LSF作业调度系统的使用

李会民
hmli@ustc.edu.cn

中国科学技术大学 超级计算中心

2015年8月4日

1. LSF作业管理系统简介
2. 查看队列情况：bqueues
3. 超算中心超算系统现有队列
4. 提交作业：bsub
5. 管理作业：bkill、bstop、bresume、btop、bbot、bmod
6. 查看作业情况：bjobs、bpeek
7. 查看计算节点信息：lsload、bhosts
8. 查看用户信息：busers
9. 查看作业历史统计信息：bacct
10. LSF作业脚本
11. 联系信息

作业调度系统的用途

- 资源管理器：管理超算系统的硬件资源及认证信息等
- 队列管理器：管理当前已经提交但还未完成的作业
- 调度器：为作业分配资源
- 主要作用：
  - 根据用户作业提出的需求分配对应的资源给作业，告诉作业给它分配哪些节点等
  - 避免作业之间无序干扰，尽量让整个系统的负载一致
  - 保证用户占用资源的长期内公平

LSF作业管理系统的简介

- 当前超算中心的超算系统主要采用IBM公司[1]的Platform LSF进行资源和作业管理（IBM JS22刀片服务器除外）
- 所有需要运行的作业均必须通过作业提交命令*bsub*提交
- 为了利用*bsub*提交作业，需在*bsub*中指定各选项和要执行的程序
- 应提交到合适的队列
- 提交后可利用相关命令查询作业状态等
  - 系统LSF作业运行和排队情况：http://scc.ustc.edu.cn/yxjk/
  - 作业运行统计：http://scc.ustc.edu.cn/zytj/

注意：

- 登录节点主要用于日常操作，如提交作业、查看作业运行情况、编辑、编译、压缩/解压缩等
- 不要在登录节点直接运行作业，以免影响其余用户的正常使用
- 如不通过作业调度系统直接在计算节点上运行将会被监护直接杀掉

[1]以前叫Platform，已被IBM收购

查看队列情况：bqueues

利用bqueues可以查看现有队列信息，例如：

bqueues

QUEUE_NAME	PRIO	STATUS	MAX	JL/U	JL/P	JL/H	NJOBS	PEND	RUN	SUSP
normal	30	Open:Active	-	8	-	-	22	2	20	0
long	30	Open:Active	-	304	-	-	52	12	40	0

- QUEUE_NAME：队列名
- PRIO：优先级，数字越大优先级越高
- STATUS：状态
  - Open:Active表示已激活，可使用
  - Closed:Active表示已关闭，不可使用
- MAX：队列对应的最大CPU核数，-表示无限，以下类似
- JL/U：单个用户同时可以的CPU核数
- NJOBS：排队、运行和被挂起的总作业所占CPU核数
- PEND：排队中的作业所需CPU核数
- RUN：运行中的作业所占CPU核数
- SUSP：被挂起的作业所占CPU核数

查看队列详细情况：bqueues -l

- 队列也许会调整
- 请注意登录系统后的提示
  - 登录后可以在登录节点执行*cat␣/etc/motd*查看
- 请利用*bqueues␣-l*查看各队列的详细情况

```
QUEUE: normal
  -- For normal priority jobs, allow 2-8 CPU cores, Max Job Slots is 40 This is the default queue.

PARAMETERS/STATISTICS
PRIO  NICE  STATUS          MAX  JL/U  JL/P  JL/H  NJOBS   PEND    RUN  SSUSP  USUSP   RSV
 30    20   Open:Active       -    40     -     -    288     48    240      0      0     0

CPULIMIT                    PROCLIMIT
 345600.0 min of E5410       2 2 8
 PROCESSLIMIT
       8
```

- CPULIMIT：单个作业运行时间限制，以系统中的某个节点E5410作为参考，运行机时（核数*墙上时间）为345600.0 CPU分钟，即如用8 CPU核计算，允许运行30天
- PROCLIMIT：单个作业核数限制，2 4 8，表示使用此队列时，最少使用2个核，最多使用8核，如提交时没用-n指定具体核数，那么使用默认4核
- PROCESSLIMIT：单个作业最大核数限制，为8

现有队列：联想7000G GPU集群

- CPU计算队列：
  - small：可运行12~64进程并行作业，提交作业时需加参数-q small
  - large：可运行64~128进程并行作业，提交作业时需加参数-q large
- GPU计算队列[2]：
  - c2050：运行于node29~node44节点，每节点1颗C2050 GPU卡，提交作业时加参数-q c2050
  - c1060：运行于node1~node16节点，每节点2颗C1060 GPU卡，提交作业时加参数-q c1060
  - gtx295：运行于node19~node28节点，每节点2颗GTX295 GPU卡，提交作业时加参数-q gtx295

建议以8的倍数申请核数，以尽量独占单个节点，避免作业间相互干扰

[2]非GPU作业不得使用GPU队列

现有队列：刀片及胖节点超级计算系统 I

- serial：1~2进程作业队列，提交作业时需加参数-q serial，每用户可最多同时运行24个serial队列作业
- normal：仅运行10~12 CPU核的作业，运行于node1~node80其中之一节点，提交作业时需加参数-q normal
- long：可运行大于12 CPU核的多节点并行作业，运行于node1~node80节点，提交作业时需加参数-q long，每用户最多可运行120进程的作业
- mem48：可运行每进程需求内存较大但一个作业总需求内存小于48GB的作业，运行于node71~node80某一节点上，提交作业时需加参数-q mem48，队列优先级较高
- mem64：可运行每进程需求内存较大但一个作业总需求内存小于64GB的作业，运行于node89~node90某一节点上，提交作业时需加参数-q mem64，队列优先级较高

现有队列：刀片及胖节点超级计算系统 II

- mem96：可运行每进程需求内存较大但一个作业总需求内存小于96GB的作业，运行于node91~node92某一节点上，提交作业时需加参数-q mem96，队列优先级较高
- fat48：运行大共享内存作业，每节点48 CPU核，提交作业时需加参数-q fat48 -R ”rusage[mem=**]”，**为以MB为单位的每进程内存数，需特殊申请使用权限
- fat64：运行大共享内存作业，每节点64 CPU核，提交作业时需加参数-q fat64 -R ”rusage[mem=**]”，**为以MB为单位的每进程内存数，需特殊申请使用权限
- 8cpu：作业运行在node93~node102节点上，作业设置请使用进程数为8的倍数，提交作业时需加参数-q 8cpu

normal和long队列，建议以12的倍数申请核数，以尽量独占单个节点，避免作业间相互干扰

现有队列：ChinaGrid高性能计算集群

- normal：每用户最多可同时运行16 CPU的作业，单作业最少8 CPU，最多16 CPU，所有用户在此队列最多同时使用160 CPU核
- long：每用户最多可同时运行128 CPU的作业，单作业最少24 CPU，最多128 CPU，所有用户在此队列最多同时使用704 CPU核
- mic：仅限MIC程序使用

normal和long队列同一用户同时最多使用160作业进程数，作业最大运行时间15天，bqueues -l查看各队列设置的详细说明

现有队列：曙光TC4600百万亿次超级计算系统

- testjob：本队列仅供免费测试作业是否可正常计算，进程数限制为1~48，正常计算的程序需提交到其他队列开始正式计算。对付费排队用户不计算机时费用
- small：付费排队队列，可运行1~24 CPU核进程作业，提交作业时需加参数-q small
- normal：付费排队队列，可运行48~192CPU核进程作业，提交作业时需加参数-q normal
- long：付费排队队列，可运行193~1200CPU核进程作业，提交作业时需加参数-q long
- bcpan、zyli、kmh、zzyzhang、qiao、houdong、thye：独占节点队列，仅相应组内用户可使用对应的队列
- ahedu：安徽省科协协作平台的校外高校用户使用队列
- qsce：中科院网格用户使用队列

提交作业：bsub

用户需要利用bsub提交作业，其基本格式为：

bsub [options] command [arguments]

- command之前的options：设置队列、CPU核数等LSF的选项
- command之后的arguments：设置作业的可执行程序本身所需要的参数
- 作业提交后，应经常检查一下作业的CPU、内存等利用率，判断实际运行效率
  - 可以ssh到对应运行作业的节点运行*top*命令
  - 查看Ganglia系统监控：http://scc.ustc.edu.cn/ganglia
- 请不要ssh到节点后直接运行作业，将会被监控自动杀掉

提交到特定队列：bsub -q

- 利用-q选项可以指定提交到哪个队列
- 如不加-q，那么将提交到系统设置的默认队列[3]
- 提交到serial队列运行串行程序executable1：

  bsub␣-q␣serial␣executable1

  如果提交成功，将显示类似下面的输出：

```
Job <79722> is submitted to default queue <serial>.
```

其中79722为此作业的作业号，以后可利用此作业号来进行查询及终止等操作。

[3]除非了解哪个是默认队列，且默认队列适合此作业，否则不要这么做

指明所需要的CPU核数： bsub -n

- 利用-n选项指定所需要的CPU核数（一般来说核数和进程数一致）
- 为了用户作业间不相互干扰，申请的核数最好为系统节点内CPU核数的整数倍，以便同一个作业占据整个节点
  - 比如对每个节点为8核的系统，申请核数为8的整数倍，节点核数为12的系统，申请核数为12的整数倍
  - 曙光TC4600百万亿次超级计算系统：每个节点24 CPU核
  - ChinaGrid高性能计算集群：每个节点16 CPU核
  - 联想1800和7000G GPU集群：每个节点8 CPU核
  - 刀片及胖节点超级计算系统：
    - long队列和normal队列每个节点12 CPU核
    - 其它队列请看队列对应的节点配置
  - 以上仅仅是建议，具体申请核数应考虑作业实际情况[4]

[4]即使同一个计算软件，在计算不同条件时，也有可能不一样，请务必仔细研究自己所使用的软件

MPI作业的提交：bsub mpijob

- MPI作业一般需要用提交时使用mpijob来执行MPI程序，可以使用normal、long等队列
- 指定利用八个CPU核（由-n 8指定）运行MPI程序：

  bsub␣-q␣normal␣-n␣8␣mpijob␣executable-mpi1

注：LSmpijob命令适合不通过作业调度系统时直接以*mpirun*或*mpiexec*方式运行的作业。

OpenMP等共享内存作业的提交：bsub OMP_NUM_THREADS

- 需要保证在同一个节点内运行，只能使用normal等队列，不能使用long等队列
- 程序启动前利用OMP_NUM_THREADS设定制定的线程数，一般应与申请的核数一致
- 指定利用八个CPU核运行OpenMP程序：

  bsub␣-q␣normal␣-n␣8␣OMP_NUM_THREADS=8␣executable -on

MIC作业的提交

- MIC作业必需使用MIC队列才可以保证运行的作业在MIC节点上运行，当前只有ChinaGrid高性能计算集群支持MIC，当前可以使用的队列为mic
- 因为与MPI结合的GPU程序要求不统一，现在系统尚未提供专有的GPU和MPI结合程序的脚本供大家来调用，如需要，请与我们联系，我们来处理。如果对LSF熟悉，也可以自己写所需要的脚本

GPU作业的提交

- GPU作业必需使用GPU队列才可以保证运行的作业在GPU节点上运行，当前只有联想7000G GPU集群支持GPU，当前可以使用的队列为c2050、c1060和gtx295
- 因为与MPI结合的GPU程序要求不统一，现在系统尚未提供专有的GPU和MPI结合程序的脚本供大家来调用，如需要，请与我们联系，我们来处理。如果对LSF熟悉，也可以自己写所需要的脚本

串行作业的提交：bsub -q serial

- 运行串行作业，请使用serial队列：
  bsub -q serial executable-serial
- 科大超算中心鼓励并行作业，因此给串行作业的资源少，请尽量用自己的系统运行串行作业，在科大超算平台上运行并行作业

运行排他性运行作业：bsub -x

- 如果需要独占节点运行，此时需要添加-x选项：
  bsub -x -q normal -n 4 executable-omp1
- 注意：
  - 排他性运行在运行期间，不允许其余的作业提交到运行此作业的节点，并且只有在某节点没有任何其余的作业在运行时才会提交到此节点上运行
  - 如果不需要采用排他性运行，请不要使用此选项，否则将导致作业必须等待完全空闲的节点才会运行，也许将增加等待时间
  - 另外使用排他性运行时，哪怕只使用某节点内的一个CPU核，也将按照此节点内的所有CPU核数进行机时计算

指明输入、输出文件运行：bsub -i -o -e

- 作业的屏幕输入文件、正常屏幕输出到的文件和错误屏幕输出的文件可以利用-i、-o和-e选项来分别指定，运行后可以通过查看指定的这些输出文件来查看运行状态，文件名可利用%J与作业号挂钩
- 屏幕输入文件指的是存储程序运行时需要手动在屏幕上输入的内容的，其内容可以利用<将此文件中的内容重定向以代替手动屏幕输入传递给可执行程序的，并不是指的程序本身自带的输入文件，如不通过作业调度系统时的提交方式为：
  - executable1 < file1，可以用下述方式提交
    bsub␣-i␣file1␣executable
  - executable1 file1或executable1 -i file1等，则不可用下述方式提交
    bsub␣-i␣file1␣executable
- 如指定executable1的屏幕输入、正常和错误屏幕输出文件分别为executable1.input、executable1-%J.log和executable1-%J.err：
  bsub -i executable1.input -o executable1-%J.log -e executable1-%J.err executable1

交互式运行作业： bsub -I

- 如需运行交互式的作业（如在运行期间需手动输入参数等），需结合-I、-Ip和-Is等参数
- 建议只在调试期间使用，一般作业尽量不要使用此选项

bsub -I executable1

终止作业：bkill

- 利用bkill命令可以终止某个运行中或者排队中的作业，如：
  bkill 79722

```
Job <79722> is being terminated
```

- 请及时终止有问题或无需再运行的程序，空出计算资源，降低费用

挂起作业：bstop

- 利用bstop命令可临时挂起某个作业以让别的作业先运行，例如：
  bstop 79727

```
Job <79727> is being stopped.
```

  - 可以将排在队列前面的作业临时挂起，以让后面的作业先运行
  - 虽然也可以作用于运行中的作业，但并不会因为此作业被挂起而允许其余作业占用此作业所占用的CPU运行，实际资源不会释放，建议不要随便对运行中的作业进行挂起操作
  - 如果运行中的作业不再想继续运行，请用bkill终止

继续运行被挂起的作业： bresume

- 利用bresume命令可继续运行某个挂起某个作业，例如：
  bresume 79727

```
Job <79727> is being resumed.
```

设置作业最先运行：btop

- 利用btop命令可最先运行排队中的某个作业，例如：
  btop 79727
- 运行成功后，将显示类似下面的输出：

```
Job <79727> has been moved to position 1 from top.
```

设置作业最后运行：bbot

- 利用bbot命令可设定最后运行排队中的某个作业，例如：
  bbot 79727

```
Job <79727> has been moved to position 1 from bottom.
```

修改排队中的作业选项：bmod

- 利用bmod命令可修改排队中的某个作业的选项，如想将排队中的作业号为79727的作业的执行命令修改为executable2并且换到long队列，并且所需要CPU核数为12：

  bmod -Z executable2 -q long -n 12 79727

```
Parameters of job <79727> are being changed.
```

查看作业的排队和运行情况：bjobs

- 利用bjobs可以查看作业的运行情况，例如：

bjobs

```
JOBID  USER  STAT  QUEUE   FROM_HOST  EXEC_HOST  JOB_NAME    SUBMIT_TIME
79726  hmli  RUN   normal  user       2*node31   *executab1  Mar 12 19:20
                                      1*node4
79727  hmli  PEND  long    user                  *executab2  Mar 12 19:20
```

显示：

- 作业79726分别在node31和node4上运行2、1个进程
- 作业79727处于排队中尚未运行

查看作业详细信息：bjobs -l

- 查看作业的详细信息-l选项：
  bjobs -l 79727

```
Job Id <79727>, User <hmli>, Project <default>, Status <PEND>,
Queue <long> , Command <executab2>
Sun Mar 12 14:15:07: Submitted from host <hpc1.ustc.edu.cn>,
CWD <$HOME>, Requested Resources <type==any && swp>35>;
PENDING REASONS:
The user has reached his/her job slot limit;
SCHEDULING PARAMETERS:
            r15s r1m r15m   ut pg    io   ls    it   tmp   swp   mem
loadSched   -     0.7 1.0   - 4.0   -    -     -    -     -     -
loadStop    -     1.5 2.5   - 8.0   -    -     -    -     -     -
```

注意：从PENDING REASONS可以看出为什么还在排队等待中。

查看作业仍在排队等待的原因：bjobs -p

- 查看作业仍在排队等待的原因可以利用-p选项：
  bjobs -p 79727

```
The user has reached his/her job slot limit;
```

上述显示达到了用户自己的作业数等限制

- 如发现很多节点空闲，自己的作业又没有达到限制，感觉应该运行而没有运行，也许是系统存在问题，请与管理员联系处理

查看运行中作业的屏幕正常输出：bpeek

- 利用*bpeek*命令可查看运行中作业的屏幕正常输出，例如：
  bpeek 79727

```
<< output from stdout >>
Radius(nm): 300.000
```

- *bpeek␣-f 作业号*，可以连续查看作业的连续屏幕输出
- 如在运行中用-o和-e分别指定了正常和错误屏幕输出，也可以通过直接查看指定的文件的内容来查看屏幕输出

查看各节点的运行情况：lsload

- 利用*lsload*命令可查看当前各节点的运行情况，例如：

lsload

```
HOST_NAME  status  r15s  r1m  r15m  ut  pg  ls  it    tmp    swp     mem
node10     ok      0.0   0.0  0.0   0%  3.5  0  2050  9032M  4000M  16G
node11     locku   0.0   0.0  0.0   0%  3.5  0  2050  9032M  4000M  16G
```

ut列表示利用率，status列中的locku表示在进行排他性运行

查看各节点的空闲情况：bhosts

- 利用*bhosts*命令可查看当前各节点的空闲情况，例如：

bhosts

HOST_NAME	STATUS	JL/U	MAX	NJOBS	RUN	SSUSP	USUSP	RSV
node12	closed	–	4	2	2	0	0	0
node10	ok	–	2	2	1	0	0	0

STATUS列中的ok表示可以接收新作业，closed表示已经被占满

查看用户信息：busers

- 利用*busers*可以查看用户信息，例如：
  busers hmli

USER/GROUP	JL/P	MAX	NJOBS	PEND	RUN	SSUSP	USUSP	RSV
hmli	–	22	40	32	8	0	0	0

  - MAX最大可以同时运行的核数
  - NJOBS当前所有运行和待运行作业所需的核数
  - PEND排队等待运行的作业所需要的核数
  - RUN已经开始运行的作业占据的核数

查看作业历史统计信息：bacct

利用bacct可以查看已经结束的作业的历史统计信息，如：

bacct

```
Accounting information about jobs that are:
  - submitted by users hmli,
  - accounted on all projects.
  - completed normally or exited
  - executed on all hosts.
  - submitted to all queues.
  - accounted on all service classes.
------------------------------------------------------------------------------
SUMMARY:      ( time unit: second )
 Total number of done jobs:       73      Total number of exited jobs:    159
 Total CPU time consumed:    14649779.0     Average CPU time consumed: 63145.6
 Maximum CPU time of a job: 4266155.5      Minimum CPU time of a job:      0.0
 Total wait time in queues: 1403570.0
 Average wait time in queue: 6049.9
 Maximum wait time in queue:904361.0      Minimum wait time in queue:     2.0
 Average turnaround time:      11671 (seconds/job)
 Maximum turnaround time:     904480      Minimum turnaround time:          2
 Average hog factor of a job:  7.12 ( cpu time / turnaround time )
 Maximum hog factor of a job:  157.84      Minimum hog factor of a job:  0.00
 Total throughput:              0.02 (jobs/hour)  during10055.28 hours
 Beginning time:       Jan 30 14:40      Ending time:          Mar 25 13:57
```

查看某个作业历史统计信息： bacct -l jobid

bacct␣-l␣13624

```
------------------------------------------------------------------------------
Job <13624>, User <hmli>, Project <default>, Status <DONE>, Queue <long>, Comma
                     nd <mpijob /opt/bin/vasp-2013.12.27>
Mon Mar 24 16:43:24: Submitted from host <chinagrid>, CWD <$HOME/qiao>, Output
                     File <%J.log>, Error File <%J.err>;
Tue Mar 25 13:55:15: Dispatched to 32 Hosts/Processors <16*node26> <16*node12>;
Tue Mar 25 13:57:07: Completed <done>.
Accounting information about this job:
     CPU_T     WAIT     TURNAROUND   STATUS     HOG_FACTOR    MEM    SWAP
    984.17    76311          76423     done         0.0129    30G     39G
------------------------------------------------------------------------------
SUMMARY:      ( time unit: second )
 Total number of done jobs:       1      Total number of exited jobs:     0
 Total CPU time consumed:       984.2    Average CPU time consumed:   984.2
 Maximum CPU time of a job:     984.2    Minimum CPU time of a job:   984.2
 Total wait time in queues: 76311.0
 Average wait time in queue:76311.0
 Maximum wait time in queue:76311.0      Minimum wait time in queue:76311.0
 Average turnaround time:       76423 (seconds/job)
 Maximum turnaround time:       76423    Minimum turnaround time:      76423
 Average hog factor of a job:    0.01 ( cpu time / turnaround time )
 Maximum hog factor of a job:    0.01    Minimum hog factor of a job:    0.01
------------------------------------------------------------------------------
```

查看某时间段内作业历史统计信息：bacct -C -D -S

在2014/03/01,2014/04/01时间段内：

- 完成的：`bacct -l -C 2014/03/01,2014/04/01`
- 开始运行的：`bacct -l -D 2014/03/01,2014/04/01`
- 提交的：`bacct -l -S 2014/03/01,2014/04/01`

LSF作业脚本

- 以在LSF脚本中设置队列等参数方式提交，如*my_script.lsf*

```
#!/bin/sh
#BSUB -q long
#BSUB -o %J.log -e %J.err
#BSUB -n 66
mpijob ./mympi-prog
```

- 不得以直接*./my_script.lsf*等常规脚本运行方式运行
- 需要传递给*bsub*命令运行：*bsub␣<␣my_script.lsf*
- 如果bsub后面更-q等LSF参数，将会覆盖掉LSF脚本中的设置
- 一般用户，没必要写此类脚本，直接通过命令行传递LSF参数即可。对于当前设置满足不了作业需求，且用户比较了解LSF中的各规定，对shell脚本编写比较在行，那么用户完全可自己编写脚本提交作业，比如提交特殊需求的MPI与GPU结合的作业。

LSF作业脚本常见变量

主要有以下变量比较常用，在作业运行后，这些变量存储对应的作业信息，具体的请参看LSF官方手册：

- *LS_JOBPID*：作业进程号
- *LSB_HOSTS*：存储系统分配的节点名
- *LSB_JOBFILENAME*：作业脚本文件名
- *LSB_JOBID*：作业号
- *LSB_QUEUE*：作业队列
- *LSB_JOBPGIDS*：作业进程组号组
- *LSB_JOBPIDS*：作业进程号组

LSF官方资料：http://scc.ustc.edu.cn/zlsc/lsf/

联系信息

- 中国科大超算中心：
  - 办公室：科大东区新图书馆一楼东侧126室
  - 电话：0551-63602248
  - 信箱：sccadmin@ustc.edu.cn
  - 主页：http://scc.ustc.edu.cn
- 李会民：
  - 电话：0551-63600316
  - 信箱：hmli@ustc.edu.cn
  - 主页：http://hmli.ustc.edu.cn