

# 异构计算实验指导

2023-2024 学年第一学期

指导老师：汤善江 孙超

实验指导：续晟羽

# 异构计算实验指导

2023-2024 学年第一学期

## 一、实验要求及评分标准

本课程实验目的为提升学生对异构计算的理解认识，培养学生编写 GPU 异构程序的能力，加深对 CUDA 和 OpenCL 异构并行编程的理解认识。

实验课程需要上交实验报告，报告评分标准如下：

实验	内容要求	评分比例	占总分比例
实验一	实验内容	10%	20%
	实验原理	10%	
	程序流程图	30%	
	实验结果及分析	40%	
	实验总结	10%	
实验二	实验内容	10%	40%
	实验原理	10%	
	程序流程图	30%	
	实验结果及分析	40%	
	实验总结	10%	
实验三	实验内容	10%	40%
	实验原理	10%	
	程序流程图	30%	
	实验结果及分析	40%	
	实验总结	10%	
	实验结果	30%	
	实验总结	10%	

其中，实验原理包括：**实验数学计算模型**和**实现方法**；实验结果及分析应包括：**实验结果数据**、**加速比曲线**和**实验结果分析**。前两次实验采用 CUDA 进行 GPU 编程，第三次实验的采用 OpenCL 编程。

## 二、实验环境介绍及使用方法

### 1. 通过 VPN 接入内部网络

可下载 OpenVPN 客户端，直接导入配置文件（config.ovpn）。

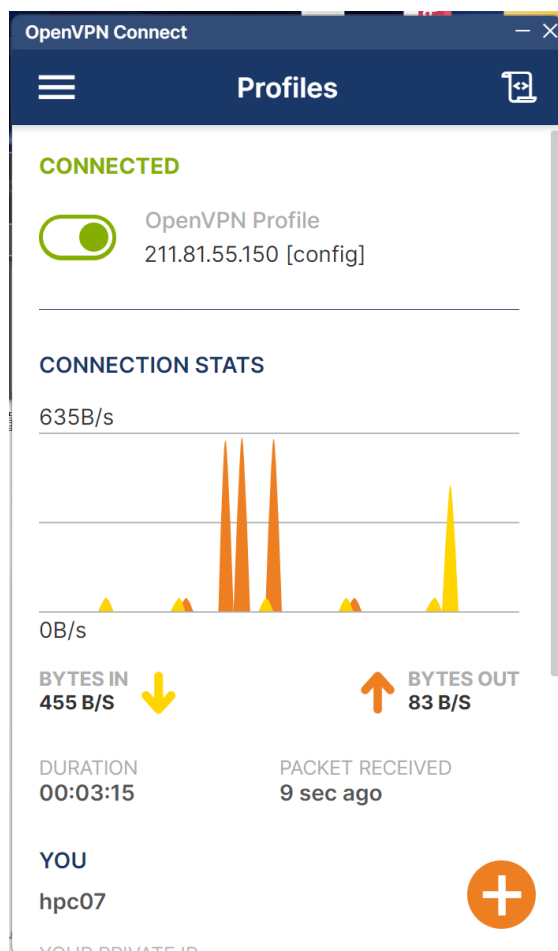
OpenVPN 下载地址：<https://openvpn.net/community-downloads/>



用户名：hpc07

密码：user@1895

输入认证信息（用户名和密码）进行连接，连接成功后，如下图：



## 2. GPU 服务器远程登录及所需软件

i. 通过远程登录方式连接 GPU 服务器

- XShell 6
- SSH Secure Shell Client
- Putty
- FinalShell

实验环境不支持图形界面。

ii. 文件传输客户端软件

- Xftp 6
- Secure File Transfer Client
- WinSCP
- FinalShell

PS: 除此之外, Linux 可以直接使用 OpenSSH 进行登录, Mac OS 也可以使用对应的 ssh 命令登录链接到集群。

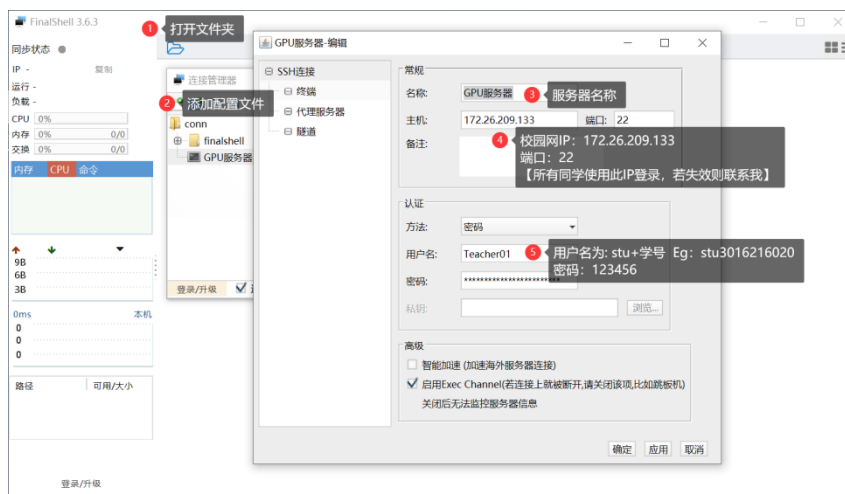
软件下载链接: XShell 6: <https://www.netsarang.com/zh/xshell-download/>

Xftp 6: <https://www.netsarang.com/zh/xftp-download/>

FinalShell: [http://www.hostbuf.com/downloads/finalshell\\_install.exe](http://www.hostbuf.com/downloads/finalshell_install.exe)

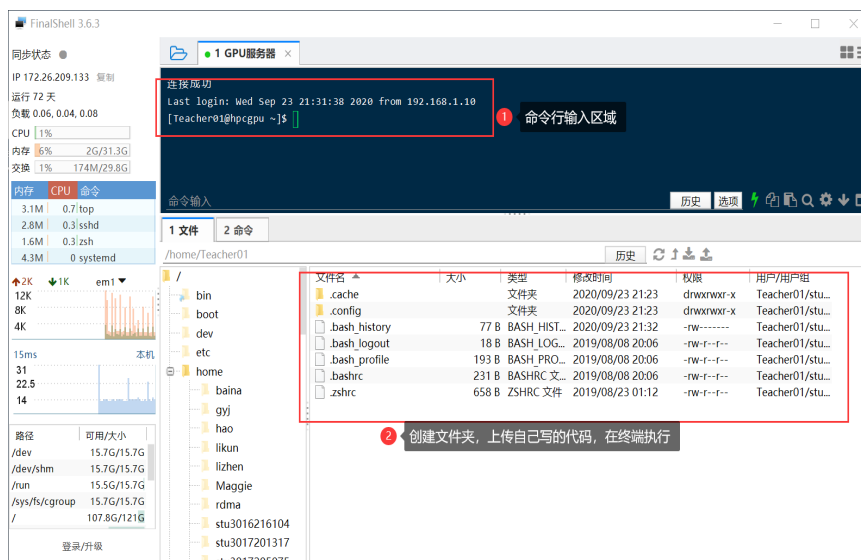
### 3. 登录 GPU 服务器

- 登录集群方式如图所示（以 FinalShell 为例）：

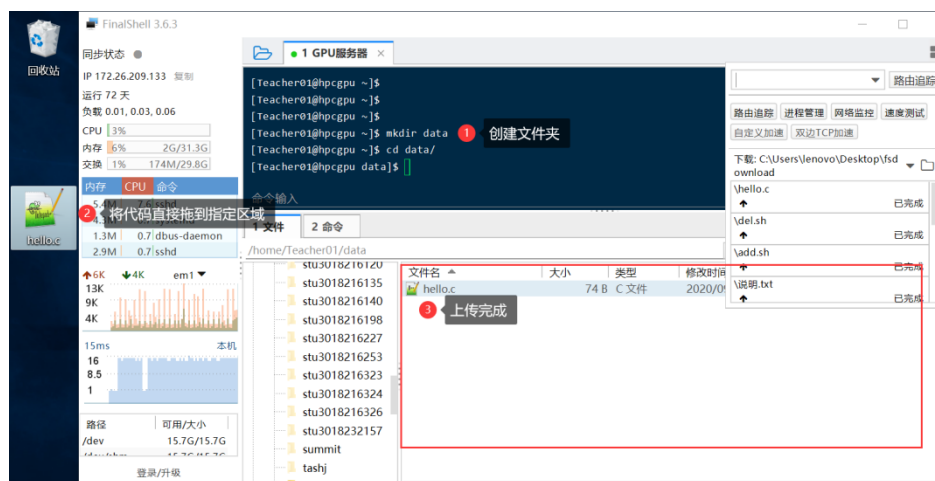


按照上图描述进行配置，其中用户名为（tju+学号后三位），密码为自己学号。登入后，可以选择修改密码（命令行输入 `passwd`），如果忘记密码，请联系老师或助教进行重置。

服务器 IP 地址为：10.10.112.44 或 10.10.112.45；端口号为：22。点击确定登录即可。



- 文件传输：



如图所示，直接将代码拖入指定区域，即可以交互方式实现本机与服务器之间的文件传输功能。

下载文件，直接右键指定文件下载即可。

#### 4. 常用 Linux 命令

- `ls`: 列出当前文件夹下文件。如: `ls -al`
- `mkdir`: 新建文件夹。如: `mkdir data`
- `cd`: 切换工作文件夹。如: `cd data/`
- `pwd`: 查看当前文件夹绝对路径
- `rm`: 删除文件或文件夹（需要加上 `-r` 参数）
- `passwd`: 修改登录密码
- `exit`: 退出登录

#### 5. 实验环境

##### i. 操作系统

CentOS 7.9

##### ii. 编译环境

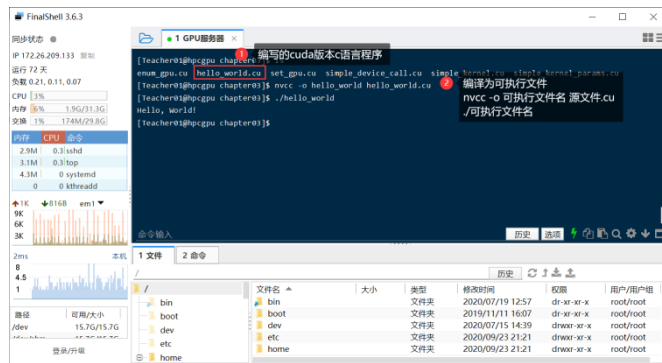
- GCC 4.8.5: `gcc`, `g++`等
- CUDA 12.2, NVCC 12.2

```
[Teacher01@hpcgpu chapter03]$ ./gpu
--- General Information for device 0 ---
Name: Tesla K40c
Compute capability: 3.5
Clock rate: 745000
Device copy overlap: Enabled
Kernel execution timeout : Disabled
--- Memory Information for device 0 ---
Total global mem: 11996954624
Total constant Mem: 65536
Max mem pitch: 2147483647
Texture Alignment: 512
--- MP Information for device 0 ---
Multiprocessor count: 15
Shared mem per mp: 49152
Registers per mp: 65536
Threads in warp: 32
Max threads per block: 1024
Max thread dimensions: (1024, 1024, 64)
Max grid dimensions: (2147483647, 65535, 65535)
```

### iii. 示例

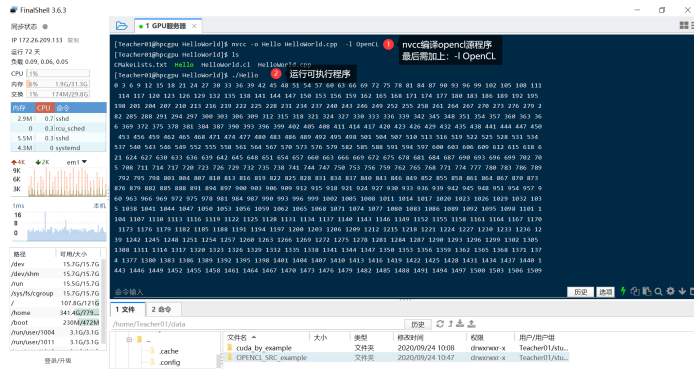
## 一、基于 CUDA 的编程执行方式:

nvcc -o hello hello.cu



## 二、基于 OpenCL 的编程执行方式:

nvcc -o hello HelloWorld.cpp -I OpenCL



## 三、实验题目

### 1.基于 CUDA 的 GPU 计算 PI 值

#### i. 积分法

$$\text{计算公式: } \pi = \int_0^1 \frac{4}{1+x^2} dx \approx \sum_{0 \leq i \leq N} \frac{4}{1+(\frac{i+0.5}{N})^2} \times \frac{1}{N}$$

ii. 幂级数计算方法

计算公式： $\pi = 4 \times \arctan(1) = 4 \times (1 - \frac{1}{3} + \frac{1}{5} - \dots + \frac{(-1)^{n+1}}{2n-1} - \dots)$

## 2. 基于 CUDA 的 GPU 实现矩阵的幂

对于一个  $m \times m$  的方阵  $A = [a_{ij}]$ ，计算  $A$  的  $n$  次幂。

首先，生成一个  $m \times m$  的方阵  $A = [a_{ij}]$ ，保证每行每列元素之和满足  $(0,1)$

i. 暴力算法

$n$  个矩阵相乘

ii. 高效算法

利用矩阵乘法的结合律

## 3. 基于 OpenCL 实现矩阵的幂

用 OpenCL 编程模型实现矩阵  $A$  的  $n$  次幂。

要求实现暴力算法和高效算法，同时对比分析一下相同 OpenCL 程序分别运行在纯多核 CPU 环境下以及异构 GPU 环境下的性能。