

# 超算使用基础及技巧概述



国家超级计算天津中心



(022)-65375580



<http://www.nsc-ty.cn>

1. 超算发展历程
2. Linux系统简介
3. 应用环境配置简介
4. Slurm作业管理系统简介
5. 超算使用技巧简介
6. 总结及建议

# 超算发展历程

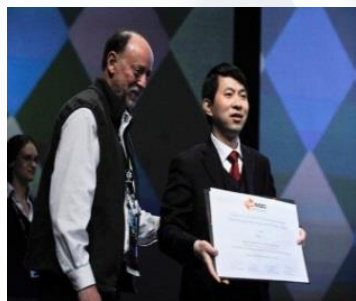
艰难起步-成果频出



- 1978，邓小平同志批示，国防科技大学开始研制“银河-1”
- 从此，开始了中国研制超级计算机的艰难和辉煌的历程

## ■ 世界超级计算机TOP500排名

从2010年开始，天河一号、天河二号和神威太湖之光相继累计获得11次世界超级计算机TOP500排名第一。中国超算人坚持自主创新，奋勇攻关，使中国超级计算机登上世界舞台，成为世界超级计算发展的重要力量。



2010年11月首次夺冠



2013年6月-2015年11月连续6次夺冠



2016年6月-2017年12月连续4次夺冠



## ■ 世界超级计算机应用最高奖-“戈登·贝尔”奖

2016年，获奖应用名称“千万核可扩展全球大气动力学全隐式模拟”

2017年，获奖应用名称“非线性地震模拟”

2021年，获奖应用名称“超大规模量子随机电路实时模拟”，3项入围

2021年，新冠特别奖入围应用名称“基于自由能微扰-绝对结合自由能方法的大规模新冠药物虚拟筛选”





# 超算发展历程

## 国家超级计算天津中心

- 于2009年5月，经国家科技部批准，由天津滨海新区和国防科大共同建设成立的**首批国家超级计算中心**；
- 部署我国首台千万亿次超级计算机“天河一号”，于**2010年荣获我国在超算领域的首个世界第一**；
- 构建超算中心、云计算中心、电子政务中心和大数据应用环境，提供超级计算，云计算，大数据服务，是**面向全国和全世界开放的公共科技服务平台**；
- 创新运营机制：不断进行体制机制创新，成为我国乃至世界上**首个实现“收支平衡，略有结余”的超算中心**；



### 国家创新支撑平台



发改委大数据  
实验室

工信部工业云  
试点单位

科技部示范型  
国际合作基地

人社部博士后  
科研工作站

联合实验室



# 超算发展历程

## 国家超级计算天津中心资源详情



资源	节点配置	系统介绍	系统特点
HPC1-4系统	Intel E5-2690 v4 @ 2.6GHz 28cores, 128GB Intel E5-2697A v4@ 2.6GHz 32cores, 128GB Intel Gold-6240 @ 2.60GHz, 36cores, 128GB/256GB	自主构建的新系统，每套系统512个计算节点	单核性能高，x86结构
3F系统	国产自主飞腾芯片, 64cores, 128GB	5000个计算节点	国产自主，性价比高
天河一系统	Intel Xeon X5670 2.93GHz , 12 cores, 24GB	采用CPU和GPU相结合的异构融合计算体系结构	国内首台TOP500
GPU系统	CPU: Intel CPU Gold 6354 @ 3.0GHz, 36cores,256GB GPU: 2*A100, 单卡显存80GB  CPU: Intel E5-2690 v4 @ 2.6GHz, 28cores, 128GB GPU: 2*K80, 单卡显存40GB	可支持仿真模拟和人工智能研究	GPU计算



## Operating system Family share for 06/2008

Operating system Family	Count	Share %	Rmax Sum (GF)	Rpeak Sum (GF)	Processor Sum
Linux	427	85.40 %	8465100	13333004	1408054
Windows	5	1.00 %	159264	211320	25472
Unix	25	5.00 %	619912	874185	73174
BSD Based	1	0.20 %	35860	40960	5120
Mixed	40	8.00 %	2391451	2941095	897127
Mac OS	2	0.40 %	28430	44816	5272
<b>Totals</b>	<b>500</b>	<b>100%</b>	<b>11700016.97</b>	<b>17445379.70</b>	<b>2414219</b>

世界超级计算机TOP500操作系统统计

命令	说明	命令	说明
cat	查看文件内容	rm	删除文件
more		pwd	显示当前所在目录
less		mkdir	创建目录
ls	显示目录内容	chmod	改变文件/目录权限
cd	切换工作目录	tar	文件打包/解包
cp	拷贝文件	ln	创建文件链接
mv	移动文件	which	显示命令所在目录
passwd	修改账号密码		

- ❑ **cd**: 改变工作目录
  - 用法: **cd** 目录
  - 进入指定的目录, 即将工作目录改为指定的目录
- ❑ **pwd**: 显示当前目录
  - 用法: **pwd**
  - 告诉用户当前的工作目录, 即当前在哪个目录下
- ❑ **ls**: 显示文件或目录信息
  - 用法: **ls** [选项] [目录或是文件]
  - 常用选项:
    - a 列出目录下的所有文件, 包括以.开头的隐含文件;
    - l 列出文件的详细信息, 可以用ll表示;
    - t 以时间排序;
- ❑ **mkdir**: 创建目录
  - 用法: **mkdir** 目录名
  - 递归创建: **mkdir -p** /path/to/dir

```
[liqf@th-hpc2-ln0 linux]$ cd messydir/
[liqf@th-hpc2-ln0 messydir]$ pwd
/THL7/home/liqf/hpc-course/linux/messydir
[liqf@th-hpc2-ln0 messydir]$ ls
boundary_decay.F90      PushParticle.F90
bparasolver.F90         restart.F90
ChargeParticle.F90      SDP_interface.F90
course                  setup.F90
emptydir1               shift.F90
emptydir2               slurm-2131625.out
emptydir3               slurm-2131655.out
eqdata.F90              slurm-2131657.out
eqplot.F90              smooth.F90
fft_gl.F90              snapshot.F90
field.F90               submit.sh
function.F90            taehdf5.F90
gromacs.sh              tcl8.5.13-src.tar.gz
hello                   temp
loaderZ.F90            test
load.F90                test2
LR_interface.F90        tmp
M3DC1_interface.F90     tracking.F90
ohm_law.F90             utility.F90
[liqf@th-hpc2-ln0 messydir]$ mkdir test123
[liqf@th-hpc2-ln0 messydir]$ mkdir -p test-mkdir/test
```



- ❑ **mv**: 移动或更名现有的文件或目录
  - 用法: **mv** [选项] 源文件或目录 目标文件或目录
- ❑ **cp**: 文件复制
  - 用法: **cp** [选项] 源文件或目录 目标文件或目录
  - 常用选项: **-r** 复制目录
- ❑ **rm**: 删除文件或目录
  - 用法: **rm** [选项] 文件或目录名
  - 常用选项:
    - f** 强制删除, 并且不给提示。
    - r** 删除整个目录, 包括文件和子目录
- ❑ **ln**: 创建链接(link)
  - 用法: **ln** [选项] 目标[链接名]
  - 常用选项: **-s** 建立符号链接(软链接)。

```
[liqf@th-hpc2-ln0 test123]$ mv ../slurm-213
slurm-2131625.out slurm-2131657.out
slurm-2131655.out slurm-2132657.out
[liqf@th-hpc2-ln0 test123]$ mv ../slurm-2131655.out .
[liqf@th-hpc2-ln0 test123]$ ls
slurm-2131655.out
[liqf@th-hpc2-ln0 test123]$ cp slurm-2131655.out test
[liqf@th-hpc2-ln0 test123]$ ls
slurm-2131655.out test
[liqf@th-hpc2-ln0 test123]$ rm test
[liqf@th-hpc2-ln0 test123]$ ls
slurm-2131655.out
[liqf@th-hpc2-ln0 test123]$ cp -r ../tmp/ .
[liqf@th-hpc2-ln0 test123]$ ls
slurm-2131655.out tmp
[liqf@th-hpc2-ln0 test123]$ rm -r tmp/
[liqf@th-hpc2-ln0 test123]$ ls
slurm-2131655.out
[liqf@th-hpc2-ln0 test123]$ ln slurm-2131655.out test
[liqf@th-hpc2-ln0 test123]$ ln -s slurm-2131655.out test1
[liqf@th-hpc2-ln0 test123]$ ls -l
total 8
-rw-rw-r-- 2 liqf liqf 536 May 10 16:17 slurm-2131655.out
-rw-rw-r-- 2 liqf liqf 536 May 10 16:17 test
lrwxrwxrwx 1 liqf liqf 17 May 10 16:20 test1 -> slurm-2131
655.out
```

### ❑ tar: 文件打包和解包

- 用法: **tar** [主选项+辅选项] 文件或者目录
- 实用用法:

解压: **tar -xvf data1.tgz**

压缩: **tar -cvf data1.tgz data1**

### ❑ which: 查找指定命令所在的位置

- 用法: **which** command

### ❑ passwd: 更改口令

- 直接在命令行中输入**passwd**, 然后根据提示, 先输入原来的口令, 回车后, 再输入新口令, 并确认一次。
- 提醒: 输入密码时, 屏幕上不会有任何输出或提示!

```
[liqf@th-hpc2-ln0 test123]$ ls
slurm-2131655.out  test  test1
[liqf@th-hpc2-ln0 test123]$ tar cvf data1.tgz slurm-2131655.out
[liqf@th-hpc2-ln0 test123]$ ls
data1.tgz  slurm-2131655.out  test  test1
[liqf@th-hpc2-ln0 test123]$ rm slurm-2131655.out
[liqf@th-hpc2-ln0 test123]$ tar xvf data1.tgz
slurm-2131655.out
[liqf@th-hpc2-ln0 test123]$ ls
data1.tgz  slurm-2131655.out  test  test1
[liqf@th-hpc2-ln0 test123]$ which tar
/usr/bin/tar
[liqf@th-hpc2-ln0 test123]$ which gcc
/usr/bin/gcc
```

```
[liqf@th-hpc2-ln0 test123]$ passwd
Changing password for user liqf.
(current) LDAP Password:
New password:
Retype new password:
```

- ❑ **cat**: 查看文件内容, 打印到屏幕
- ❑ **head**: 查看文件的开头部分的内容
  - 常用选项: -n 指定打印行数
- ❑ **tail**: 查看文件末尾部分的内容
  - 常用选项:
    - n 指定打印行数
    - f 持续输出

```
[liqf@th-hpc2-ln0 tmp]$ cat gromacs.sh
#!/bin/sh
gmx_mpi_grompp -f rf_verlet.mdp
yhrun -N2 -n56 gmx_mpi mdrun -ntomp 1

[liqf@th-hpc2-ln0 tmp]$ head gromacs.sh
#!/bin/sh
gmx_mpi_grompp -f rf_verlet.mdp
yhrun -N2 -n56 gmx_mpi mdrun -ntomp 1

[liqf@th-hpc2-ln0 tmp]$ head -n1 gromacs.sh
#!/bin/sh
[liqf@th-hpc2-ln0 tmp]$ tail gromacs.sh
#!/bin/sh
gmx_mpi_grompp -f rf_verlet.mdp
yhrun -N2 -n56 gmx_mpi mdrun -ntomp 1

[liqf@th-hpc2-ln0 tmp]$ tail -n1 gromacs.sh
yhrun -N2 -n56 gmx_mpi mdrun -ntomp 1

[liqf@th-hpc2-ln0 tmp]$ tail -n2 gromacs.sh
yhrun -N2 -n56 gmx_mpi mdrun -ntomp 1

[liqf@th-hpc2-ln0 tmp]$ tail -f gromacs.sh
#!/bin/sh
gmx_mpi_grompp -f rf_verlet.mdp
yhrun -N2 -n56 gmx_mpi mdrun -ntomp 1
```

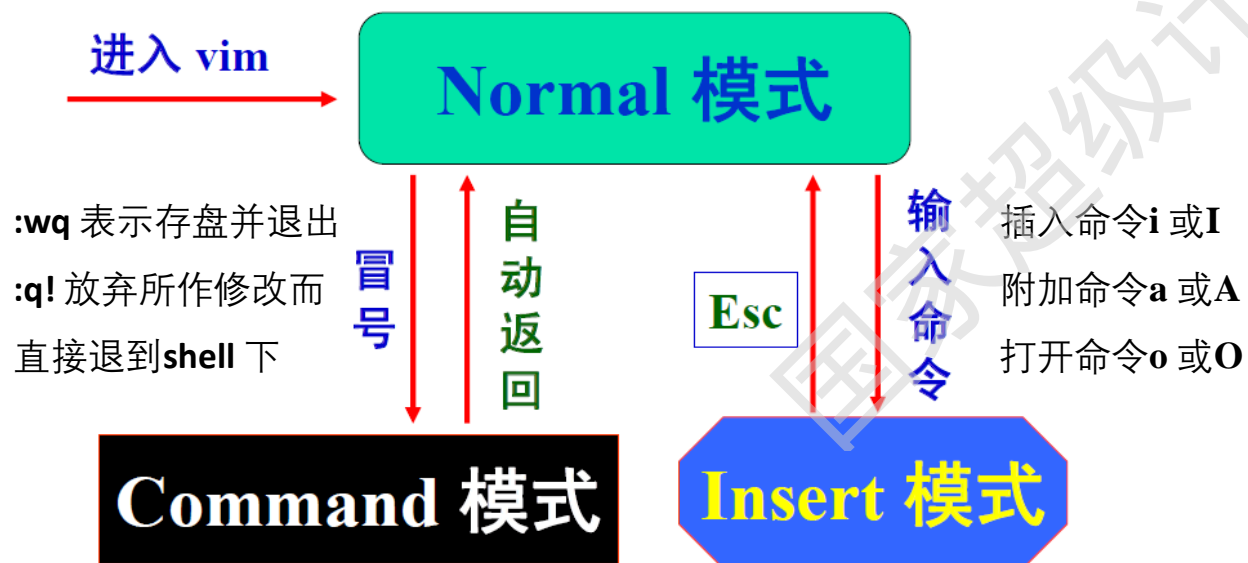


### □ Vim的工作模式

- **Normal** 模式(正常模式、编辑模式)
- **Insert** 模式 (插入模式)
- **Command** 模式 (命令模式、末行模式)

### □ Normal 模式下的常用操作

nG	光标移到第 <b>n</b> 行, 若不指定 <b>n</b> , 则移到最后一行
gg	光标移到第一行
yy	复制行
p	粘贴行
u	撤销操作
Ctrl+r	重复操作
/word	光标移动到向下第一个搜索到的word处



- 环境变量：

环境变量相当于给系统或应用程序设置的一些环境参数，当要求系统运行一个程序而没有告诉它程序所在的完整路径时，系统会到环境变量中指定的路径去找。超算系统中常用的环境变量如下：

- |                   |       |
|-------------------|-------|
| • PATH            | 可执行文件 |
| • LD_LIBRARY_PATH | 库文件   |
| • C_LIBRARY_PATH  | 头文件   |

加载方式：

```
export PATH=/path/to/bin:$PATH
```

### ● Module常用命令

命令	作用
module av	查看系统中可用的软件
module help	查看具体软件的信息
module add   load	加载环境变量
module rm   unload	卸载环境变量
module list	显示用户已加载的环境变量
module show	显示具体的 Module 配置文件
module swap   switch	替换环境变量
module purge	卸载当前 shell 环境下的所有环境变量



# 应用环境配置简介

## Module环境部署工具



### module av

```
[liqf@th-hpc2-ln0 ~]$ module av

----- /THL7/software/modules/3.2.10/Modules/3.2.10/THL7/Applications -----
abinit/8.2.2-icc16-mvapich2(default)      espresso/5.4.0-icc16-openmpi2             hifiles/1.0.0-icc16-IMPI5.1(default)      phonopy/1.11.10.4-py27-gcc4.8
activeTcl/8.5.18(default)                 espresso/6.1-icc16-IMPI5.1                hifiles/1.0.0-icc16-mvapich2              phonopy/1.11.2-py27-gcc4.8(default)
autodock/4.2.6(default)                   espresso/6.1-icc16-mvapich2              hmmer/3.1b2-icc16-mvapich2(default)        plink2/20170320(default)
autodock_vina/1.1.2(default)              espresso/6.1-icc16-openmpi2              impute2/2.3.2-static(default)            samtools/1.2-gcc4.8(default)
bcftools/1.2-gcc4.8(default)              espresso/6.1-icc17-IMPI2017.4            lammps/11Aug17-icc16-IMPI5.1              scalasca/2.3.1-icc16-IMPI5.1(default)
beagle/4.0(default)                       espresso/6.2.1-icc16-IMPI5.1             lammps/11May18-icc16-IMPI5.1              scorep/3.0-icc16-IMPI5.1(default)
bedtools2/2.25.0-gcc4.8                   espresso/6.2.1-icc16-mvapich2            lammps/17Nov16-icc16-IMPI5.1(default)      siesta/4.0-icc16-IMPI5.1(default)
bedtools2/2.26.0-gcc4.8(default)          espresso/6.2.1-icc16-openmpi2            lammps/22Jul16-default-icc16-IMPI5.1      siesta/4.0-icc16-mvapich2
bedtools2/4.2.6                           espresso/6.2.1-icc17-IMPI2017.4          lammps/22Jul16-default-icc16-mvapich2     siesta/4.0-icc17-IMPI2017.4
berkeleyGW/1.0.6-real-icc16-mvapich2     espresso/6.3-icc17-IMPI2017.4            lastz/1.04.00-gcc4.8(default)             tophat/2.1.1(default)
blast/2.6.0(default)                      fastqc/0.11.5                           namd/2.12-icc16-IMPI5.1(default)           varScan/2.3.9(default)
blat/36(default)                          ffmpeg/3.2.4-gcc4.8(default)             namd/2.12-icc16-mvapich2                  vcftools/0.1.14-icc16(default)
boltztrap/1.2.5-icc16(default)            fishingCNV/1.5.3(default)                ncview/2.1.7-icc16-mvapich2(default)       velvet/1.2.10-gcc48(default)
bowtie/1.2.0-gcc4.8(default)              gromacs/4.6.7-double-icc16-IMPI5.1       nekter++/4.4.1-icc16-mvapich2(default)    vmd/1.9.3
bowtie2/2.3.1-gcc4.8(default)             gromacs/4.6.7-single-icc16-IMPI5.1       nview/6.6-icc16-mvapich2(default)         vmd/1.9.3-Text-mode(default)
breakdancer/1.4.5-gcc4.8                 gromacs/5.1.2-double-icc16-IMPI5.1       openmx/2.2.0-icc16(default)              vtstscripts/935(default)
bwa/0.7.15-gcc4.8(default)               gromacs/5.1.2-double-icc16-IMPI5.1-AVX256 openfoam/3.0.1-gcc4.8-mvapich2            wannier90/1.2-icc16(default)
cnvnator/0.3-gcc4.8(default)              gromacs/5.1.2-double-icc16-mvapich2      openfoam/3.0.1-icc16-mvapich2(default)     wannier90/2.0.0-icc16-IMPI5.1
control-freec/10.5(default)              gromacs/5.1.2-single-icc16-IMPI5.1(default) openmx/3.8-icc17-IMPI2017.4(default)      wannier90/2.1.0-icc16-IMPI5.1
cp2k/4.1-icc16-IMPI5.1(default)           gromacs/5.1.2-single-icc16-IMPI5.1-AVX256 packmol/17.076-gcc4.8                     wps/3.8-icc16-mvapich2(default)
espresso/5.4.0-icc16-IMPI5.1(default)     gromacs/5.1.2-single-icc16-mvapich2      packmol/17.076-icc16(default)             wrf/3.8-emreal-dmpar-icc16-mvapich2(default)
espresso/5.4.0-icc16-mvapich2             harminv/1.4-icc16(default)               phono3py/1.10.9-py27-gcc4.8              yambo/4.1.4-icc16-IMPI5.1(default)

----- /THL7/software/modules/3.2.10/Modules/3.2.10/THL7/Compilers -----
GCC/4.8.5                                java/jdk1.8.0_121(default)               MPI/mpich-3.1/intel2016u3                 PGI/16.10(default)
Intel_compiler/16.0.3(default)           MPI/Intel/IMPI/5.1.3.210                 MPI/mvapich2-2.2/gcc4.8.5                MPI/openmpi-2.0.2/gcc4.8.5
Intel_compiler/16.0.4                   MPI/mpich-3.1/gcc4.8.5                   MPI/mvapich2-2.2/intel2016u3             MPI/openmpi-2.0.2/intel2016u3
                                           yasm/1.3.0-icc16(default)
```

### module add / module list

```
[liqf@th-hpc2-ln0 ~]$ module add Intel_compiler
[liqf@th-hpc2-ln0 ~]$ module add MPI/Intel/IMPI/5.1.3.210
[liqf@th-hpc2-ln0 ~]$ module list
Currently Loaded Modulefiles:
  1) Intel_compiler/16.0.3          2) MPI/Intel/IMPI/5.1.3.210
[liqf@th-hpc2-ln0 ~]$
```

### 查看节点状态 --- yhi

用户可以使用yhi命令查看节点的使用情况  
可以查看自己可用的分区



```
[liqf@th-hpc2-ln0 ~]$ yhi
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug2      up       30:00      4   idle cn[552-555]
TH_LONG2    up    10-00:00:0      1 down* cn924
TH_LONG2    up    10-00:00:0    401 alloc cn[572-623,625-645,647-655,658-6
60,963,965-971,973-997,1004-1034,1036-1063]
TH_LONG2    up    10-00:00:0     25  idle cn[624,646,656-657,669-670,673-6
```

PARTITION 表示分区， TIMELIMIT 表示该分区的时间限制，  
NODES 表示节点数， STATE 表示节点运行状态。

### 作业提交

目前天河系统部署的资源管理系统包括多种作业提交方式，包括交互模式yhrun和批处理模式yhbatch

yhrun

交互模式

用户的作业命令在提交节点上运行，关闭当前登陆终端会导致程序退出

yhbatch

批处理模式

用户的作业脚本在分配的节点上运行，关闭当前登陆终端不会影响作业运行

➤ 交互式作业提交 --- yhrun

```
yhrun -N 20 -n 240 -p debug2 ./a.out
```

➤ 批处理式作业提交 --- yhbatch

1.编写脚本sub.sh

```
#!/bin/bash
```

```
yhrun -N 20 -n 240 -p debug2 ./a.out
```

2.提交脚本

```
yhbatch -N 20 -n 240 -p debug2 ./sub.sh
```



### 查看作业状态 --- yhq

用户可以使用yhq命令查看作业状态（仅限自己账户）

```
[zhenggang@ln1%tianhe ~]$ yhq
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)
6167633 debug test.sh zhenggang PD 0:00 1 (Priority)
6167645 TH_SR tmp_th zhenggang R 2:30:48 8 cn[4901-4908]
6167836 TH_SR tmp_th zhenggang R 2-12:30:39 20 cn[4457,4469-4488]
```

关键词	含义
JOBID	任务 ID
PARTITION	用户作业正在使用的计算分区
NAME	任务名称
USER	用户名
ST	作业状态：R: 运行状态, PD: 排队等待状态, S: 挂起状态, CG: 作业正在退出状态
NODES	结点个数
NODELIST	结点名字列表
REASON	Priority: 优先级 Association Resource Limit: 关联的资源限制已满 Resources: 当前可用资源不能满足作业需求 Partition Down: 作业所在的分区处于 down 状态

### ● 作业取消 --- yhcancel

取消已提交的作业

通过yhq命令查看作业队列，获取作业id号

取消作业命令：yhcancel 6167937

取消某用户全部作业命令：

yhcancel -u user

# 超算使用技巧

## 天河超级计算机组织结构

### 1. 登录系统

- 链接网络
- 登录 VPN 建立虚拟专用网络 (EasyConnect)
- 登录账号链接系统 (shell客户端)

### 2. 编译可执行文件

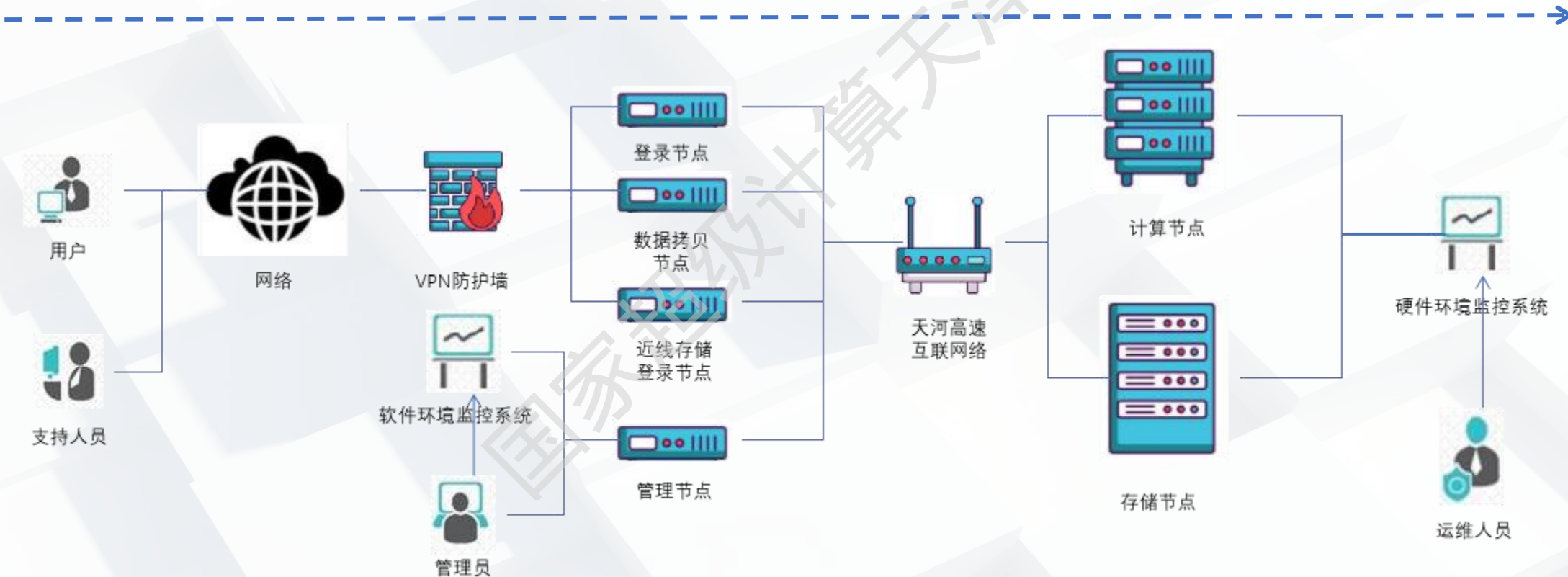
- 上传程序和数据 (FTP客户端)
- 加载编译环境
- 编写编译脚本
- 编译调试

### 3. 提交运行

- 编写运行脚本
- 提交运行
- 监控运行状态

### 4. 确认结果

- 确定输出文件和结果
- 下载或处理结果数据



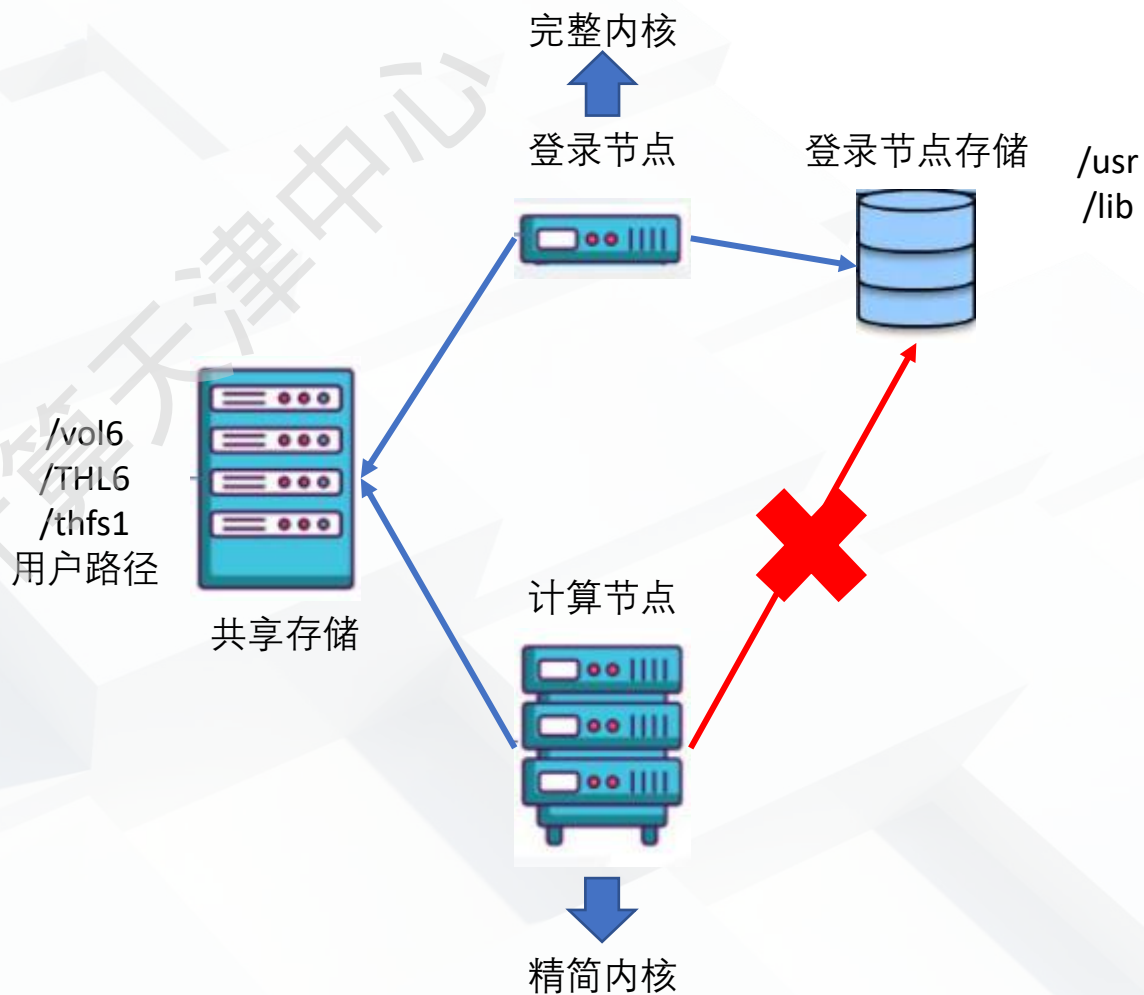
“天河”超级计算机系统网络框架图



# 超算使用技巧

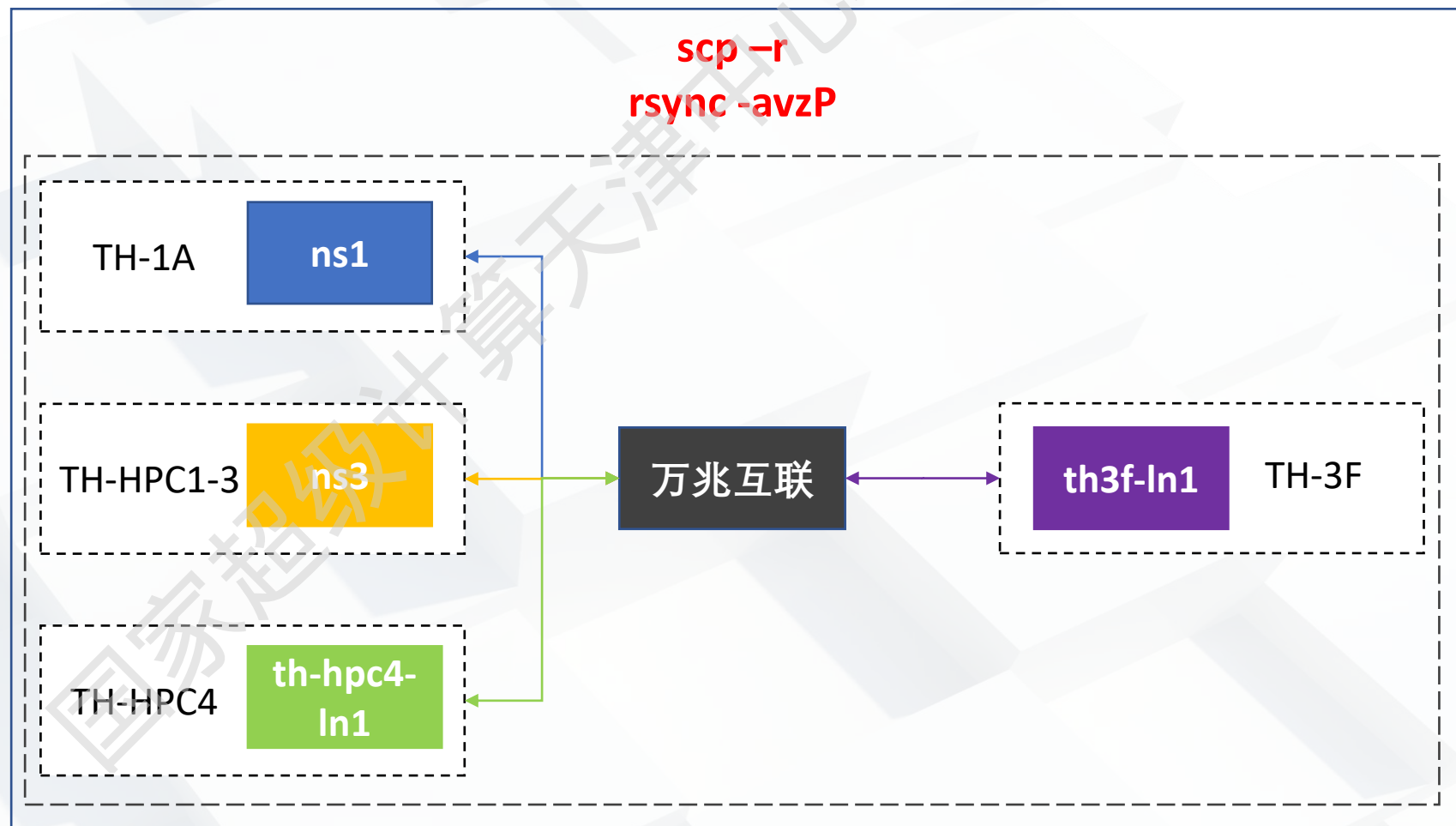
## 登录节点与计算节点

- 登录节点和计算节点均可访问共享存储
- 登录节点存储无法被计算节点访问（包括：系统内核库，apt安装的库等，计算节点精简内核缺少系统库），可通过添加loginnode的module实现
- 可通过which和locate命令查看可执行文件和库的所在位置（如which gcc; locate libm.so）
- 可通过ldd命令查看可执行文件依赖库的情况，如果依赖库中有/usr/lib等登录节点存储中的库，建议添加loginnode



bin	dev	home	lib64	media	opt	root	sbin	sys	THL6	THL8	usr
boot	etc	lib	login_info_exporter.log	mnt	proc	run	srv	THL5	THL7	tmp	var

1. 登录系统的对应登录节点 (TH-1A:LN3/TH-HPC:任意/TH-HPC4:LN1/TH-3F:LN1)
2. 输入传输命令 (例:  
`scp -r user@ns1:/path/to/th1a .)`
3. 输入远端服务器帐号的密码
  - TH1A系统需要先ssh到ns1节点再最后续处理



\*\*出于安全角度考虑，超算系统不能直接连接互联网。但为了满足用户的应用需求，推出了一个折中方案，即使用代理服务加白名单的形式提供互联网数据下载。

代理服务使用模式：

1. 使用module命令加载代理模块 :`module add proxy`
2. 执行下载任务，如使用 `wget` 等下载命令，或者在配置python`环境时直接使用 `pip` 等命令。

目前部分已添加的网站包括：

- Python类：python 官网、pip源、常用conda源、国内代理源
- 气候气象类： `svn-ccsm-inputdata.cgd.ucar.edu` 、 `rda.ucar.edu` 、 `ftp.ncep.noaa.gov` 、 `svn.cgd.ucar.edu`
- Github类： `github` 和 `raw.githubusercontent.com` 仓库等



### □ 命令行自动补齐功能

- 通常用户在**bash**下输入命令时不必把命令输全，**shell**就能判断出你所要输入的命令。
- 该功能的**核心思想**是：**bash**根据用户已输入的信息来查找以这些信息开头的命令，从而试图完成当前命令的输入工作。用来执行这项功能的键是**Tab** 键，
- 按下一次**Tab** 键后，**bash** 就试图完成整个命令的输入
- 按下两次**Tab** 键后，**bash** 将列出所有能够与当前输入字符相匹配的命令列表。

### □ 管道

- UNIX 系统的一个基本哲学是：一连串的小命令能够解决大问题。其中每个小命令都能够很好地完成一项单一的工作。现在需要有一些东西能够将这些简单的命令连接起来，这样管道就应运而生。
- 管道“|”的基本含义是：将前一个命令的输出作为后一个命令的输入
- **ls /home | grep liqf**
- 利用管道可以实现一些很强的功能。

### □ 管道举例

一个较复杂的例子：输出系统中用户名的一个排序列表。这里需要用到三个命令：**cat**、**awk**、**sort**，其中**cat**用来显示文件**/etc/passwd** 的内容，**awk** 用来提取用户名，**sort** 用来排序。

- **cat /etc/passwd | awk -F: '{print \$1}' | sort**

### □ 通配符

**bash** 提供许多功能用来帮助用户节省输入命令的时间，其中最常用的一种方法就是使用通配符，可以用来在引用文件名时简化命令的书写。

□ 在**bash** 中可以使用三种通配符：**\***、**?**、**[]**。

<b>*</b>	匹配任意长度的字符串（包括零个字符）
<b>?</b>	匹配任何单个字符
<b>[]</b>	创建一个字符表列，方括号中的字符用来匹配或不匹配单个字符。 如： <b>[xyz]</b> 匹配 <b>x</b> 、 <b>y</b> 或 <b>z</b> ，但不能匹配 <b>xx</b> ， <b>xy</b> 或者其它任意组合。 无论列表中有多少个字符，它只匹配一个字符。 <b>[abcde]</b> 可以简写为 <b>[a-e]</b> 。 另外，用感叹号作为列表的第一个字符可以起到反意作用，如： <b>[!xyz]</b> 表示匹配 <b>x</b> 、 <b>y</b> 、 <b>z</b> 以外的任意一个字符。

- 通配符“**\***”的常用方法就是查找具有相同扩展名的文件

**ls \*.tar.gz**                      **rm \*.old**

- 问号通配符“**?**”必须匹配一个且只能匹配一个字符，通常用来查找比**\*** 更为精确的匹配。

**ls \*.???**

- 方括号通配符使用括号内的字符作为被匹配的字符，且只能匹配其中的一个字符。如列出以**a**、**b**、**c** 开头，且以**.dat** 为扩展名的所有文件：

**ls [abc]\*.dat**

- 可以在方括号中使用连字符- 来指定一个范围，如列出以字母开头，数字结尾的所有文件：

**ls [a-zA-Z]\*[0-9]**



### □ nohup 命令

用户退出系统后能继续运行的进程通常当用户退出系统后，所有属于该用户的进程将全部被终止。如果希望程序在退出系统后仍然能够继续运行，需使用**nohup** 命令后台启动该进程。

- 用法： **nohup** 命令[选项] [参数] &

### □ 查看正在运行的程序： **ps**

- **ps** [选项]
- 常用： **ps -ef**

### □ 终止前台进程使用： **Ctrl+c**

### □ 终止后台进程： **kill**

- 正常结束: **kill pid**
- 强制结束: **kill -9 pid**

### ❑ 超算上python的使用模式

- 用户个人路径下安装python

例： /thfs1/software/mirror/anaconda

下有很多python安装脚本， ./执行即可

- 使用共享路径下的python

例： module add python/3.7\_anaconda

### ❑ 超算上python包管理

1. 加载python环境（用户路径安装 /module/conda）： module add

python/3.7\_anaconda

2. 添加代理： module add proxy

3. 安装python包（pip/conda）： pip install  
numpy -U

\* 需要注意， -U参数的意义是将包装到用户路径，如使用用户路径下的python，该参数可以不添加

### “青索”客户端

#### ◆ 开发初衷

解决超算使用软件栈的散落，**提供免费快速安全的超算使用方式。**

#### ◆ 实现功能

**实现了SSH、SFTP、VPN三大件的集成**，目前具备一键使用配置、VPN常驻、数据断点续传、机器智能匹配、多账户登录、第三方应用集成、复杂启动脚本定制、资源实时查看等非常多的功能。

#### ◆ 客观评价

目前来看，它可能是**国内集成度最高的超算登录工具。**





## “HPC云” 平台

- 自主搭建的资源和任务管理平台，为了方便用户的系统使用。
- 主要包括统一资源查询、远程登录系统、模块化软件环境、Web文件管理、数据快递服务、可视化应用、多方位技术支持等**6大核心功能**。网址：<https://hpc.nscg-tj.cn/>。

### 统一资源查询

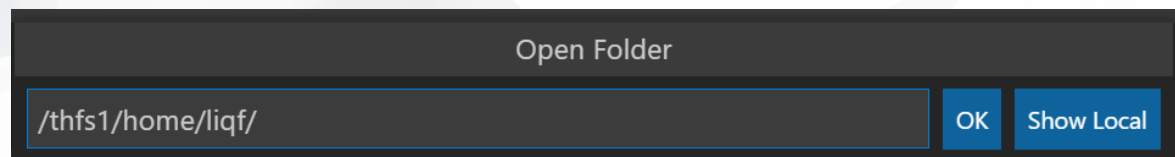
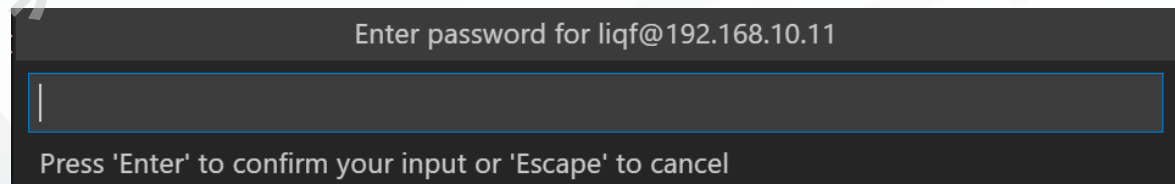
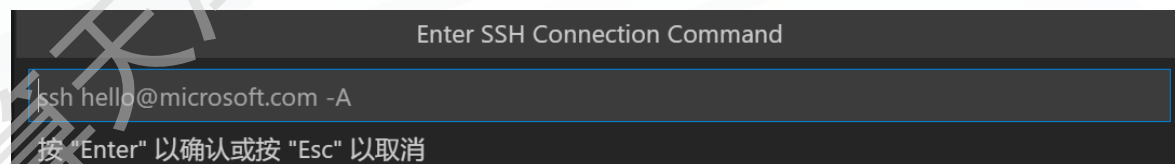


### 远程登陆系统



使用vscode的远程连接功能，可以在vscode界面实现对代码/脚本的编辑功能，同时也提供内置shell操作界面。其优势在于：

- 1、节约vim的学习成本，转换为更为熟悉的windows操作方式（可添加vim插件）
- 2、更便于做代码和脚本的调试
- 3、可以自动保存
- 4、与单独打开的远程ssh应用不冲突
- 5、不存在编码冲突（常见于本地编辑上传后）



```
> .cache  
> .cmake  
> .conda  
> .config
```

```
$ .bash_aliases  
≡ .bash_history  
$ .bash_logout  
$ .bashrc
```

- LINUX系统是功能十分强大的系统，但系统地全面学习该系统的投入产出比并不高，建议具备基础使用能力后，根据需求学习特定功能（搜索引擎关键词检索：linux/shell + 命令/描述）；
- 超算系统的使用模式是不断进步的，结合中心开发的工具和第三方工具可以极大加速科研进度；
- 如果您在使用过程中遇到问题、使用模式限制，或者有新鲜的使用模式和使用思路，请随时和中心负责人联系，我们会解决、进步，并将共性技术推送给其他用户以提升科研效率







国家超级计算天津中心

谢谢