Univ. of Pittsburgh

CS1550 Recitation #12

Zhipeng (Patrick) Luo

ZHL78@pitt.edu

Nov. 30th & Dec. 4th, 2015

Computer Science Department

Univ. of Pittsburgh

Outline

- Introduction to FUSE
- The first FUSE example
- Introduction to project 4
- Coding project 4
- Quiz 2 feedback

Part 0

Univ. of Pittsburgh

Introduction to FUSE

Part 1

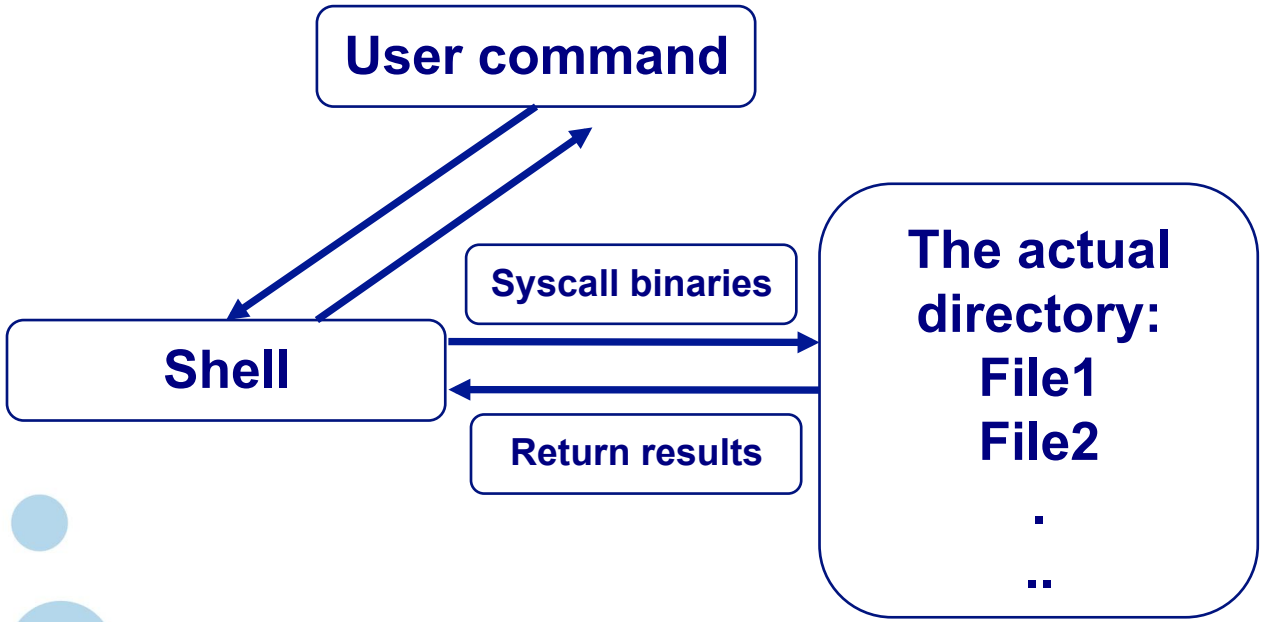
- FUSE is a Linux kernel extension that allows for a user space program to provide the implementations for the various file-related syscalls.
- Provides us with a virtual coding environment to implement various syscalls in the user space, without modifying any kernel code
- In essence, just a simulation of file systems.
  - In this project we only need to modify several file system functions!

Univ. of Pittsburgh

Introduction to FUSE

Part 1

- Below is a common command processing WITHOUT using FUSE:

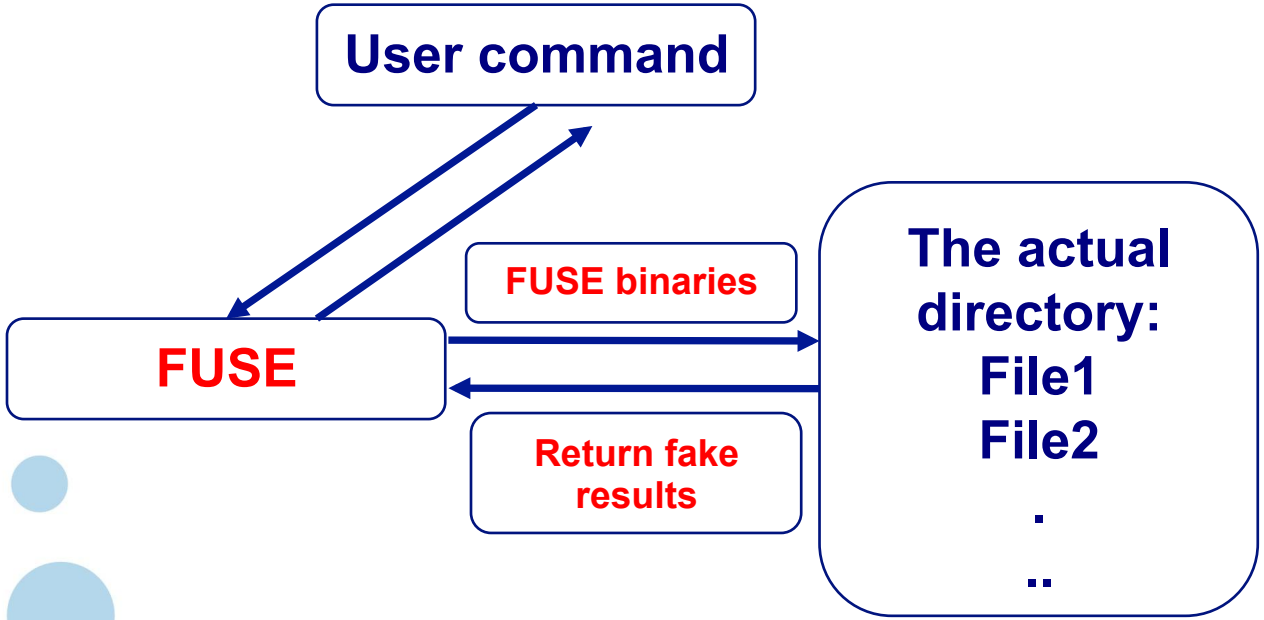


Univ. of Pittsburgh

Introduction to FUSE

Part 1

- However when we run FUSE, it temporarily takes over the partial shell functions and returns us “fake” outputs.



- So our task is to customize these FUSE binaries.

Univ. of Pittsburgh

The first FUSE example

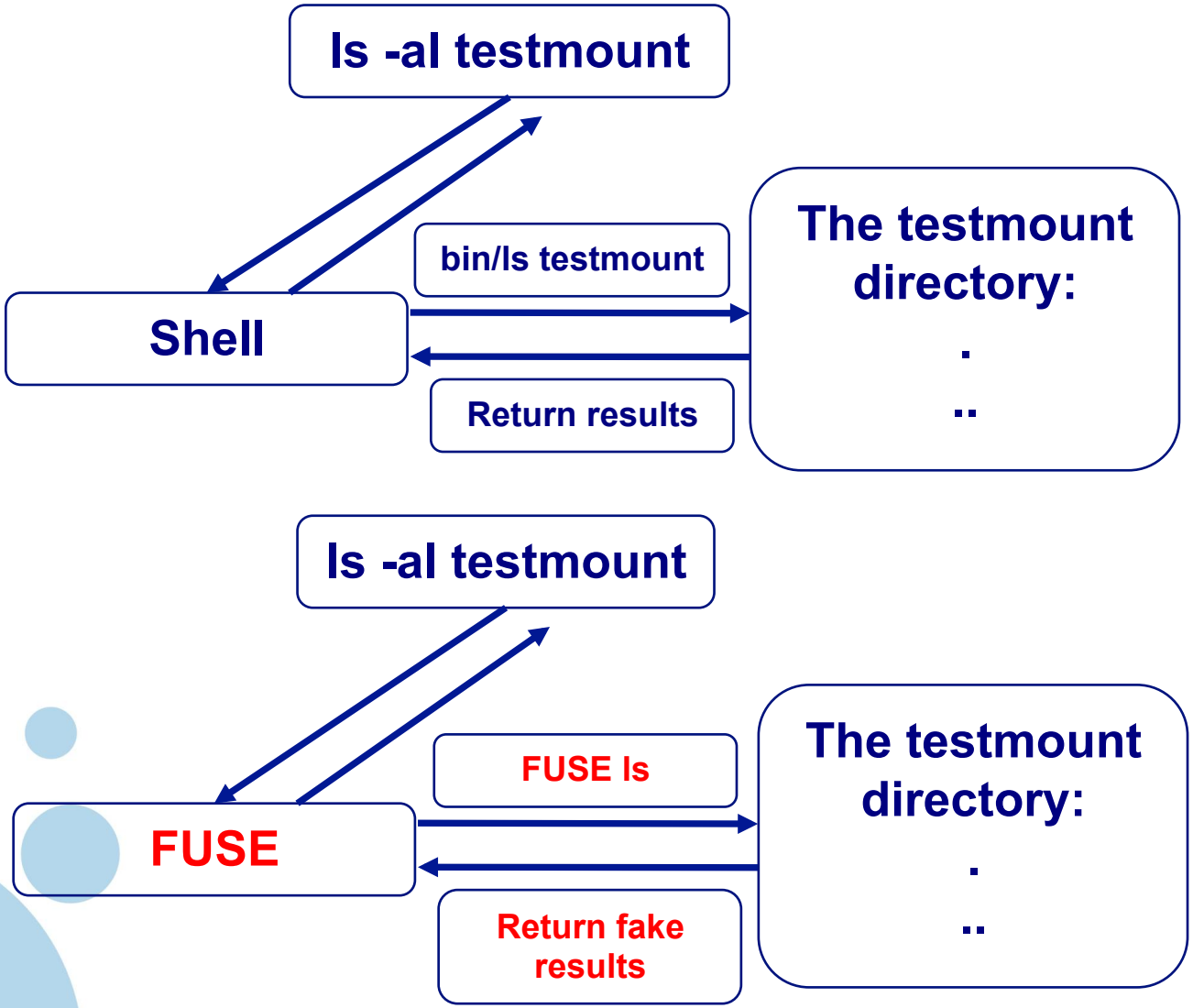
Part 2

- Demo of the first example
- What we did:
  - *mkdir testmount:* Created a mount point called **testmount**, the root of a new file system.
  - *ls -al testmount*: normal **readdir** via shell
  - *./hello testmount*: the application **hello** of FUSE takes over shell.
  - *ls -al testmount*: FUSE **readdir**
  - *cat testmount/hello*: read a regular file **hello**
  - **fusermount -u testmount/**

Univ. of Pittsburgh

The first FUSE example

Part 2



Univ. of Pittsburgh

The first FUSE example

Part 2

- Let's look at the FUSE hello.c
- hello_getattr()
- **hello_readdir()**
  - The ls command
- hello_open()
- **hello_read()**
  - The cat command
- main()
  - Trivial in this project

Univ. of Pittsburgh

The first FUSE example

Part 2

- int hello_readdir(path, filler, buf)
- This function should look up the input path, ensuring that it is a directory, and then list the contents.
- To list the contents, you need to use the filler() function
- Return values
  - 0 on success
  - -ENOENT if the directory is not valid or found

Univ. of Pittsburgh

The first FUSE example

Part 2

```
static const char *hello_str = "Hello World!\n";
static const char *hello_path = "/hello";
static int hello_readdir(const char *path, void *buf,
fuse_fill_dir_t filler, off_t offset, struct fuse_file_info *fi) {
    (void) offset;
    (void) fi;

    if (strcmp(path, "/") != 0)
        return -ENOENT;

    filler(buf, ".", NULL, 0);
    filler(buf, "..", NULL, 0);
    filler(buf, hello_path + 1, NULL, 0);
    return 0;
}
```

Univ. of Pittsburgh

The first FUSE example

Part 2

- int hello_read(path, offset, buf)
- This function should read the data in the file denoted by path into buf, starting at offset.
- Return values:
  - size read on success
  - -EISDIR if the path is a directory

Univ. of Pittsburgh

The first FUSE example

Part 2

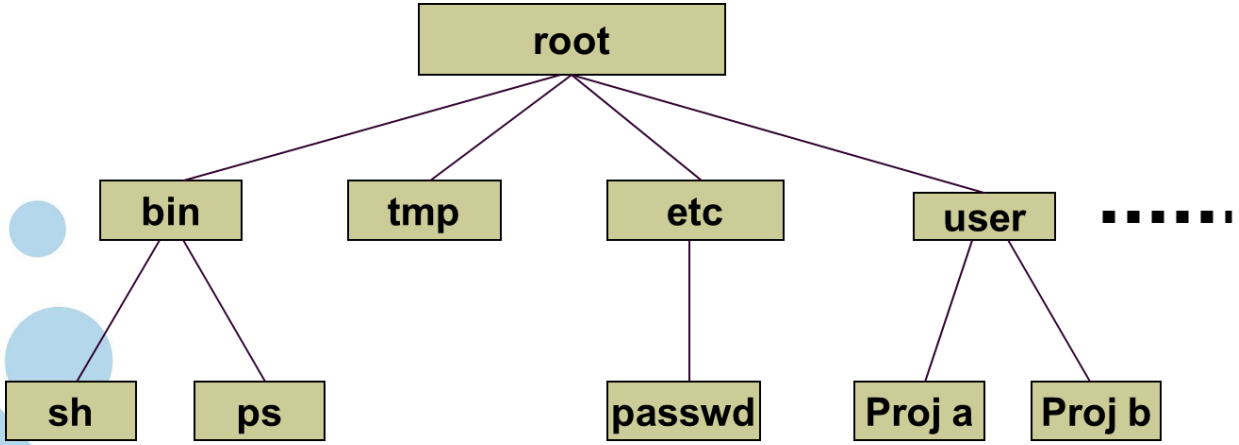
```
static const char *hello_str = "Hello World!\n";
static const char *hello_path = "/hello";
static int hello_read(const char *path, char *buf, size_t
size, off_t offset, struct fuse_file_info *fi) {
    size_t len;
    (void) fi;
    if(strcmp(path, hello_path) != 0)
        return -ENOENT;
    len = strlen(hello_str);
    if (offset < len) {
        if (offset + size > len)
            size = len - offset;
        memcpy(buf, hello_str + offset, size);
    } else
        size = 0;
    return size;
}
```

Univ. of Pittsburgh

Project 4 file system structure

Part 3

- Create a simple 3-level file system.
- The root directory “/” will only contain other subdirectories, and no regular files
- The subdirectories will only contain regular files, and no subdirectories of their own



Univ. of Pittsburgh

Part 3

File system disk management

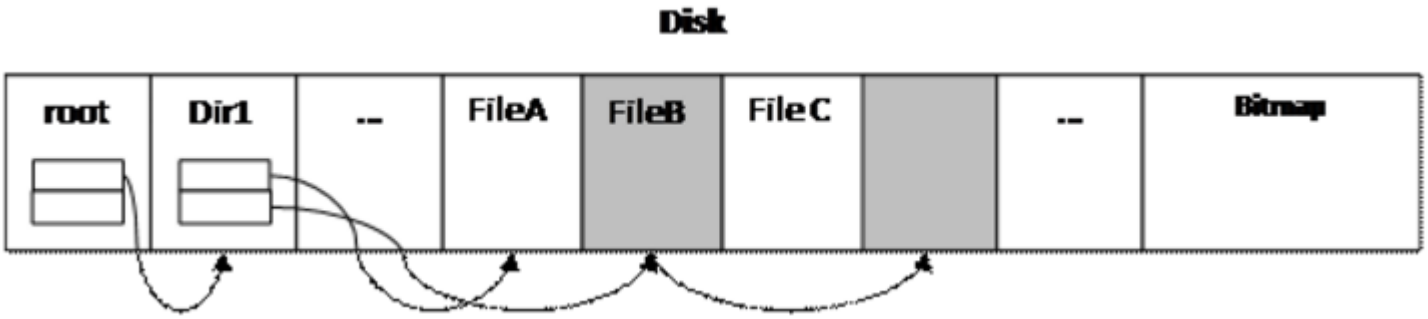
- Our physical disk is the “.disk” file
  - dd bs=1K count=5K if=/dev/zero of=.disk
  - 5MB in total with block size = 512B
- There are 3 types of block
  - Block 0 is for the root block only
    - It contains its own attributes and pointers to blocks of its subdirectories
  - Each subdirectory block takes exactly one block
    - Similarly, it contains its own attributes and pointers to iNode of its files
  - Regular file block
    - Pointer to the next block if possible
    - The actual data.

Univ. of Pittsburgh

File system disk management

Part 3

- There are 3 types of block (iNode)
  - Block 0, the root block
  - Subdirectory block(s)
  - Regular file block(s)

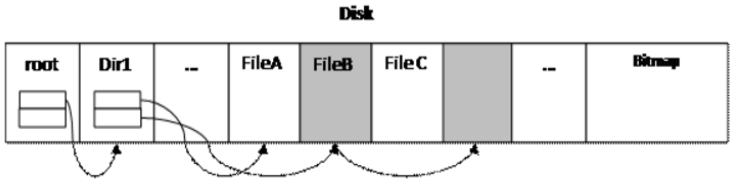


- Bitmap section is the meta-data of blocks
  - Like which blocks are occupied or not

Univ. of Pittsburgh

The root block

Part 3



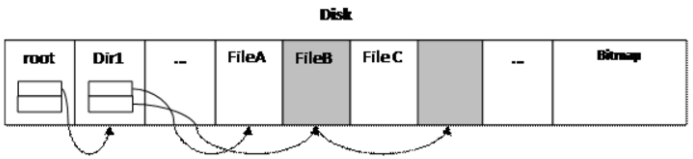

```
struct cs1550_root_directory
{
        int nDirectories; //How many subdirectories are in the root
        struct cs1550_directory
        {
                char dname[MAX_FILENAME + 1];   //directory name
                long nStartBlock;  //where the directory block is on disk
        } __attribute__((packed)) directories[MAX_DIRS_IN_ROOT]
        //There is an array of these


        //This is some space to get this to be exactly the size of the disk block.
        //Don't use it for anything.
        char padding[BLOCK_SIZE - MAX_DIRS_IN_ROOT * sizeof(struct cs1550_directory) - sizeof(int)];
} ;
```

Univ. of Pittsburgh

The subdirectory block

Part 3



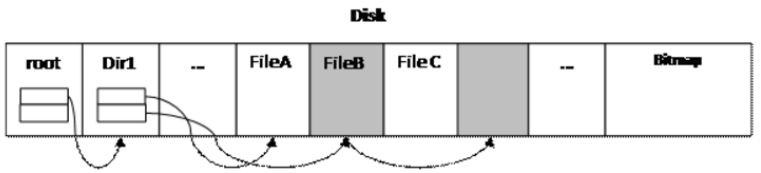

```
struct cs1550_directory_entry
{
	int nFiles;	//How many files are in this directory.
	struct cs1550_file_directory
	{
		char fname[MAX_FILENAME + 1];	//filename
		char fext[MAX_EXTENSION + 1];	//extension
		size_t fsize;			//file size
		long nStartBlock;	//where the first block is on disk
	} __attribute__((packed))
files[MAX_FILES_IN_DIR];			//There is an array of
these


	//This is some space to get this to be exactly the size of the
disk block.
	//Don't use it for anything.
	char padding[BLOCK_SIZE - MAX_FILES_IN_DIR *
sizeof(struct cs1550_file_directory) - sizeof(int)];
} ;
```

Univ. of Pittsburgh

The regular file block

Part 3




```
struct cs1550_disk_block
{
        //The next disk block, if needed. This is the next pointer in the linked
        //allocation list
        long nNextBlock;

        //And all the rest of the space in the block can be used for actual data
        //storage.
        char data[MAX_DATA_IN_BLOCK];
};
```

Univ. of Pittsburgh

Start coding this **** project

Part 4

- In general, what you ONLY need to do is implement several file operations:
  - cs1550_getattr()
  - cs1550_mkdir()
  - cs1550_readdir()
  - cs1550_mknod()
  - cs1550_write()
  - cs1550_read()
  - cs1550_unlink()
- Easy to start with a good environment provided.
- But never start until you have a decent understanding of things already given.

Univ. of Pittsburgh

Part 4

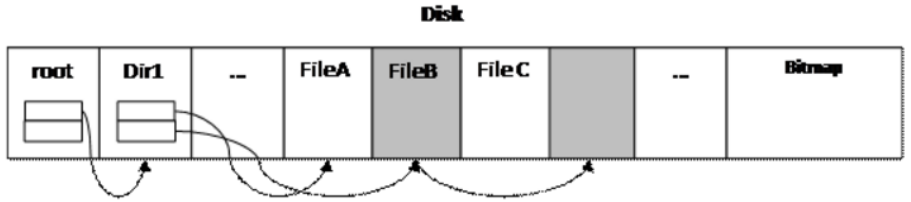
Getattr()

```
static int cs1550_getattr(const char *path, struct stat *stbuf)
{
        // struct stat *stbuf is the buffer to store your getattr results.
        // return value is just an indicator of success.
        //is path the root dir?
         if (strcmp(path, "/") == 0) {
                // store root dir permission and # of links into stbuf
         } else {
                sscanf(path, "/%[^/]/%[^.].%s", directory_name, filename, extension);
                //Check if name is subdirectory
                //OtherwiseCheck if name is a regular file
                //Else that path doesn't exist
         }
        return corresponding success or not.
}
```

Univ. of Pittsburgh

A getattr() example

Part 4



- getattr("/Dir1/FileA.html")
- Is this a root? No
- Directory_name="Dir1" & filename="FileA" & extension="html"
- Is "Dir1" a valid subdirectory?
  - Load block 0 (the first 512B of .disk) into memory as a **struct cs1550_root_directory** and check its array if there is a "Dir1"
  - Yes, its block is on block 1
- Is "FileA.html" a valid file in 'DirA'?
  - Load block 1 into memory as a struct **cs1550_directory_entry and check** its array if there is a "FileA" with extension "html"?
  - Yes, its size is 100B, say.
- Modify stbuf
  - stbuf->st_mode = S_IFREG | 0666;
  - stbuf->st_nlink = 1; //file links
  - stbuf->st_size = 100;
  - Return 0;

Univ. of Pittsburgh

A mkdir() example

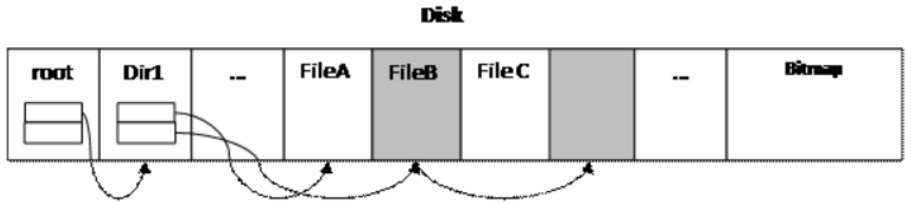
Part 4

- mkdir("/Dir222")
- Is this the root? No
- Directory_name="Dir222" & filename="" & extension=""
- Is "Dir222" an existing subdirectory?
  - Load block 0 (the first 512B of .disk) into memory as a **struct cs1550_root_directory** and check its array if there is a "Dir222"
  - If yes return error
- Otherwise create the new subdirectory
  - Load your bitmap into memory and allocate an empty block for this directory
  - Malloc a new **cs1550_directory_entry** for this directory (in memory)
  - Add a new link to this directory to the root block (in memory)
  - Write your updated bitmap, the root block and the new block back to your ".disk" file

Univ. of Pittsburgh

A readdir() example

Part 4



- readdir("/Dir1")
- Is this a root? No
- Directory_name="Dir1" & filename="" & extension=""
- Is "Dir1" a valid subdirectory?
  - Load block 0 (the first 512B of .disk) into memory as a **struct cs1550_root_directory** and check its array if there is a "Dir1"
  - Yes it's in block 1
- Load block 1 into memory
  - Use filler to fill '.' and '..' into your result buffer
  - Use filler to fill all files it contains into your result buffer.
  - Return sucesss.

Univ. of Pittsburgh

Quiz 2

- 40/47 submissions
- Average=37 on submissions
- Min=17
- Max=50

Part 5

Univ. of Pittsburgh

Thanks!

Zhipeng (Patrick) Luo

ZHL78@pitt.edu

Nov. 30th & Dec. 4th, 2015

Computer Science Department