瓦谢尔计算生物研究院
高性能计算平台培训会

|2022.11.10|

A

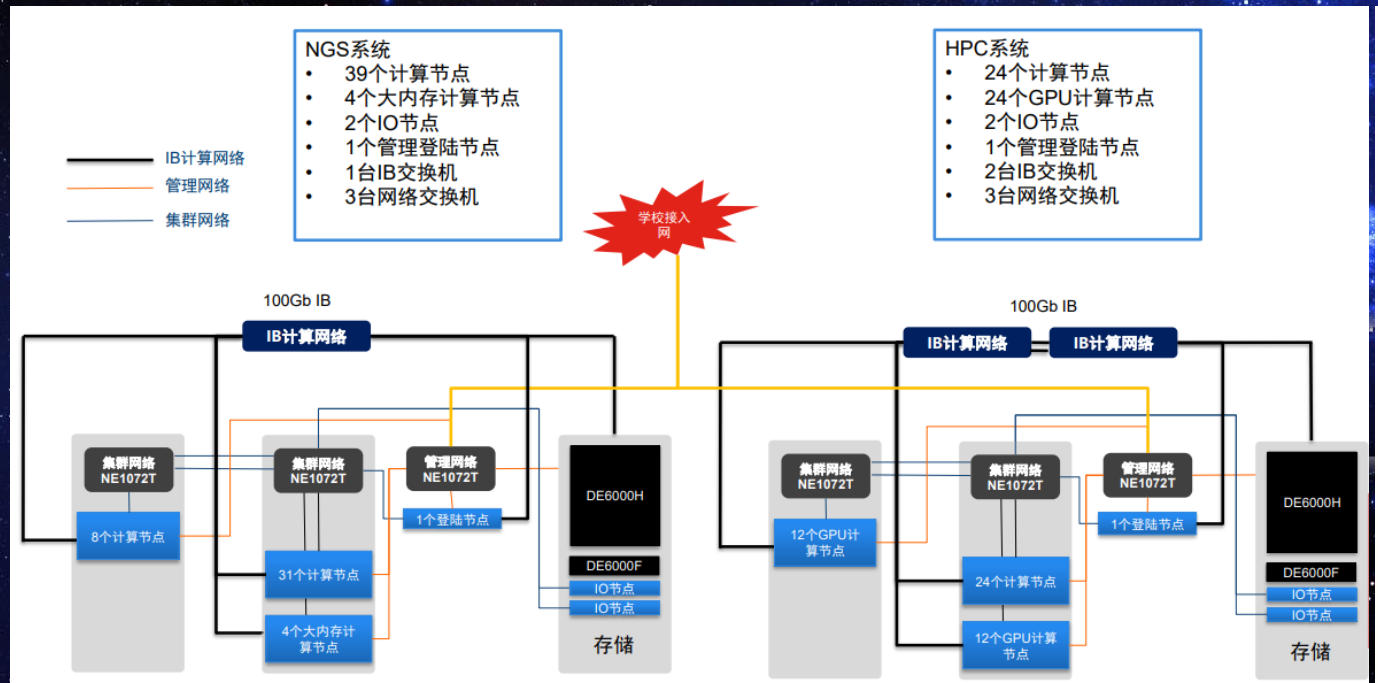
集群信息

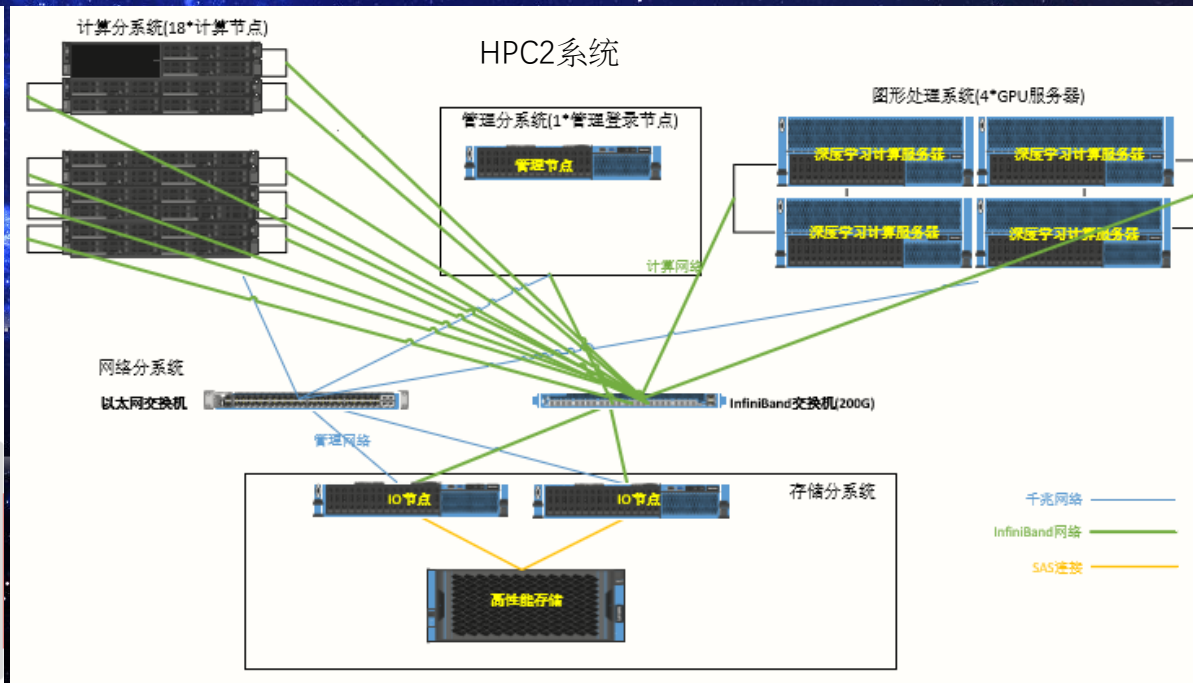
集群信息

NGS集群
计算节点、大内存计算节点、存储
40TB high-iops Flash storage, 1.4PB high-bandwidth NL-SAS storage
100Gb IB

HPC集群
计算节点、GPU计算节点(2080Ti)、存储
40TB high-iops Flash storage, 1.4PB high-bandwidth NL-SAS storage
100Gb IB

HPC2集群
计算节点、GPU计算节点(A100)、存储
24TB high-iops Flash storage, 685TB high-bandwidth NL-SAS storage
200Gb IB





NGS集群信息

NGS cluster consists four type of block:

- Login node: 1x 2-sockets nodes
- General Purpose: 39x 2-sockets nodes
- Big Memory: 4x 8-sockets with 6TB memory
- Storage: 40TB high-iops Flash storage, 1.4PB high-bandwidth NL-SAS storage
- Network: Mellanox EDR Infiniband

NGS集群信息

NGS cluster hardware configuration:

ComputeNodes	Thin Nodes	Fat Nodes
Processor	Intel(R) Xeon(R) Gold 6248 CPU @ 2.50GHz	Intel(R) Xeon(R) Platinum 8253 CPU @ 2.20GHz
Cores per Node	40	128
Memory per node (GByte)	192GB	6T
Number of Nodes	39	4
Number of Cores	1560	512

NGS集群信息

Filesystems:

Storage subsystem consists of two SR650 nodes, directly attacthed 1x DE6000F with 18x 3.84TB SSD and 1x DE6000H with 200x 10TB NL-SAS hard drive

mount point	Capacity
/data	1.4PB
/share	40TB

NGS集群信息

管理/登录节点（ngs-mn01）：NGS/AI集群的核心，承担集群管理、监控、调度、策略管理以及用户和帐户管理等主要功能。将集群连接到外部网络或集群。用户必须使用登录节点登录和上传应用程序数据、开发编译器以及提交调度任务。

计算节点（c001-c039、s001-s004）：完成计算任务。

并行文件系统(GPFS): 提供共享存储功能。其通过高速网络连接到集群节点。

管理网络:集群带内管理网。

高速网络：用于支持并行文件系统，传输计算数据。

NGS集群信息

Type	Description
Operation System	CentOS Linux release 7.6.1810
Batch Scheduling System	IBM Spectrum LSF 10.2.0.6 Workgroup Edition
High Performance Parallel Filesystem	IBM Spectrum Scale 5.0.4.0
Programming Environment	Intel Parallel Studio XE, GNU compilers
Parallel Environment	OpenMPI, IntelMPI, etc.
GPU Environment	CUDA-9.2, CUDA-10.1, CUDA-10.2

HPC集群信息

HPC cluster consists four type of block:

- Login node: 1x 2-sockets nodes
- General Purpose: 24x 2-sockets nodes
- GPU node: 24x 8GPU nodes
- Storage: 40TB high-iops Flash storage, 1.4PB high-bandwidth NL-SAS storage
- Network: Mellanox EDR Infiniband

HPC集群信息

HPC cluster hardware configuration:

ComputeNodes	Thin Nodes	GPU Nodes
Processor	Intel(R) Xeon(R) Gold 6248 CPU @ 2.50GHz	Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz
Cores per Node	40	32
Memory per node (GByte)	192GB	192GB
GPU	None	8x GeForce RTX 2080 Ti
Number of Nodes	24	24
Number of Cores	960	768

HPC集群信息

Filesystems:

Storage subsystem consists of two SR650 nodes, directly attacthed 1x DE6000F with 18x 3.84TB SSD and 1x DE6000H with 200x 10TB NL-SAS hard drive

mount point	Capacity
/data	1.4PB
/share	40TB

HPC集群信息

管理/登录节点（hpc-mn01）：HPC/AI集群的核心，承担集群管理、监控、调度、策略管理以及用户和帐户管理等主要功能。将集群连接到外部网络或集群。用户必须使用登录节点登录和上传应用程序数据、开发编译器以及提交调度任务。

计算节点（h001-h024、gpu01-24）：完成计算任务。

并行文件系统(GPFS): 提供共享存储功能。其通过高速网络连接到集群节点。

管理网络:集群带内管理网。

高速网络：用于支持并行文件系统，传输计算数据。

HPC集群信息

Type	Description
Operation System	CentOS Linux release 7.6.1810
Batch Scheduling System	IBM Spectrum LSF 10.2.0.6 Workgroup Edition
High Performance Parallel Filesystem	IBM Spectrum Scale 5.0.4.0
Programming Environment	Intel Parallel Studio XE, GNU compilers
Parallel Environment	OpenMPI, IntelMPI, etc.
GPU Environment	CUDA-9.2, CUDA-10.1, CUDA-10.2

HPC2集群信息

HPC2 cluster consists four type of block:

- Login node: 1x 2-sockets nodes
- General Purpose: 16x 2-sockets nodes
- GPU node: 4x 4GPU nodes
- Storage: 24TB high-iops Flash storage, 685TB high-bandwidth NL-SAS storage
- Network: Mellanox HDR 100 Infiniband

HPC2集群信息

HPC2 cluster hardware configuration:

ComputeNodes	CPU Nodes	GPU Nodes
Processor	Intel(R) Xeon(R) Gold 6240R CPU @ 2.40GHz	Intel(R) Xeon(R) Gold 6326 CPU @ 2.90GHz
Cores per Node	48	32
Memory per node (GByte)	192GB	256GB
GPU	None	4 x NVIDIA A100 40GB
Number of Nodes	16	4
Number of Cores	768	128

HPC2集群信息

Filesystems:

Storage subsystem consists of two SR655 nodes, directly attacthed 1x DE6000H with **60x 16TB NL-SAS** hard drive

mount point	Capacity
/scratch	24TB
/share	685TB

HPC2集群信息

管理/登录节点（hpc2-mn01）：HPC2/AI 集群的核心，承担集群管理、监控、调度、策略管理以及用户和帐户管理等主要功能。将集群连接到外部网络或集群。用户必须使用登录节点登录和上传应用程序数据、开发编译器以及提交调度任务。

计算节点（c001-c016、gpu01-04）：完成计算任务。

并行文件系统(GPFS): 提供共享存储功能。其通过高速网络连接到集群节点。

管理网络:集群带内管理网。

高速网络：用于支持并行文件系统，传输计算数据。

HPC2集群信息

名称	版本
操作系统	CentOS 7.9
编译器	GCC/Intel
作业调度	Spectrum LSF 10.1
文件系统	Spectrum Scale 5.1
集群管理	LICO 5.5 + xCAT

B

登录使用

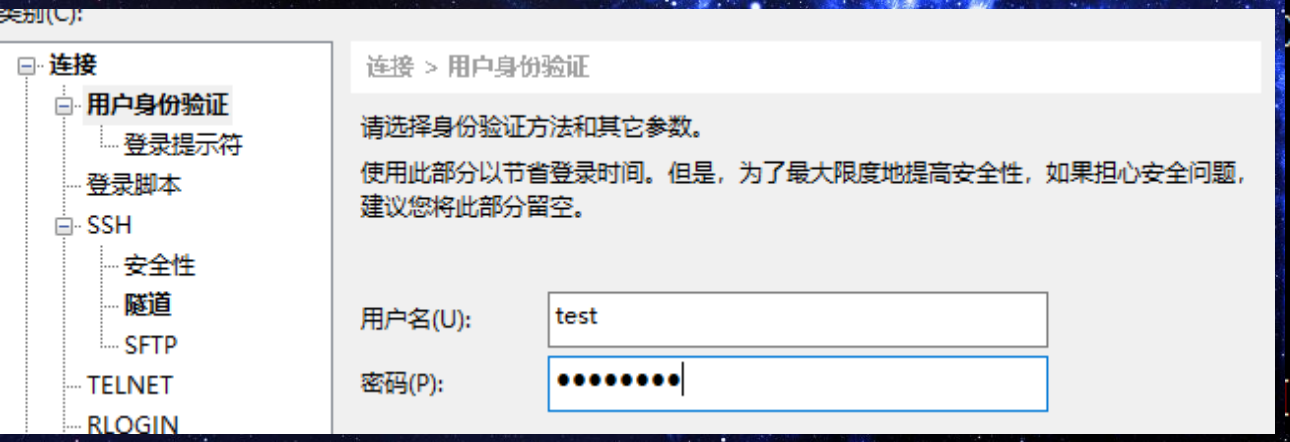
登录使用

登录信息

主机名	IP地址
hpc2-mn01	10.26.4.249
hpc-mn01	10.26.4.250
ngs-mn01	10.26.4.251

如何登录 Xshell

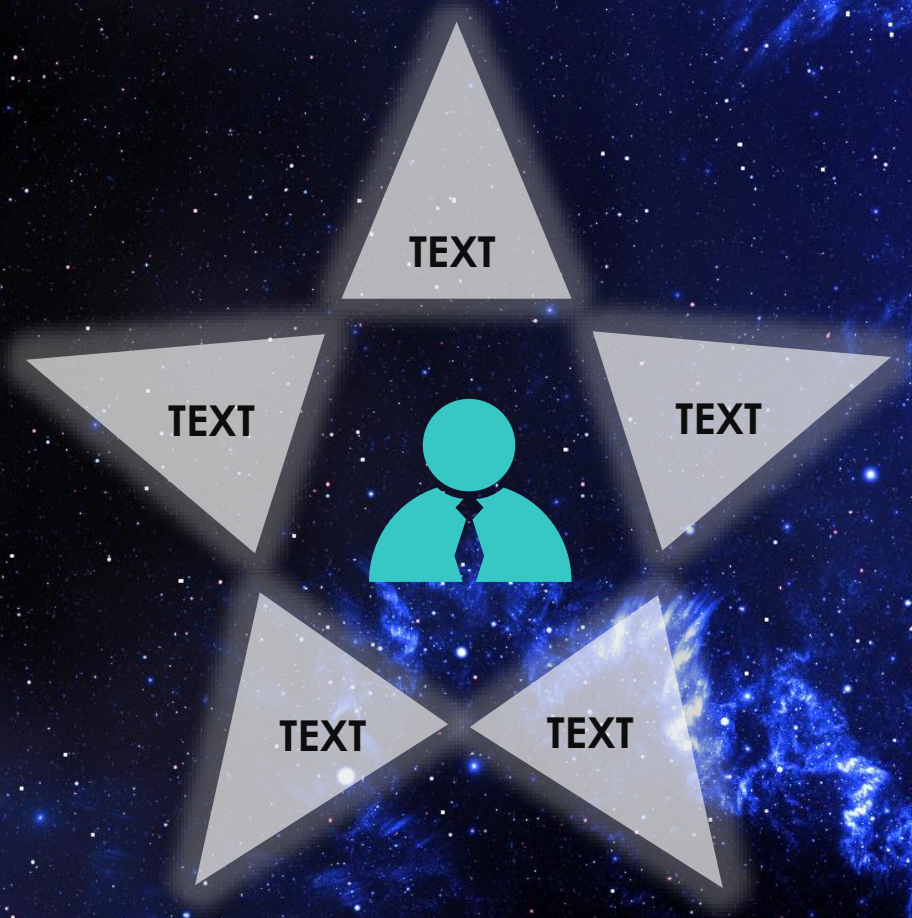

Xshell 7 (Free for Home/School)
文件(F) 编辑(E) 查看(V) 工具(T) 选项卡(B) 窗口(W)
连接
用户身份验证
登录提示符
登录脚本
SSH
安全性
隧道
SFTP
TELNET
连接
常规
名称(N):
新建会话 (2)
协议(P):
SSH
主机(H):
10.20.3.98
端口号(O):
22


连接
用户身份验证
登录提示符
登录脚本
SSH
安全性
隧道
SFTP
TELNET
RLOGIN
连接 > 用户身份验证
请选择身份验证方法和其它参数。
使用此部分以节省登录时间。但是，为了最大限度地提高安全性，如果担心安全问题，建议您将此部分留空。
用户名(U):
test
密码(P):


Commands:
bjobs #View where a job runs
bsub < bsub.scriptfile #Submit jobs
bkill jobid #Kill a job
bhosts #View job slot limits for hosts and host groups
bqueues #View job slot limits for queues
[test@mn01 ~]$

登录信息

用户更改密码



如何更改密码

yppasswd

```
[test@mn01 ~]$ yppasswd
Changing NIS account information for test on mn01.
Please enter old password:
Changing NIS password for test on mn01.
Please enter new password:
You cannot reuse the old password.
Please enter new password:
Please retype new password:

The NIS password has been changed on mn01.
```

文件传输

1

将scp / sftp指向登录节点

2

Winscp Xmanager xftp


C
LSF
作业调度系统

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作业调度系统

当前超算平台主要采用IBM公司的Platform LSF进行资源和作业管理

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

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

不要在登录节点不通过作业调度系统直接运行作业，以免影响其余用户的正常使用

提交作业准备

准备作业脚本

用户准备数据输入与作业脚本文件

上传数据文件

ftp上传数据文件至用户目录

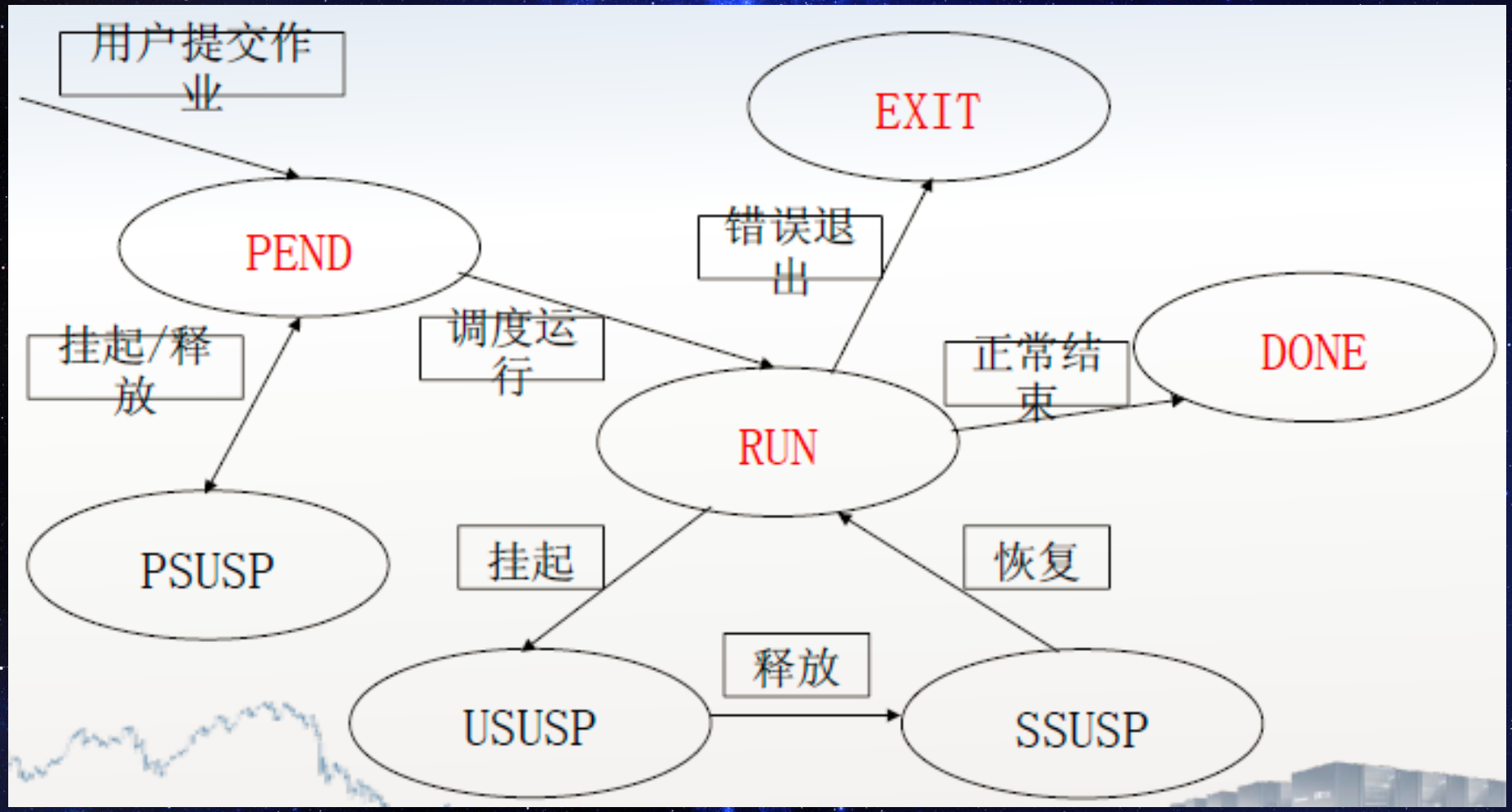
提交作业

bsub < myjob.lsf

作业完成

及时备份自己的作业输出文件

作业运行


用户提交作业
PEND
挂起/释放
PSUSP
调度运行
RUN
挂起
USUSP
释放
SSUSP
恢复
错误退出
EXIT
正常结束
DONE

查看队列情况 bqueues

利用*bqueues*可以查看现有队列信息，例如：*bqueues*

```
[test@mn01 ~]$ bqueues
QUEUE_NAME      PRIO STATUS          MAX JL/U JL/P JL/H NJOBS  PEND   RUN  SUSP
admin            50  Closed:Active     -    -    -    -     0     0     0     0
short            35  Open:Active       -    -    -    -     0     0     0     0
normal           30  Open:Active       -    -    -    -     0     0     0     0
```

QUEUE_NAME：队列名

PRIO：优先级，数字越大优先级越高

STATU状态：

- **Open：队列开放，可以接受提交新作业**
- **Active：队列已激活，队列中未开始运行的作业可以开始运行**
- **Closed：队列已关闭，不接受提交新作业**
- **Inact：队列未激活，可接受提交新作业，但队列中的等待运行的作业 不会开始运行**

查看队列情况 bqueues

```
QUEUE_NAME  PRIO  STATUS       MAX  JL/U  JL/P  JL/H  NJOBS  PEND  RUN  SUSP
normal       30   Open:Active   -     -     -     -    224     0   224     0
```

MAX：队列对应的最大作业槽数（Job Slot，一般与CPU核数一致，以下通称CPU核数），-表示无限

JL/U：单个用户同时可以使用的CPU核数 JL/P：每个处理器可以接受的CPU核数

JL/H：每个节点可以接受的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 RSV：为排队作业预留的CPU核数

查看队列情况 bqueues -l

队列也许会调整，利用*bqueues -l [队列名]*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 4 8
 PROCESSLIMIT
       8
```

QUEUE：队列名，跟着的下一行是描述 P

RIO：优先值，越大越优先

NICE：作业运行时的nice值，即作业运行时的操作系统调度优先值， 从-20到19，越小越优先

查看节点的运行情况 lsload

利用*lsload*命令可查看当前各节点的运行情况，例如：

lsload

```
[test@mn01 ~]$ lsload
HOST_NAME       status  r15s   r1m  r15m   ut    pg  ls    it   tmp   swp   mem
c002                ok   0.0   0.0   0.1   0%   0.0   0  4054  433G  3.9G  352G
c003                ok   0.0   0.0   0.1   0%   0.0   0  4054  433G  3.9G  352G
```

ut列表示利用率 status列：
ok：表示可以接受新作业，只有这种状态可以接受新作业
closed：表示系统在运行，但已被调度系统关闭，不接受新作业 locku：表示在进行排他性运行
unavail：作业调度系统服务有问题
r15s、r1m、r15m列：分别表示15秒、1分钟、15分钟利用率平均
tmp：/tmp目录大小
swp：swp虚拟内存大小
mem：内存大小

查看节点空闲情况 bhosts

利用*bhosts*命令可查看当前各节点的空闲情况，例如：

bhosts

```
[test@mn01 ~]$ bhosts
HOST_NAME          STATUS       JL/U    MAX  NJOBS    RUN  SSUSP  USUSP    RSV
c001               ok              -     48      0      0      0      0      0
c002               closed          -     48      0      0      0      0      0
```

STATUS：

ok：表示可以接收新作业，只有这种状态可以接受新作业

closed：表示已被作业占满，不接受新作业

unavail和unreach：系统停机或作业调度系统服务有问题

查看c001节点：*bhosts c001*

bhosts –l 会显示节点详细信息

查看用户信息 busers

利用*busers*可以查看用户信息，例如：

busers test

```
[test@mn01 ~]$ busers test
USER/GROUP          JL/P    MAX  NJOBS   PEND    RUN  SSUSP  USUSP    RSV
test                   -      -      0      0      0      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

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

请不要ssh到节点后直接运行作业，以免影响作业调度系统分配到此节点的作业

提交作业 busb

bsub		
说明	bsub用来提交job，常用的参数有：-n、-q、-o、-e、-J	
	-n	指定计算工作所需的核心数目。可省略，默认值是：1
	-q	指定执行计算工作的队列的名称。可省略
	-o	指定(stdout) 的输出目录名称。可省略，预设名称是：$job_id.out
	-e	指定(stderr) 的输出目录名称。可省略，预设名称是：$job_id.err
	-J	指定计算工作在队列中的名称。可省略，预设名称是所执行的程序名称

提交作业 bsub参数

Basic LSF Job Specifications			
Specification	Option	Example	Example-Purpose
Job Name	-J [SomeText]	-J MyJob1	Set the job name to "MyJob1"
Shell	-L [Shell]	-L /bin/bash	Uses the bash shell to initialize the job's execution environment.
Wall Clock Limit	-W [hh:mm]	-W 24:15	Set wall clock limit to 24 hour 15 min
Core count	-n ##	-n 480	Assigns 480 job cores.
Cores per node	-R "span[ptile=##]"	-R "span[ptile=48]"	Request 48 cores per node.
Memory Per Core	-M [##]	-M 2GB	Sets the per process memory limit to 2GB.
Memory Per Core	-R "rusage[mem=[##]]"	-R "rusage[mem=2GB]"	Schedules job on nodes that have at least 2GBs available per core.
Combined stdout and stderr	-o [OutputName].%j	-o stdout1.%j	Collect stdout/err in stdout.[JobID]

提交作业 bsub参数

Optional LSF Job Specifications			
Specification	Option	Example	Example-Purpose
Set Allocation	-P ######	-P projectA	Set allocation to charge to Project projectA
Specify Queue	-q [queue]	-q short	Request only nodes in short subset.
Exclusive Node Usage	-x		Assigns a whole node exclusively for the job.
Specific node type	-R "select[gpu|phi]"	-R "select[gpu]"	Requests a node with a GPU to be used for the job.

提交作业　　作业脚本

```
[test@mn01 hpl_test]$ vi mytest.lsf
#!/bin/bash
#
#BSUB -J MPIJob          ### set the job Name
#BSUB -q short           ### specify queue
#BSUB -n 40              ### ask for number of cores (default: 1)
#BSUB -m nodename
#BSUB –R “span[ptile=40]”        ### ask for 40 cores per node
#BSUB -W 10:00           ### set walltime limit: hh:mm
#BSUB -o stdout_%J.out           ### Specify the output and error file. %J is the job-id
#BSUB -e stderr_%J.err           ### -o and -e mean append, -oo and -eo mean overwrite

# here follow the commands you want to execute

# load the necessary modules
# NOTE: this is just an example, check with the available modules
module load intel/2018.4
module load mpi/intel/2018.4

### This uses the LSB_DJOB_NUMPROC to assign all the cores reserved
### This is a very basic syntax. For more complex examples, see the documentation
mpirun -np $LSB_DJOB_NUMPROC ./MPI_program
[test@mn01 hpl_test]$ bsub < mytest.lsf
Job <387> is submitted to queue <short>.
```

作业脚本范例　　vasp

```
#!/bin/sh
#BSUB -J N_F                                ##job name
#BSUB -q short                             ##queue name
#BSUB -n 80                                ##number of total cores
#BSUB -R "span[ptile=40]"                  ##40 cores per node
#BSUB -W 12:00                             ##walltime in hh:mm
#BSUB -R "select[hname!='c001']"           ##exclusive c001
#BSUB -e err.log                           ##error log
#BSUB -o H.log                             ##output log
module load intel/2018.4 mpi/intel/2018.4 vasp/5.4.4
#mpirun /share/home/xxx-xxx/vasp.5.4.4/vasp.5.4.4/bin/vasp_std &>log
mpirun vasp_std &>log
```

- 将log修改为$LSB_JOBID.log，生成每个作业对应的log文件

作业脚本范例 mpi

```
#!/bin/bash
#BSUB -J test
#BSUB -q short
#BSUB -n 48
#BSUB -e %J.err
#BSUB -o %J.out
#BSUB -R "span[ptile=48]"

module load fftw/2.1.5
module load intel/2018.4
module load mpi/intel/2018.4

cd $LS_SUBCWD
echo "processes will start at:"
date

mpirun -machinefile $LSB_DJOB_HOSTFILE -np 320  ./main > $LSB_JOBID.log 2>&1

echo "processes end at:"
date
```

- 编译mpicc必须与mpirun一致,这里是2018.4
- 不推荐使用自己编译mpi软件和fftw库
- main必须是可执行文件(chown a+x main)

查看作业情况 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  test  RUN      short  mn01          2*c011     *executab1  Mar  12  19:20
79727  test  PEND     normal  mn01         1*c022     *executab2  Mar  12  19:20
```

显示：

作业79726分别在node1和node2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 <test>, Project <default>, Status <PEND>,
Queue <normal> , Command <executab2>
Sun Mar 12 14:15:07: Submitted from host <mn01>,CWD
<$HOME>, Requested Resources <type==any && swp>35>;
PENDING REASONS:
The user has reached his/her job slot limit;
SCHEDULING PARAMETERS:
                    r15s  rlm  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。

bjobs –p jobid 也可以查看排队原因

作业监控 bjobs

使用bjobs可以监控到以下参数

PEND-作业在队列中等待调度与分派
RUN-作业已经辈分派到节点机上，正在运行
DONE-作业已经正常结束
EXIT-作业已经结束，但可能是异常退出或者手动被终止
UNKWN-执行节点失去联系，从而使作业状态不确定

查看任务实时输出
bpeek 输出所有内容
bpeek -f JOBID 类似于tail -f
另外也可以通过重定向的方式将标准输出和标准错误重定向到指定的文件

如果任务有异常退出，可以查看计算目录文件下的output.JOBID来查看具体的原因。

终止作业　　bkill

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

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

D

GPFS
并行文件系统

并行文件系统　GPFS

IBM通用并行文件系统（GPFS）是一个高性能的共享磁盘文件系统，提供从集群的所有节点到全局文件系统的快速，可靠的数据访问，GPFS允许并行应用程序同时访问一组文件（甚至是单个文件）来自任何安装了GPFS文件系统的节点，同时提供对所有文件系统操作的高级控制。此外，GPFS可以在单个I / O操作中读取或写入大块数据，从而最大限度地减少开销。

集群节点中的计算机中有可用的GPFS文件系统：

- /share/home：这是所有用户的主目录，当您登录时，默认情况下从主目录开始。每个用户都有自己的主目录来存储自己的个人数据。
- **/scratch**:这个空间是用来存储你的作业在执行过程中的用户数据和临时文件。

磁盘限额 GPFS-quota

配额是用户或组可用的存储量。 您可以将其描绘为随时可用的小磁盘。 默认值将应用于所有用户和组，并且不能超出。

您可以随时在每个文件系统中使用以下命令检查配额：

```
$ mmlsquota -g groupname --block-size auto
$ mmlsquota -u username --block-size auto
```

```
[test@mn01 ~]$ mmlsquota -u test --block-size auto
                     Block Limits                                    |     File Limits
Filesystem Fileset    type         blocks      quota      limit   in_doubt    grace |    files   quota    limit in_doubt    grace  Re
marks
share      root       USR          10.13G         1T         2T          0     none |    15944       0        0        0     none
[test@mn01 ~]$
```

E

常见问题

如何查询集群资源是否还有空闲?

bhosts 可以查询节点的状态，如果STATUS为ok，RUN为0，那么这个节点是没有作业在运行，为空闲状态

节点使用的核数?

注意：为了节点使用效率更高，单节点上使用核数需为24或者48，作业的总核数需为48的整数倍。在提交脚本里面应加上参数：

#BSUB -R "span[ptile=24]"或者#BSUB -R "span[ptile=48]"

作业跑到默认队列?
提交作业使用 qsub test.lsf或者 bsub test.lsf都是错误的。这两种方式，都会把作业提交到默认队列 hpc:q6248 q2080 ngs:normal hpc2:q6240
或者作业脚本#BUSB –q 没有指定队列名
正确的方式是bsub < test.lsf

作业异常退出怎么办?
如果您的作业在计算过程中异常退出，请在登录节点使用bjobs -l jobid (比如bjobs –l 1232343) 查看作业退出码

127 - 命令没有找到
128 - 命令调用没有执行，应用软件的License是否可用等

作业脚本里的命令未生效?

命令行输入可能存在中文符号

为何程序在其他平台能用，在这不能用?

（a）程序在我们计算平台上没有进行编译,直接运行会出错。跨平台后,程序需要重新编译才可以使用。

（b）提交的脚本，要根据计算平台使用手册，重新修改后提交。

如何查询各队列的作业时间限制?
bqueues -l 队列名或查看手册。RUNLIMIT 2880.0 min

作业报错出现quota或无法写入字样?
账号存储空间不足。

作业脚本中排除节点的命令是什么?

#BSUB -R "select[hname!='node2']" 排除node2

作业运行但没输出（空跑）？

原因可能是脚本、程序、算例、节点，需要根据具体情况来分析。

账号登陆出现-bash-4.2$?

账号环境变量文件出现问题,联系管理员。

THANK YOU

NGS、HPC、HPC2 manual

Yixian Huang

CUHK-shenzhen

How to install software?

Install software

Personal use

- Just install software in your directory

Public use in your lab

- Install software in /share/home/your group/apps directory, so all the users in /share/home/your group can use your software.
- Each group has a manager

How to run your scripts?

- The first step: write a script

For every line of resource specifications, #BSUB directive must be the first text of the line, and all specifications must come before any executable lines

```
#!/bin/bash
#
#BSUB -J jobname            ### set the job name
#BSUB -n number-of-slots    ### ask for number of cores
#BSUB -o %J.stdout          ### Specify the output file. %J is the job-id
#BSUB -e %J.stderr          ### Specify the error file. %J is the job-id
#BSUB -q normal or smp      ### specify queue，normal for 40-core host and smp for 128-core host

# here follow the commands you want to execute
# load the necessary modules
# NOTE: this is just an example, check with the available modules
module load mpi/intel/2018.4
module load app/version

mpirunapp.exe
```

How to run your scripts?

- The second step: submit it by typing the below command in terminal:

```
bsub < job.lsf
```

How to check your job status?

- you can check the status of your submission issuing the command:

  bjobs

  bjobs -l

  - The basic job states are:
    - **PEND** - the job is in the queue, waiting to be scheduled
    - **PSUSP** - the job was submitted, but was put in the suspended state (ineligible to run)
    - **RUN** - the job has been granted an allocation. If it's a batch job, the batch script has been run
    - **DONE** - the job has completed successfully
  - A pending job can remain pending for a number of reasons
    - **Dependency** - the pending job is waiting for another job to complete
    - **Priority** - the job is not high enough in the queue
    - **Resources** - the job is high in the queue, but there are not enough resources to satisfy the job's request

Other useful commands in LSF

- bhosts: display hosts and their static and dynamic resources
- bkill: remove a job from the queue or kill it if it is running
- bjobs -u all: display all jobs in the scheduling queues, one job per line
- lshosts: display hosts and their static resource information

Module usage

What is module?

- The module utility helps you identify software that is available on the system and then load packages that are compatible.

- The module utility manages complex combinations of paths, variables, and dependencies so you can compile and run jobs efficiently and make the most of your allocation.

Essential module commands

- module av – Show which modules are available for use with the currently loaded compiler.
- module load – Load the default version of a software package, or load a specified version.
- module unload – Unload the specified software package.

Conda configuration and use in System

What is conda?

- Conda is an open source package management and environment management system for installing and easily switching between multiple versions of packages and their dependencies.

How to configure conda?

The first step: configure .bashrc to initialize conda

- Open .bashrc and add below statements add the end.

```
# >>> conda initialize >>>
# !! Contents within this block are managed by 'conda init' !!
__conda_setup="$('/share/apps/anaconda3/XXX/bin/conda' 'shell.bash' 'hook' 2> /dev/null)"
if [ $? -eq 0 ]; then
    eval "$__conda_setup"
else
    if [ -f "/share/apps/anaconda3/XXX/etc/profile.d/conda.sh" ]; then
        . "/share/apps/anaconda3/XXX/etc/profile.d/conda.sh"
    else
        export PATH="/share/apps/anaconda3/XXX/bin:$PATH"
    fi
fi
unset __conda_setup
# <<< conda initialize <<<
```

How to configure conda?

The second step: create .condarc to add channels

- Create and write below statements.

```
channels:
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/bioconda/
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge/
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main/
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
  - conda-forge
  - bioconda
  - defaults

auto_activate_base: false
```

How to use conda?

- Create conda environment

conda create -n python2 python=2

#-n: set the name for the new environment

#python=2: specifies the version of python for the new environment

- Activate conda environment

conda info –e

show available environments

conda activate python2

#python2: environment you want to activate

- Search package

conda search package

#search whether the package exits in conda

How to use conda?

- Install packages in this environment

conda install package=1.3

#1.3: package version

- Deactivate the environment after using packages

conda deactivate

- Remove the environment if you don’t use it any more

conda remove -n python2 --all

remove environment

使用注意事項

- 系统类
  - 禁止在登陆节点上执行程序
  - 禁止挖矿
  - 资料请自行备份。离开研究院，账号与目录将删除
- 资源类
  - 各设备空间有限，程序跑完后中间挡案，请移除；测试文件亦是如此
  - 若软件为课题组大多成员都会使用的，请课题组管理员统一安装在apps，节省资源
    - grp-baichen: baichen; grp-hiraoh: hiraoh; grp-lizy: wangzhuo; grp-zhulz: wenping; grp-hwangjk: jones; grp-chenggj: yangwei; grp-huangxd: huanghsiyuan; grp-wangyf: yfwang ; grp-sunhao: sunhao
- GPU类
  - GPU4张卡或8张卡共享32核，单卡最好不要占用超过8核或4核