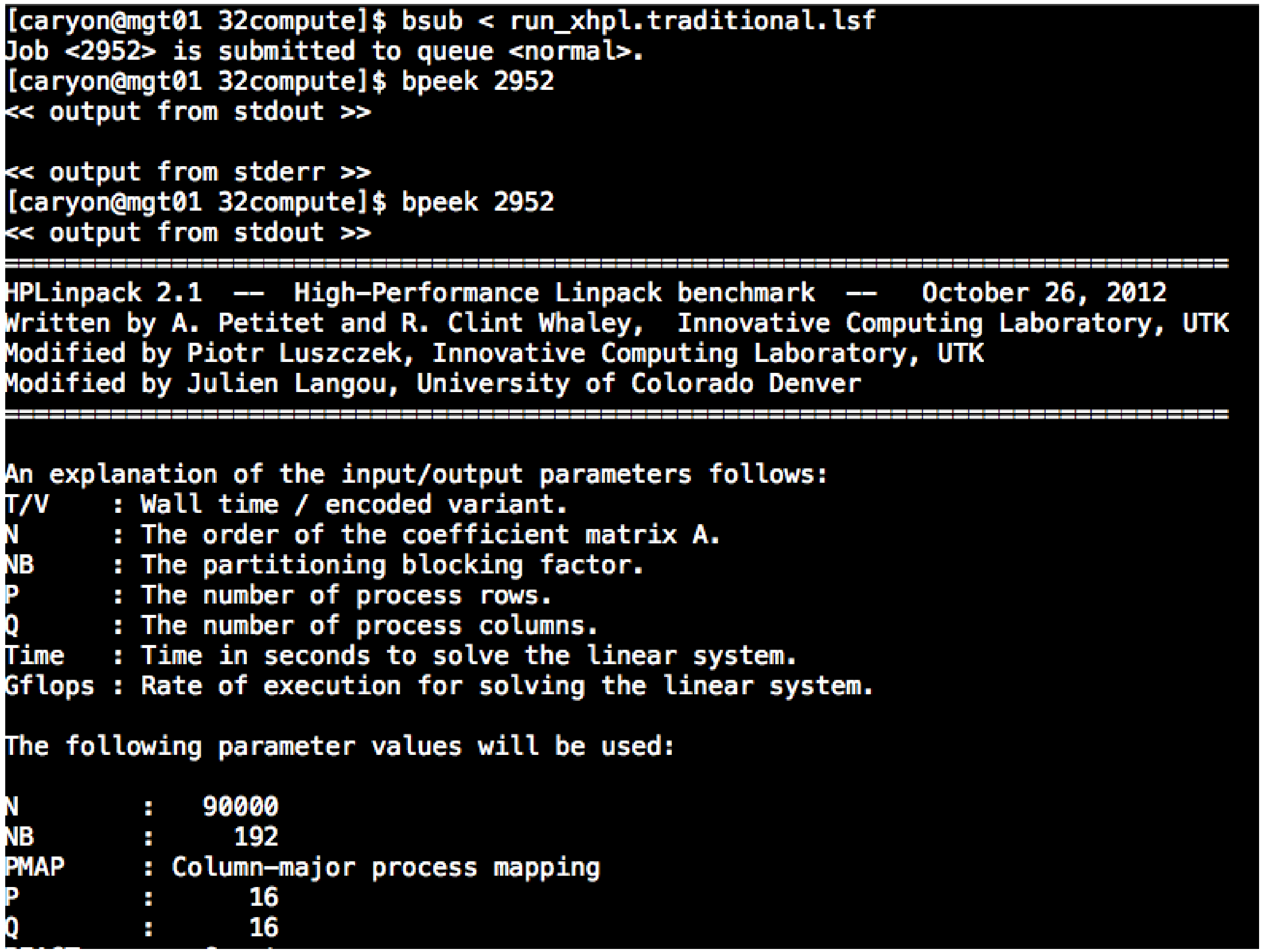
```
[paratera@console ~]$ bjobs
JOBID   USER    STAT  QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME   SUBMIT_TIME
1       parater RUN   pool3      console     20*c0306    paratest   Oct 11 22:33
2       parater RUN   pool2      console     12*c0210    paratest   Oct 11 22:45
4       parater RUN   pool1      console     8*c0103     paratest   Oct 11 22:47
                                             8*c0106
                                             8*c0105
                                             8*c0104
```

查看某个作业的详细信息：bjobs –l jobid



```
[paratera@console ~]$ bjobs -l 2

Job <2>, Job Name <paratest>, User <paratera>, Project <default>, Status <RUN>,
                      Queue <pool2>, Command <#!/bin/bash; #BSUB -J paratest;#B
                     SUB -q pool2;#BSUB -R span[hosts=1];#BSUB -n 12;#BSUB -o o
                     ut.%J; source /share/apps/ics-2013.1.046/icsxe/2013.1.046/
                     ictvars.sh >/dev/null 2>&1;mpirun -bootstrap lsf  /home/pa
                     ratera/jyang/bin/vasp.5.3.3.psym >
Sat Oct 11 22:45:38: Submitted from host <console>, CWD <$HOME/weiad/jyang/band
                     .2>, Output File <out.%J>, 12 Processors Requested, Reques
                     ted Resources <span[hosts=1]>;
Sat Oct 11 22:45:40: Started on 12 Hosts/Processors <12*c0210>, Execution Home
                     </home/paratera>, Execution CWD </home/paratera/weiad/jyan
                     g/band.2>;
Sat Oct 11 23:34:19: Resource usage collected.
                     The CPU time used is 34576 seconds.
                     MEM: 7.5 Gbytes;  SWAP: 11.4 Gbytes;  NTHREAD: 19
                     PGID: 5594;  PIDs: 5594 5598 5602 5656 5681
                     PGID: 5682;  PIDs: 5682
                     PGID: 5686;  PIDs: 5686
```

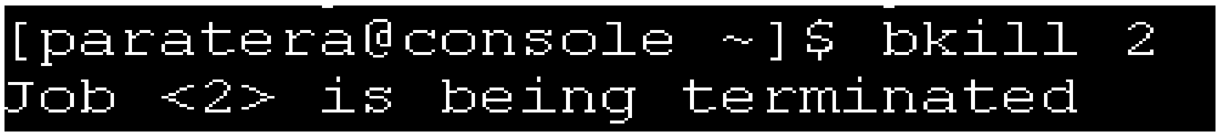
查看某个作业运行过程的屏幕输出： bpeek jobid



```
[caryon@mgt01 32compute]$ bsub < run_xhpl.traditional.lsf
Job <2952> is submitted to queue <normal>.
[caryon@mgt01 32compute]$ bpeek 2952
<< output from stdout >>

<< output from stderr >>
[caryon@mgt01 32compute]$ bpeek 2952
<< output from stdout >>
================================================================================
HPLinpack 2.1  --  High-Performance Linpack benchmark  --   October 26, 2012
Written by A. Petitet and R. Clint Whaley,  Innovative Computing Laboratory, UTK
Modified by Piotr Luszczek, Innovative Computing Laboratory, UTK
Modified by Julien Langou, University of Colorado Denver
================================================================================

An explanation of the input/output parameters follows:
T/V    : Wall time / encoded variant.
N      : The order of the coefficient matrix A.
NB     : The partitioning blocking factor.
P      : The number of process rows.
Q      : The number of process columns.
Time   : Time in seconds to solve the linear system.
Gflops : Rate of execution for solving the linear system.

The following parameter values will be used:

N        :   90000
NB       :     192
PMAP     : Column-major process mapping
P        :      16
Q        :      16
```

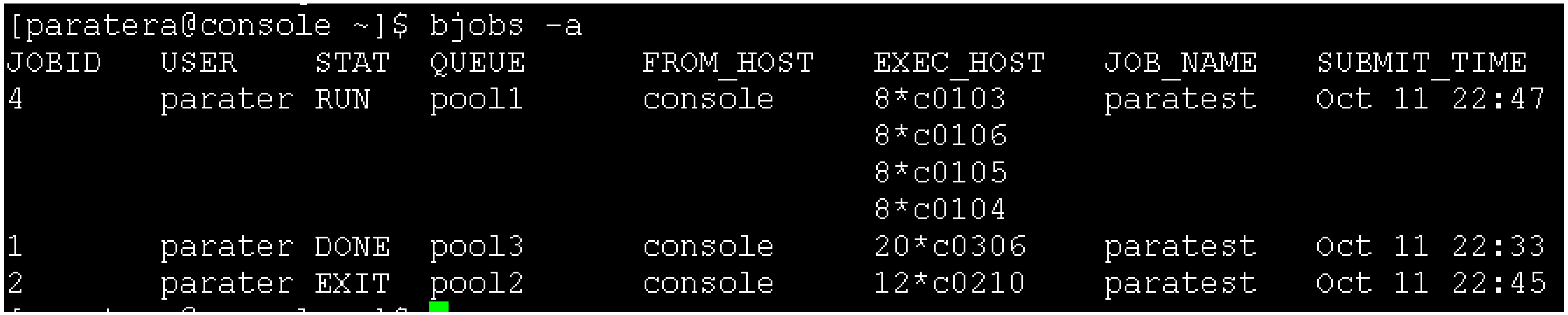
杀掉作业命令：bkill jobid

普通用户只可以杀掉自己作业，管理员可以杀掉任意用户作业



```
[paratera@console ~]$ bkill 2
Job <2> is being terminated
```

再次查看作业状态：bjobs –a
-a查看所有作业，包括近期（一小时内）结束的作业



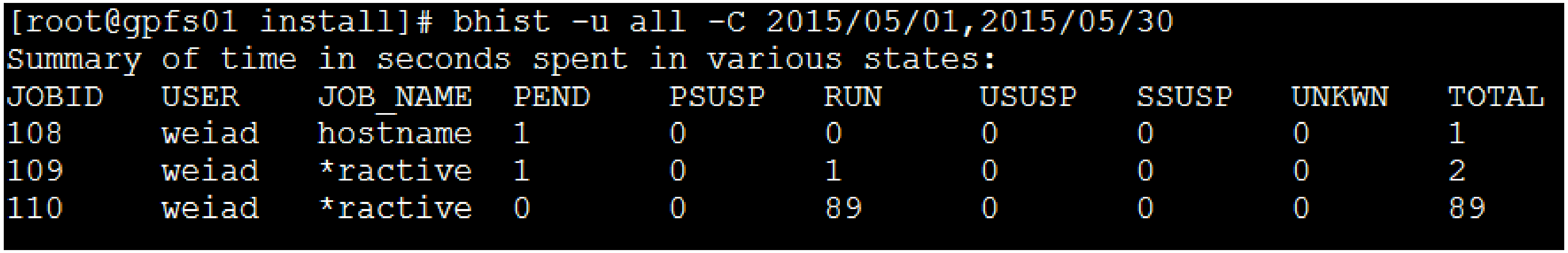
```
[paratera@console ~]$ bjobs -a
JOBID   USER    STAT  QUEUE      FROM_HOST    EXEC_HOST    JOB_NAME   SUBMIT_TIME
4       parater RUN   pool1      console      8*c0103      paratest   Oct 11 22:47
                                              8*c0106
                                              8*c0105
                                              8*c0104
1       parater DONE  pool3      console      20*c0306     paratest   Oct 11 22:33
2       parater EXIT  pool2      console      12*c0210     paratest   Oct 11 22:45
```

作业状态：RUN表示作业正在运行
DONE表示作业运行正常结束
EXIT表示作业退出，出错退出或被用户杀掉

查看历史作业信息：bhist

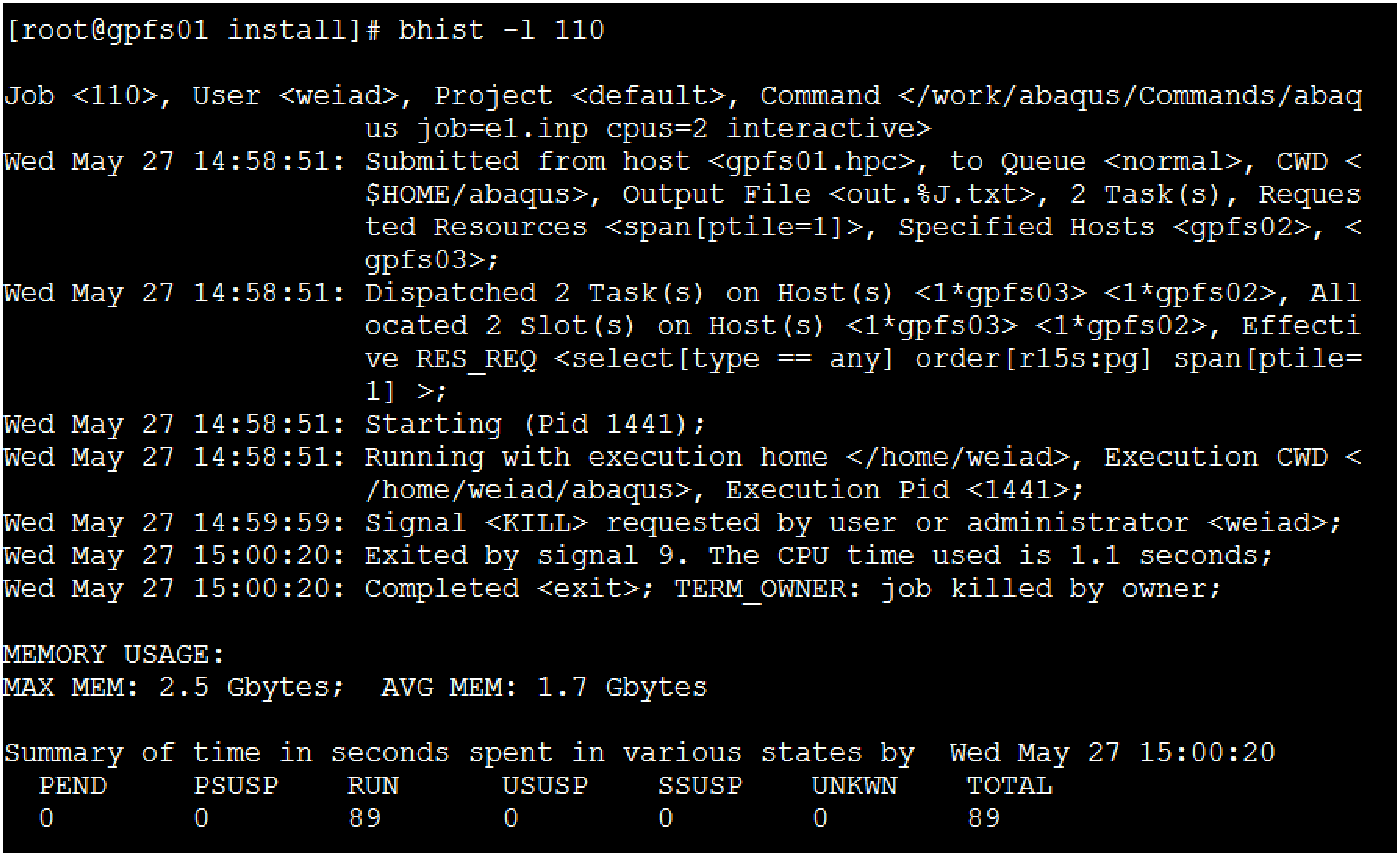
-u all 指定查询用户，all代表所有用户

-C 指定查询时间段



```
[root@gpfs01 install]# bhist -u all -C 2015/05/01,2015/05/30
Summary of time in seconds spent in various states:
JOBID   USER    JOB_NAME  PEND    PSUSP   RUN     USUSP   SSUSP   UNKWN   TOTAL
108     weiad   hostname  1       0       0       0       0       0       1
109     weiad   *ractive  1       0       1       0       0       0       2
110     weiad   *ractive  0       0       89      0       0       0       89
```

-l 表示显示详细信息，110表示作业号

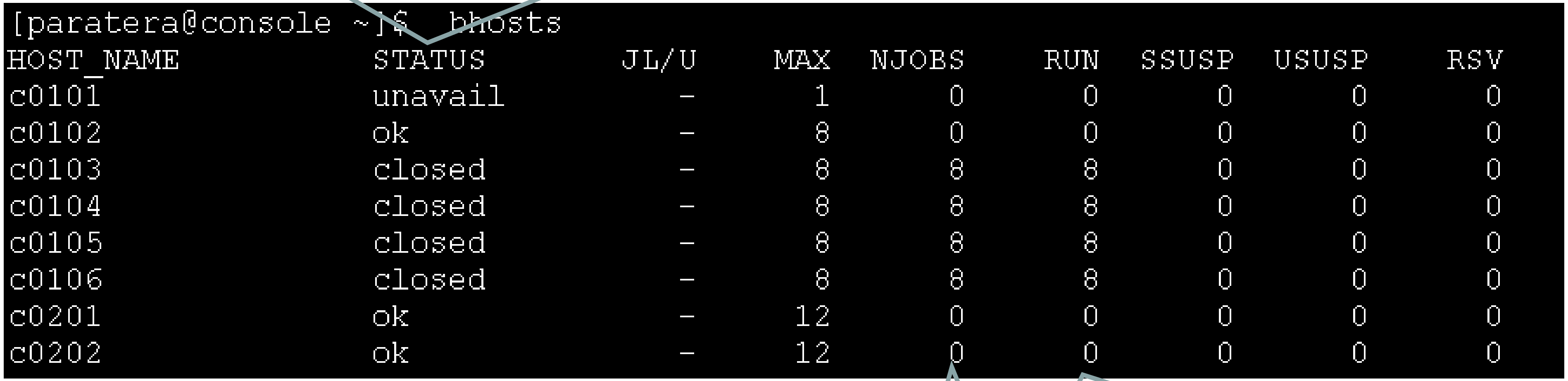


```
[root@gpfs01 install]# bhist -l 110

Job <110>, User <weiad>, Project <default>, Command </work/abaqus/Commands/abaq
                     us job=e1.inp cpus=2 interactive>
Wed May 27 14:58:51: Submitted from host <gpfs01.hpc>, to Queue <normal>, CWD <
                     $HOME/abaqus>, Output File <out.%J.txt>, 2 Task(s), Reques
                     ted Resources <span[ptile=1]>, Specified Hosts <gpfs02>, <
                     gpfs03>;
Wed May 27 14:58:51: Dispatched 2 Task(s) on Host(s) <1*gpfs03> <1*gpfs02>, All
                     ocated 2 Slot(s) on Host(s) <1*gpfs03> <1*gpfs02>, Effecti
                     ve RES_REQ <select[type == any] order[r15s:pg] span[ptile=
                     1] >;
Wed May 27 14:58:51: Starting (Pid 1441);
Wed May 27 14:58:51: Running with execution home </home/weiad>, Execution CWD <
                     /home/weiad/abaqus>, Execution Pid <1441>;
Wed May 27 14:59:59: Signal <KILL> requested by user or administrator <weiad>;
Wed May 27 15:00:20: Exited by signal 9. The CPU time used is 1.1 seconds;
Wed May 27 15:00:20: Completed <exit>; TERM_OWNER: job killed by owner;

MEMORY USAGE:
MAX MEM: 2.5 Gbytes;  AVG MEM: 1.7 Gbytes

Summary of time in seconds spent in various states by  Wed May 27 15:00:20
  PEND     PSUSP    RUN      USUSP    SSUSP    UNKWN    TOTAL
  0        0        89       0        0        0        89
```

其他常用命令

查看节点状态：bhosts

节点状态：**ok**节点正常可用，**closed**节点核数已满，**unavail**节点不可达，一般是因为节点未开机或**lsf**进程没有正常启动



```
[paratera@console ~]$  bhosts
HOST_NAME               STATUS          JL/U    MAX  NJOBS    RUN  SSUSP  USUSP    RSV
c0101                   unavail            -      1      0      0      0      0      0
c0102                   ok                 -      8      0      0      0      0      0
c0103                   closed             -      8      8      8      0      0      0
c0104                   closed             -      8      8      8      0      0      0
c0105                   closed             -      8      8      8      0      0      0
c0106                   closed             -      8      8      8      0      0      0
c0201                   ok                 -     12      0      0      0      0      0
c0202                   ok                 -     12      0      0      0      0      0
```

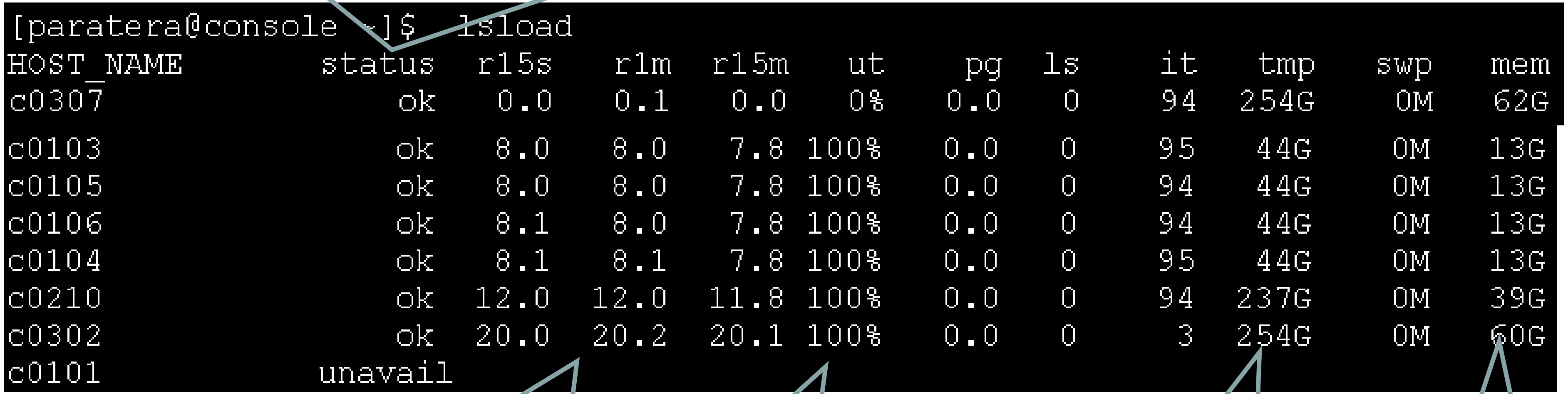
节点的核数

作业正在使用的核数

提交的该节点的作业核数

查看节点负载：lsload

节点负载状态：**ok表示正常，unavail表示节点未正常启动或lsf未启动，busy表示节点负载过高，如20个核的节点运行了40个vasp进程就会出现busy的情况**



```
[paratera@console ~]$  lsload
HOST_NAME          status  r15s   r1m  r15m    ut    pg  ls    it   tmp   swp   mem
c0307                  ok   0.0   0.1   0.0    0%   0.0   0    94  254G    0M   62G
c0103                  ok   8.0   8.0   7.8  100%   0.0   0    95   44G    0M   13G
c0105                  ok   8.0   8.0   7.8  100%   0.0   0    94   44G    0M   13G
c0106                  ok   8.1   8.0   7.8  100%   0.0   0    94   44G    0M   13G
c0104                  ok   8.1   8.1   7.8  100%   0.0   0    95   44G    0M   13G
c0210                  ok  12.0  12.0  11.8  100%   0.0   0    94  237G    0M   39G
c0302                  ok  20.0  20.2  20.1  100%   0.0   0     3  254G    0M   60G
c0101             unavail
```

r15s, r1m, r15m分别表示CPU队列长度的15秒，1分钟，15分钟平均值，正常情况下此值与该节点运行的作业使用的CPU核数接近

CPU利用率

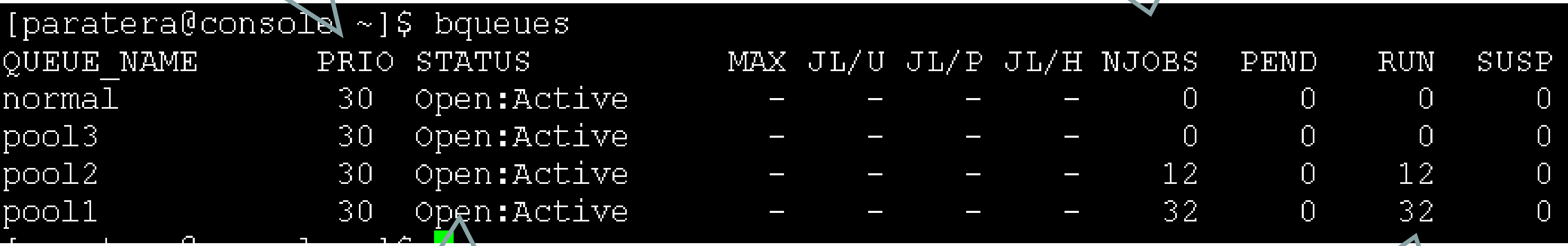
该节点/tmp目录剩余可使用存储空间

该节点剩余可使用内存

查看队列状态：bqueues

队列优先级

提交到该队列的作业核数

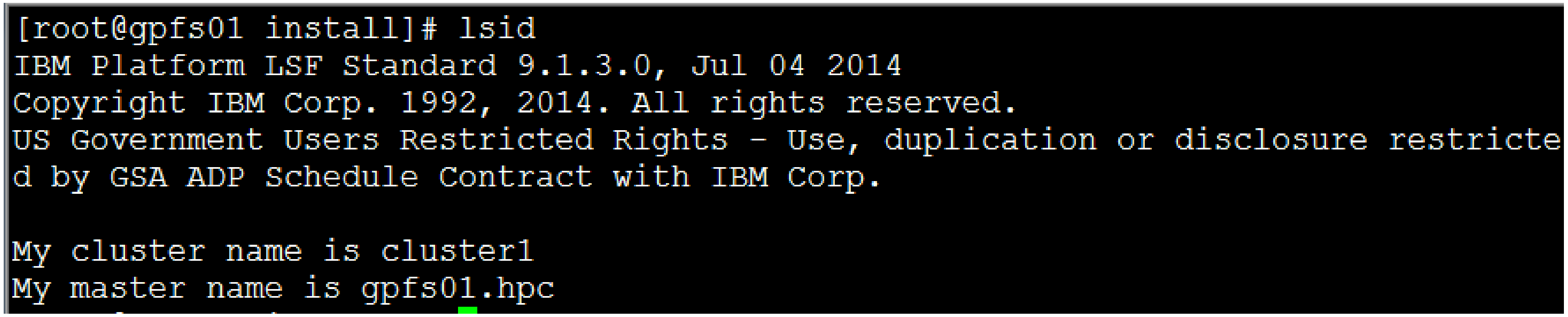


```
[paratera@console ~]$ bqueues
QUEUE_NAME      PRIO STATUS          MAX JL/U JL/P JL/H NJOBS  PEND   RUN  SUSP
normal           30  Open:Active       -    -    -    -     0     0     0     0
pool3            30  Open:Active       -    -    -    -     0     0     0     0
pool2            30  Open:Active       -    -    -    -    12     0    12     0
pool1            30  Open:Active       -    -    -    -    32     0    32     0
```

队列状态

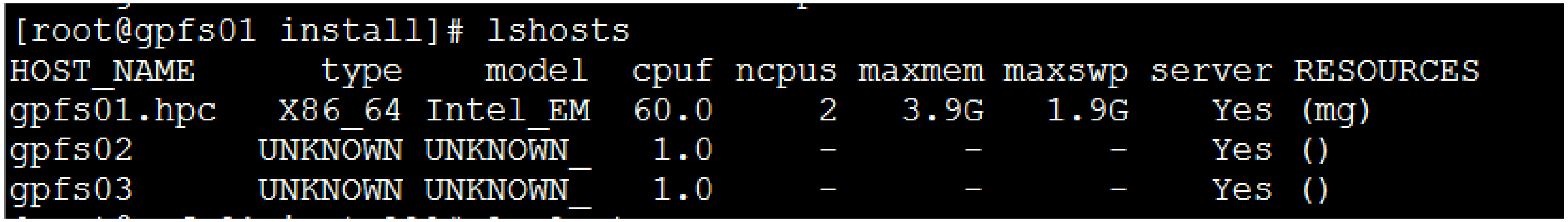
该队列运行作业的总CPU核数

查看集群状态：lsid



```
[root@gpfs01 install]# lsid
IBM Platform LSF Standard 9.1.3.0, Jul 04 2014
Copyright IBM Corp. 1992, 2014. All rights reserved.
US Government Users Restricted Rights - Use, duplication or disclosure restricte
d by GSA ADP Schedule Contract with IBM Corp.

My cluster name is cluster1
My master name is gpfs01.hpc
```

查看节点配置：lshosts



```
[root@gpfs01 install]# lshosts
HOST_NAME        type    model  cpuf ncpus maxmem maxswp server RESOURCES
gpfs01.hpc     X86_64 Intel_EM  60.0     2   3.9G   1.9G    Yes (mg)
gpfs02        UNKNOWN UNKNOWN_   1.0     -      -      -    Yes ()
gpfs03        UNKNOWN UNKNOWN_   1.0     -      -      -    Yes ()
```

想了解更多关于上述命令的参数，可使用-h参数或man, 如

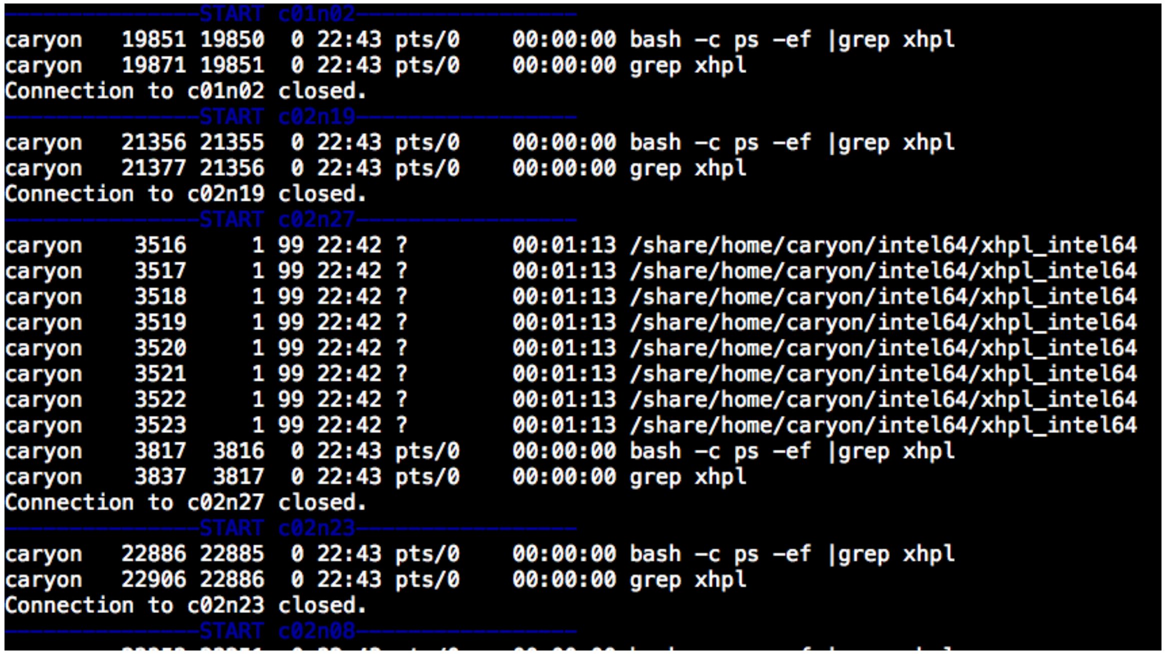
```
bqueues –h
```

```
man bqueues
```

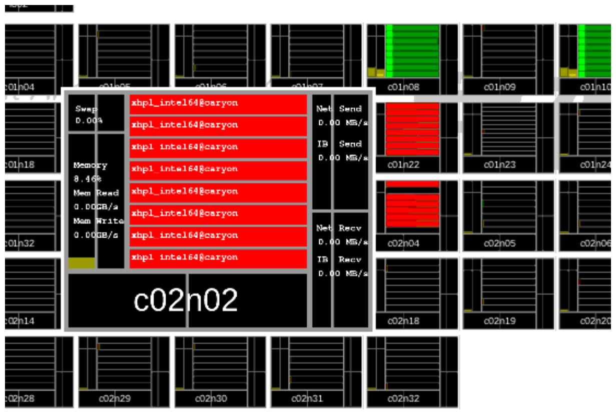
bkill <jobid>后作业有残留进程（开源软件弱点），怎么办？

bsub < run_xhpl.traditional.lsf
bpeek 2949
bkill 2949

/share/bin/ido.sh ./hostlist.2949 "ps -ef |grep xhpl"



```
-------------------START c01n02-------------------
caryon   19851 19850  0 22:43 pts/0    00:00:00 bash -c ps -ef |grep xhpl
caryon   19871 19851  0 22:43 pts/0    00:00:00 grep xhpl
Connection to c01n02 closed.
-------------------START c02n19-------------------
caryon   21356 21355  0 22:43 pts/0    00:00:00 bash -c ps -ef |grep xhpl
caryon   21377 21356  0 22:43 pts/0    00:00:00 grep xhpl
Connection to c02n19 closed.
-------------------START c02n27-------------------
caryon    3516     1 99 22:42 ?        00:01:13 /share/home/caryon/intel64/xhpl_intel64
caryon    3517     1 99 22:42 ?        00:01:13 /share/home/caryon/intel64/xhpl_intel64
caryon    3518     1 99 22:42 ?        00:01:13 /share/home/caryon/intel64/xhpl_intel64
caryon    3519     1 99 22:42 ?        00:01:13 /share/home/caryon/intel64/xhpl_intel64
caryon    3520     1 99 22:42 ?        00:01:13 /share/home/caryon/intel64/xhpl_intel64
caryon    3521     1 99 22:42 ?        00:01:13 /share/home/caryon/intel64/xhpl_intel64
caryon    3522     1 99 22:42 ?        00:01:13 /share/home/caryon/intel64/xhpl_intel64
caryon    3523     1 99 22:42 ?        00:01:13 /share/home/caryon/intel64/xhpl_intel64
caryon    3817  3816  0 22:43 pts/0    00:00:00 bash -c ps -ef |grep xhpl
caryon    3837  3817  0 22:43 pts/0    00:00:00 grep xhpl
Connection to c02n27 closed.
-------------------START c02n23-------------------
caryon   22886 22885  0 22:43 pts/0    00:00:00 bash -c ps -ef |grep xhpl
caryon   22906 22886  0 22:43 pts/0    00:00:00 grep xhpl
Connection to c02n23 closed.
-------------------START c02n08-------------------
```



/share/bin/ido.sh ./hostlist.2949 "killall -9 xhpl_intel64”

遍历计算节点，清除残留进程

相关配置文件

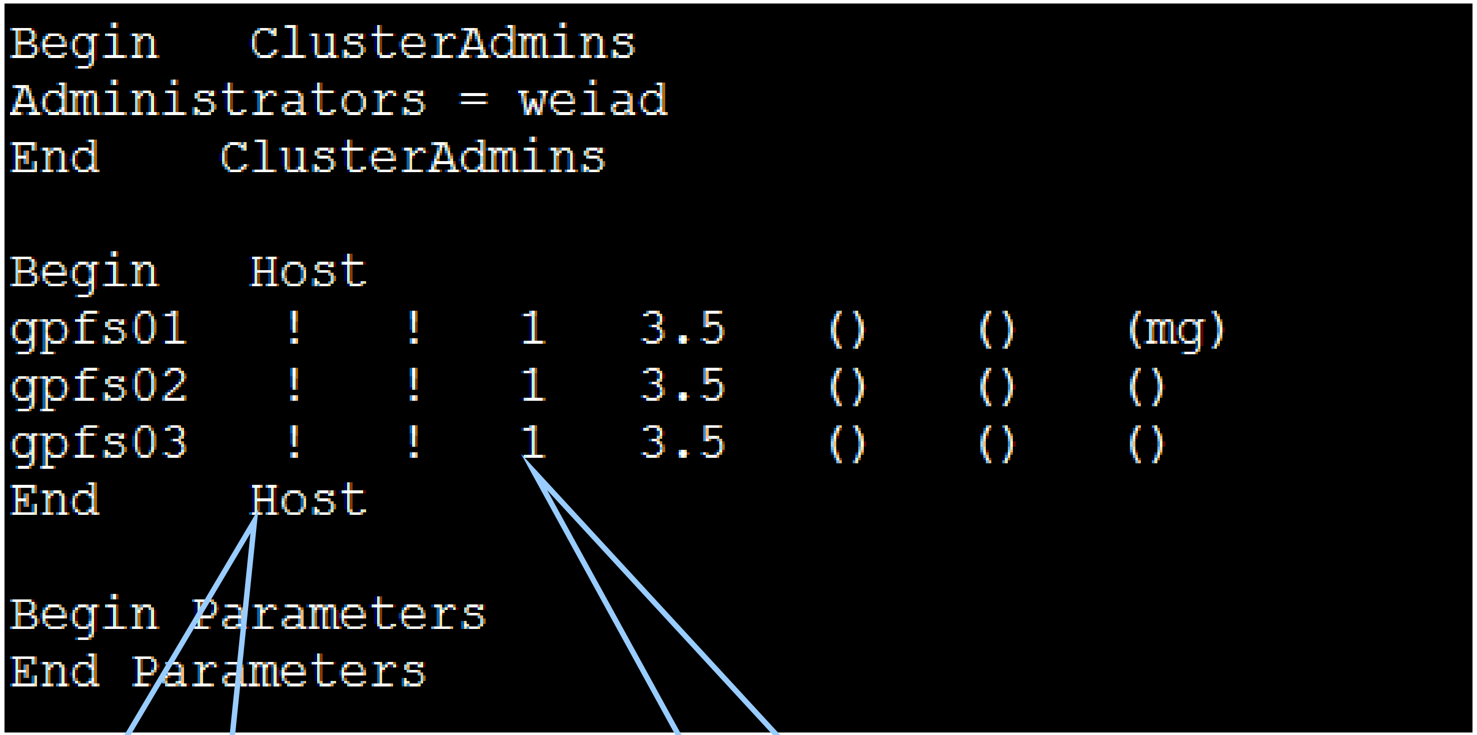
lsf.conf

所在目录：$LSF_TOP/conf
lsf安装后生产，保存了lsf的配置信息。lsf进程及一些lsf命令会读取此配置文件，修改此文件一定要小心谨慎。

lsf.cluster.*clustername*

所在目录：$LSF_TOP/conf
保存了lsf集群的配置信息，包括lsf管理员，lsf主机，lsf资源等信息：
内容见下页

若修改上述文件需执行
```
lsadmin reconfig
badmin reconfig
```
使之生效



```
Begin   ClusterAdmins
Administrators = weiad
End     ClusterAdmins

Begin   Host
gpfs01   !   !   1   3.5   ()   ()   (mg)
gpfs02   !   !   1   3.5   ()   ()   ()
gpfs03   !   !   1   3.5   ()   ()   ()
End     Host

Begin Parameters
End Parameters
```

lsf主机配置部分，
增加或删除lsf节点，
即修改此处

1表示为LSF server，
0表示为LSF client

lsb.queues

所在目录：$LSF_TOP/lsbatch/*$clustername*/configdir
此文件用来配置lsf队列信息。

lsb.hosts

所在目录：$LSF_TOP/lsbatch/*$clustername*/configdir
此文件用来配置lsf主机相关信息。

lsb.users

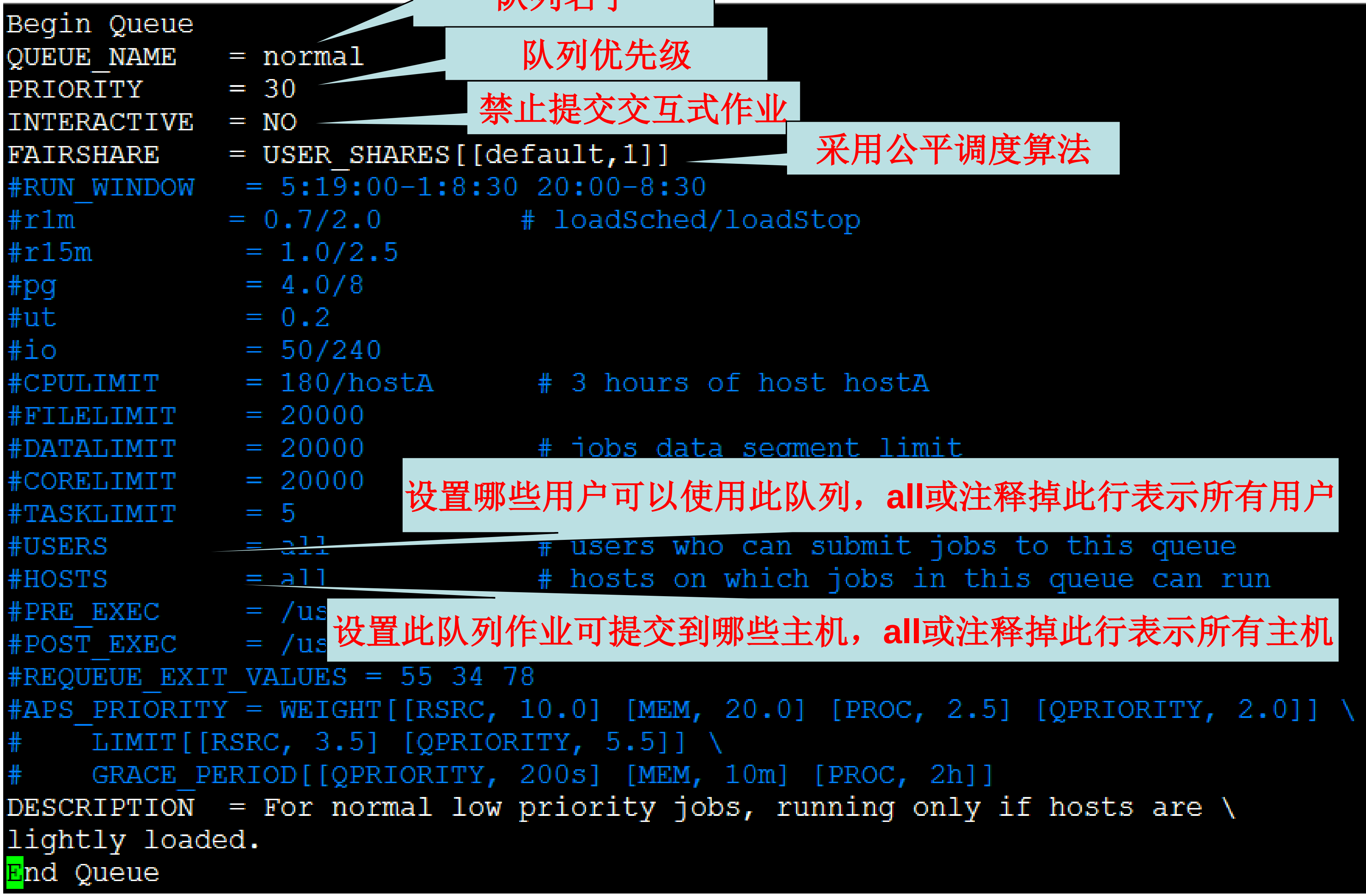
所在目录：$LSF_TOP/lsbatch/*$clustername*/configdir
此文件用来配置lsf用户相关信息。

lsb.applications

所在目录：$LSF_TOP/lsbatch/*$clustername*/configdir
此文件用来配置lsf应用相关信息。

若修改上述文件需执行
```
badmin reconfig
```
使之生效

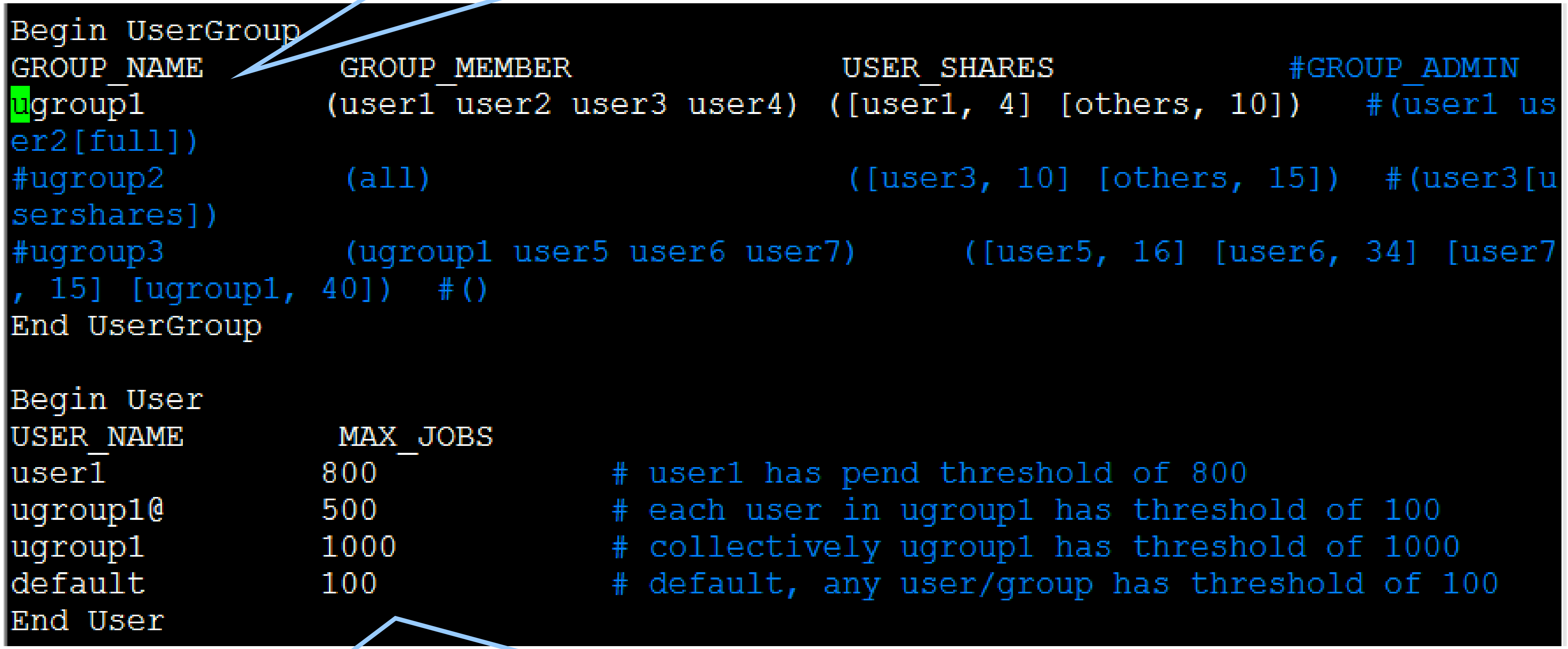
lsb.queues


队列名字
队列优先级
禁止提交交互式作业
采用公平调度算法
设置哪些用户可以使用此队列，all或注释掉此行表示所有用户
设置此队列作业可提交到哪些主机，all或注释掉此行表示所有主机

```
Begin Queue
QUEUE_NAME    = normal
PRIORITY      = 30
INTERACTIVE   = NO
FAIRSHARE     = USER_SHARES[[default,1]]
#RUN_WINDOW    = 5:19:00-1:8:30 20:00-8:30
#r1m          = 0.7/2.0        # loadSched/loadStop
#r15m          = 1.0/2.5
#pg            = 4.0/8
#ut            = 0.2
#io            = 50/240
#CPULIMIT      = 180/hostA      # 3 hours of host hostA
#FILELIMIT     = 20000
#DATALIMIT     = 20000          # jobs data segment limit
#CORELIMIT     = 20000
#TASKLIMIT     = 5
#USERS         = all            # users who can submit jobs to this queue
#HOSTS         = all            # hosts on which jobs in this queue can run
#PRE_EXEC      = /us
#POST_EXEC     = /us
#REQUEUE_EXIT_VALUES = 55 34 78
#APS_PRIORITY = WEIGHT[[RSRC, 10.0] [MEM, 20.0] [PROC, 2.5] [QPRIORITY, 2.0]] \
#     LIMIT[[RSRC, 3.5] [QPRIORITY, 5.5]] \
#     GRACE_PERIOD[[QPRIORITY, 200s] [MEM, 10m] [PROC, 2h]]
DESCRIPTION   = For normal low priority jobs, running only if hosts are \
lightly loaded.
End Queue
```

lsb.users

设置用户组及用户组成员



```
Begin UserGroup
GROUP_NAME        GROUP_MEMBER              USER_SHARES            #GROUP_ADMIN
ugroup1          (user1 user2 user3 user4) ([user1, 4] [others, 10])   #(user1 us
er2[full])
#ugroup2          (all)                     ([user3, 10] [others, 15])  #(user3[u
sershares])
#ugroup3          (ugroup1 user5 user6 user7)     ([user5, 16] [user6, 34] [user7
, 15] [ugroup1, 40])  #()
End UserGroup

Begin User
USER_NAME         MAX_JOBS
user1            800            # user1 has pend threshold of 800
ugroup1@         500            # each user in ugroup1 has threshold of 100
ugroup1          1000           # collectively ugroup1 has threshold of 1000
default          100            # default, any user/group has threshold of 100
End User
```

设置用户及用户组的最大可提交作业数，即**jobslots,**默认即为**cpu cores**