

并行计算实验

汤善江 副教授

天津大学智能与计算学部

tashj@tju.edu.cn

<http://cic.tju.edu.cn/faculty/tangshanjiang/>



实验环境介绍

- 操作系统
 - CentOS 7.6
- 编译环境
 - GCC 4.8.5
 - gcc,g++,gfortran等
 - Intel 19.1.0.166
 - icc, icpc, ifort等
 - Intel MPI 5.0.2.044
 - Mpicc,mpicxx等等（基于GCC）
 - Mpiicc,mpiicpc等等（基于Intel）
- 作业管理系统
 - Torque 4.2.10



使用方式

- 通过远程登录方式连接集群
 - 不支持图形界面
 - Xshell
 - SSH Secure Shell Client
 - Putty
- 文件传输客户端软件
 - XFTP
 - Secure File Transfer Client
 - WinSCP

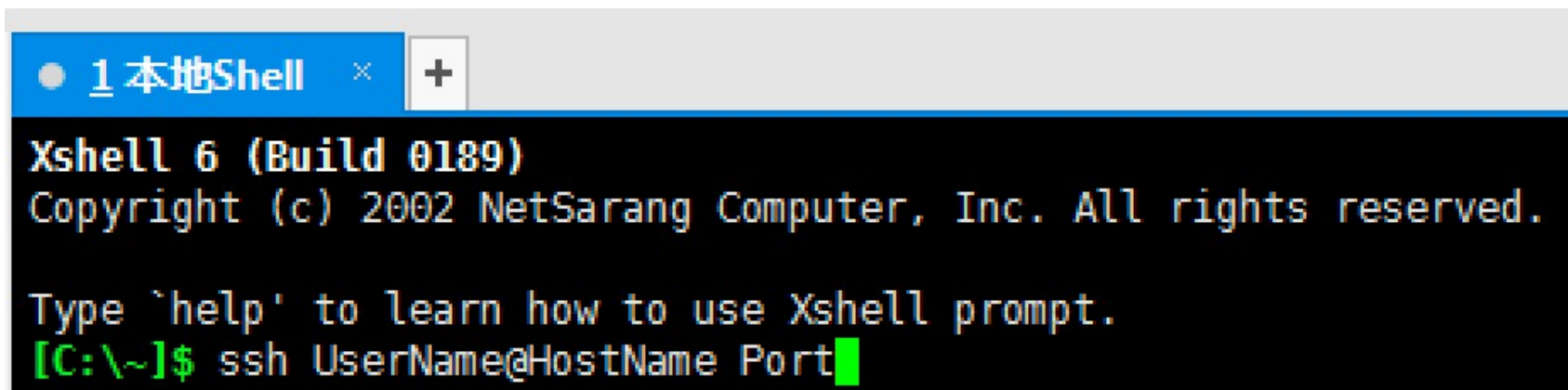


使用流程（演示）

- 登录集群
- 编写程序
- 编译程序
- 编写PBS脚本
- 提交作业
- 查看运行状态
- 查看结果



登录集群



The image shows a screenshot of an Xshell terminal window. The title bar at the top is blue and contains the text '1 本地Shell' followed by a close button (X) and a plus sign (+). The terminal content is as follows:

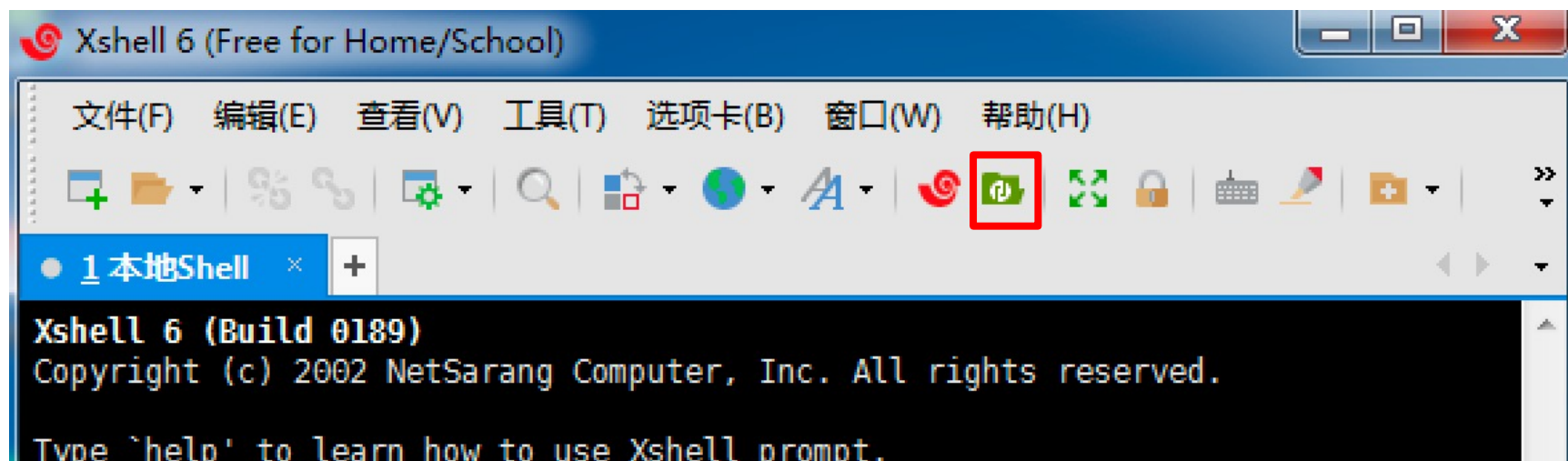
```
Xshell 6 (Build 0189)
Copyright (c) 2002 NetSarang Computer, Inc. All rights reserved.

Type `help' to learn how to use Xshell prompt.
[C:\~]$ ssh UserName@HostName Port
```

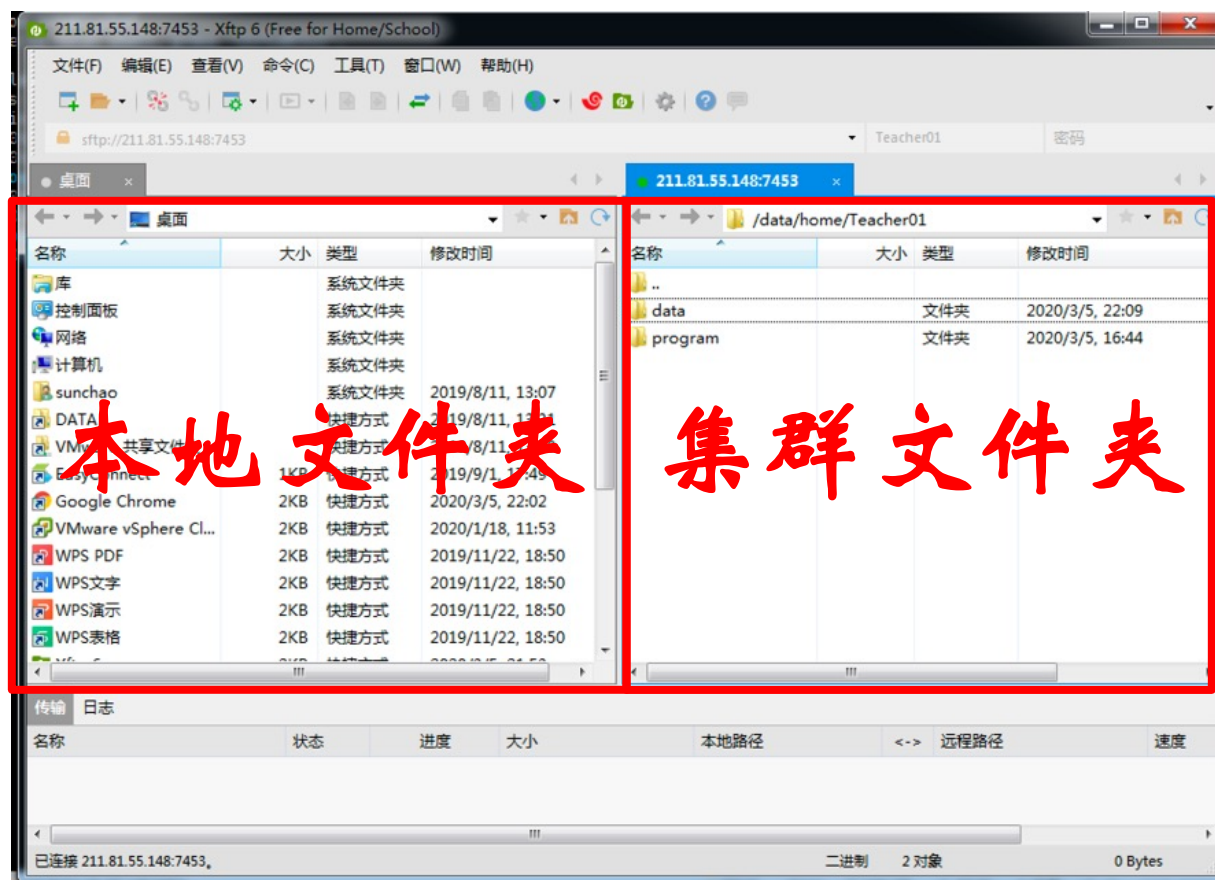
- Host Name: 211.81.50.36
- User Name: BMXXX, 如BM001
- Password: 学号
- Port: 9022



文件传输



文件传输



常用Linux命令

- ls 列出当前目录下文件
 - 如: ls -al
- mkdir 创建目录 (文件夹)
 - 如: mkdir data
- cd 切换工作目录
 - 如: cd data/
- pwd 查看当前目录
- rm 删除文件或目录
- passwd 修改登陆密码
- exit 退出登陆



一般流程

- 先在本机（可以是Windows）编写好串行程序代码test.cpp，能够正确运行
- 通过文件传输界面，将文件传到集群上，最好在data目录下。
- 能过命令行界面切换到相应目录，上一步的data
 - cd data
- 编译程序
 - icc -o test test.cpp
- 执行test程序
 - ./test



编译命令

- 示例(多线程与多线程):
 - 编译pthread多线程程序test.c(C程序)生成test
 - `icc -pthread -o test test.c`
 - 编译pthread多线程程序test.cpp(C++程序)生成test
 - `icpc -pthread -o test test.cpp`
 - 编译openmp多线程程序test.c(C程序)生成test
 - `icc -openmp -o test test.c`
 - 编译openmp多线程程序test.cpp(C++程序)生成test
 - `icpc -openmp -o test test.cpp`
 - 编译MPI多进程程序mpi.c(C程序)生成mpi
 - `mpiicc -o mpi mpi.c`
 - 编译MPI多进程程序mpi.cpp(C++程序)生成mpi
 - `mpiicpc -o mpi mpi.cpp`



使用队列

- PBS脚本示例（串行或多线程程序）
 - `#!/bin/bash`
 - `#PBS -N test`
 - `#PBS -q qstudent`
 - `#PBS -l nodes=1:ppn=1`
 - `#PBS -j oe`
 - `cd $PBS_O_WORKDIR`
 - `date +%s.%N` # 输出程序开始时间
 - `./test` # 如果程序需要参数，可直接写上
 - `date +%s.%N` # 输出程序结束时间



使用队列

- PBS脚本示例 (MPI并行)

- `#!/bin/bash`
- `#PBS -N test`
- `#PBS -q qstudent`
- `#PBS -l nodes=2:ppn=4`
- `#PBS -j oe`

- `cd $PBS_O_WORKDIR`
- `procs=$(cat $PBS_NODEFILE | wc -l)`
- `date +%s.%N # 输出程序开始时间`
- `mpirun -np $procs -machinefile $PBS_NODEFILE ./test # 如果程序需要参数, 可添加`
- `date +%s.%N # 输出程序结束时间`



使用队列

- PBS脚本的编写

- 第一行: `#!/bin/bash`
- `#PBS -N 作业名字(test)`
- `#PBS -q 使用队列(qstudent)`
- `#PBS -l 申请资源(nodes=1:ppn=8)`
- `#PBS -j oe`
- `cd $PBS_O_WORKDIR`
- `procs=$(cat $PBS_NODEFILE | wc -l)`
- 运行程序
 - `mpirun -np $procs -machinefile $PBS_NODEFILE ./test`
 - 不要使用后台执行方式(包含`&`字符)



使用队列

- 提交作业
 - qsub test.pbs
- 查看作业
 - qstat -R
 - qstat -f jobid
- 删除作业
 - qdel jobid



演示

```
TT01@node63:~> cd data/
TT01@node63:~/data> ls
test.pbs  test.pbs~
TT01@node63:~/data> qsub test.pbs
35.node63
TT01@node63:~/data> qstat
```

Job id	Name	User	Time Use	S	Queue
35.node63	test	TT01		0	R qstudent

```
TT01@node63:~/data> qstat -R
```

node63:

Job ID	FAST	PFS	Username	Queue	NDS	TSK	Req'd Memory	Req'd Time	Elap S	Time
35.node63	--	--	TT01	qstudent	1	4	--	01:00	R	--

```
TT01@node63:~/data> qdel 35
```



查看结果

- 作业结果文件
 - (name).o(id)
- 上例的结果为：
 - test.o35
- 可以在PBS脚本的最后一行加上重定向
 - ./test >& run.log



多线程代码示例

使用命令行参数的程序

```
#include <stdio>
#include <stdlib>
using namespace std;

int main(int argc, char *argv[]) {
    if (argc != 1) {
        printf("error\n");
    }
    int threadnum=atoi(argv[1]);
    ....
    return 0;
}
```

直接运行的方法: ./test 8

PBS脚本可以用变量: ./test \$procs

