HW #3

Multi-Threads with Synchronization Tools

Yunmin Go

School of CSEE

HANDONG GLOBAL UNIVERSITY

Requirement

- Implement a multi-thread based word search program

- Required programming skills
  - Command line argument
  - Multi-thread programming
  - Synchronization tools

Requirements

- Program should follow usage below

* For the command line argument, find out about 'getopt()' and 'optarg'.

```
$ ./mtws -b <buffer size> -t <num threads> -d <directory> -w <word>
```

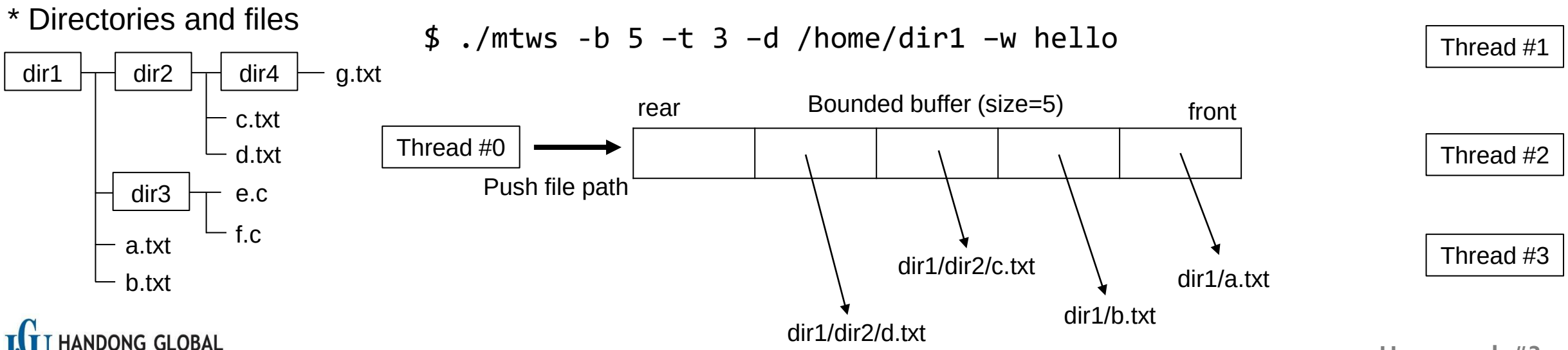
  - `-b` : size of bounded buffer
  - `-t` : number of threads searching for a word (except for main thread)
  - `-d` : search directory (search all of sub directories)
  - `-w` : search word (case-insensitive)

```
yunmin@mcnl-server:~/workspace/os/hw3$ ./mtws
Usage: ./mtws
  -b : bounded buffer size
  -t : number of threads searching word (except for main thread)
  -d : search directory
  -w : search word
```

* Program name is 'mtws'
* Search files are text file
* Search word is case-insensitive

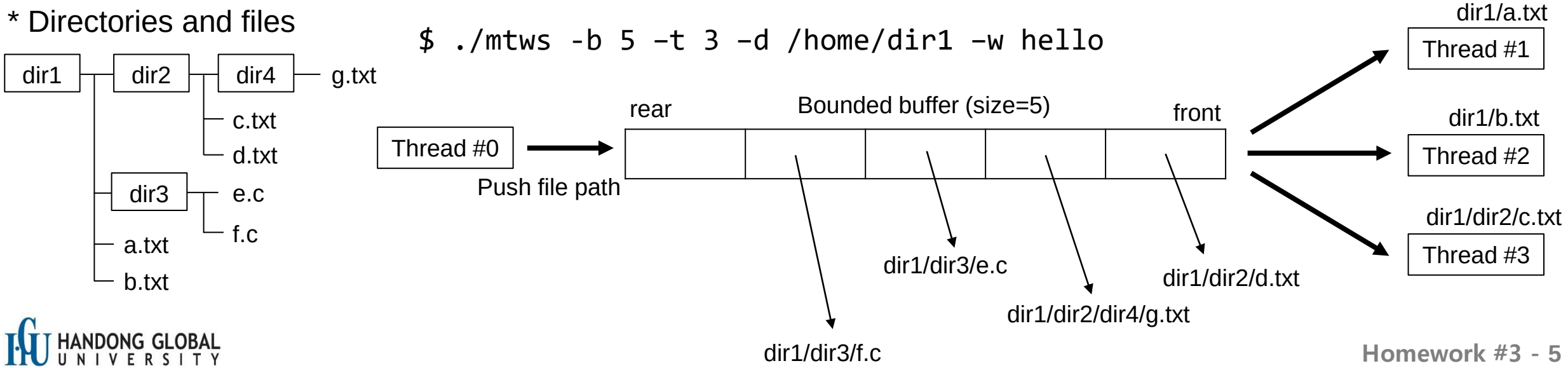
Requirements

- Program creates a bounded buffer with size given in parameter '–b'.
- One thread finds all files in search directory given in parameter '–d' and put file name (including path) into the bounded buffer.
- Program creates threads as many as the parameter given in parameter '-t'.



Requirements

- The created thread gets a file name from the bounded buffer. And then, it searches a word (parameter `'-w'`) in the selected file and shows number of found words.
- The created thread should repeat above operation until it receives thread cancel request from main thread.



Requirements

- The order in which items are inserted into the bounded buffer and the order in which threads complete processing them can be different.
- You must use mutex (or semaphore) and conditional variables when the threads access the bounded buffer.
- Finally, the program should aggregate all the numbers of found and print the results.
  - However, there are no requirements about how the program aggregates the numbers of found. It is up to programmer's implementation.

Requirements

- Expected results

Thread ID can be obtained by `pthread_self()`

```
yunmin@mcnl-server:~/workspace/os/hw3$ ./mtws -b 10 -t 3 -d /home/yunmin/os/dir1 -w printf
Buffer size=10, Num threads=3, Directory=/home/yunmin/os/dir1, SearchWord=printf
[Thread 139857109804800] started searching 'printf'...
[Thread 139857126590208] started searching 'printf'...
[Thread 139857118197504] started searching 'printf'...
[Thread 139857109804800-0] /home/yunmin/os/dir1/DateClient.java : 0 found
[Thread 139857118197504-2] /home/yunmin/os/dir1/dir3/simple-shell.c : 1 found
[Thread 139857126590208-1] /home/yunmin/os/dir1/dir3/win32-pipe-child.c : 2 found
[Thread 139857109804800-3] /home/yunmin/os/dir1/dir3/win32-pipe-parent.c : 3 found
[Thread 139857118197504-4] /home/yunmin/os/dir1/dir3/unix_pipe.c : 3 found
[Thread 139857118197504-7] /home/yunmin/os/dir1/dir2/fig3-32.c : 0 found
[Thread 139857126590208-5] /home/yunmin/os/dir1/dir3/shm-posix-producer.c : 4 found
[Thread 139857118197504-8] /home/yunmin/os/dir1/dir2/fig3-30.c : 1 found
[Thread 139857109804800-6] /home/yunmin/os/dir1/dir3/shm-posix-consumer.c : 4 found
[Thread 139857126590208-9] /home/yunmin/os/dir1/dir2/fig3-34.c : 5 found
[Thread 139857109804800-11] /home/yunmin/os/dir1/dir2/dir4/multi-fork.c : 4 found
[Thread 139857126590208-12] /home/yunmin/os/dir1/dir2/dir4/multi-fork : 0 found
[Thread 139857118197504-10] /home/yunmin/os/dir1/dir2/fig3-33.c : 3 found
[Thread 139857109804800-13] /home/yunmin/os/dir1/dir2/dir4/newproc-win32.c : 1 found
[Thread 139857109804800-16] /home/yunmin/os/dir1/dir2/fig3-31.c : 0 found
[Thread 139857118197504-15] /home/yunmin/os/dir1/dir2/dir4/newproc-posix.c : 4 found
[Thread 139857109804800-17] /home/yunmin/os/dir1/dir2/fig3-35.c : 2 found
[Thread 139857126590208-14] /home/yunmin/os/dir1/dir2/dir4/pid.c : 0 found
[Thread 139857118197504-18] /home/yunmin/os/dir1/DateServer.java : 0 found
Total found = 37 (Num files=19)
```

Requirements

- Your program should be executed on Ubuntu
- Write clean source code
  - Add proper comment in your source code
  - Consider code indentation for enhancing readability
- Main source code name is `mtws.c`
  - You can include additional files in `mtws.c`.
- Upload tar.gz file on LMS by compressing all your source codes
  - File name: hw03_student id.tar.gz (ex: hw03_20400022.tar.gz)

- Due date: 11:59pm, 5/24 (Tue)