

GEBZE TECHNICAL UNIVERSITY

CSE344 HW5 REPORT

Student Name: Emre Oytun

Student No: 200104004099

Important Note – Differences Between HW4 :

- There is no difference between HW4. The codes are exactly the same. The program works exactly the same with condition variables and the barrier as I explained.

1) General Structure of Program:

Main Thread:

- It takes arguments and validates them. If it cannot validate, then it prints the usage and exits.
- If destination directory does not exist, it creates it.
- It checks if source and destination directories are subdirectory of each other. If it is, then it prints error and exits.
- It checks if given buffer size and number of workers are valid.
- It initializes the buffer with given buffer size and initializes the mutexes, semaphores, condition variables and the barrier. Then, it creates and starts the manager and worker threads.
- Main thread is waiting for manager and worker threads to finish using join, then it exits by printing the statistics and destroying the created structures like mutexes, semaphores, condition variables and the barrier. Also, it checks if there are files in buffer. If there are, it consumes all of them and closes the file descriptors.

Manager Thread:

- It traverses all subdirectories and files in “copy_directory_files” function recursively.
- In “copy_directory_files” function, it checks if there is such source directory. If there is no remaining directory to traverse, it exits. Otherwise, it increments the “num_directories” variable to keep statistic by using mutex lock to prevent race conditions.
- Later in “copy_directory_files” function, it traverses all files inside the directory by using the “readdir” function until there is no remaining file. Inside the loop, it checks if the file returned by readdir is the current directory(.) or parent directory(..), it does not do anything for these and continue to loop.
- Again in “copy_directory_files” function inside traverse loop, it checks if the file is a regular file. If it is, it increments the regular file counter, opens it by truncating and adds it to the buffer. If it is a directory, it recursively calls itself with the new path by creating the directory if it does not exist. If it is a fifo, it opens the fifo in “O_RDONLY | O_NONBLOCK” mode to prevent program being stuck. It makes fifo in the destination path and adds it to the buffer.
- An important thing here is in “buffer_add” function, in this function the manager thread does not wait on condition variable if “force_quit” is 1 meaning “ctrl+c or kill” signals have arrived. If these signals have arrived, it does not add to the buffer and exits immediately.
- When the first call of “copy_directory_files” returns, the manager thread sets “running” to 0 so that the worker threads will consume the remaining buffer and exit. It also broadcast on the “full” and “empty” condition variables to get the waiting threads up to prevent program being stuck. After that, it exits.

Worker Threads:

- Worker threads try to get a task from the buffer by waiting on the condition variable “full” in “buffer_remove” function.
- When they wake up and take an item, they copy the file from source to destination in “copy_task_file” function.
- Each thread waits on the barrier after they make copy so that they can continue at the same time after a buffer size many files are copied. Before they come to barrier, they increment the “next_phase_counter” so that the last thread coming to the barrier can make a check if the threads will continue in the next phase.
- The last thread coming to the barrier checks if “ctrl+c” signal came and “is_force_quit()” is true, then the threads should not continue in the next phase. It also checks if the manager thread exits by checking “running” variable. If the “running” is set to 0 and there is no remaining item in the buffer, they exit. If the “running” is set to 0, and there is a remaining item in the buffer they continue to the next phase to consume the remaining items.
- Also, another important thing is in the “buffer_remove” function. In this function, the threads does not wait on the condition variable if “running” is 0, or “is_force_quit” is 1 to prevent the threads being stuck on waiting. Also if it is force quit or manager exited and no remaining item in the buffer, it should not copy any file. If force quit and there are files in the buffer, first thread comes here consumes all files in the buffer and closes file descriptors.

Ctrl+c (SIGINT) and SIGTERM Signals Handling:

- When a signal is arrived, the handle function sets the “force_quit” variable to 1 so that all threads exits as soon as possible as I explained above.

Condition Variables:

- There are 2 condition variables as “full” and “empty”.
- “full”: It indicates how many full slots are there in the buffer. Worker threads waiting on this condition variable until there is at least 1 item available. When manager threads add an item, it signals to one of the threads.
- “empty”: It indicates how many empty slots are there in the buffer. Manager thread waits on this condition variable until there is at least 1 slot available to insert a new task. When a worker thread consumes an item, it signals to the manager thread.

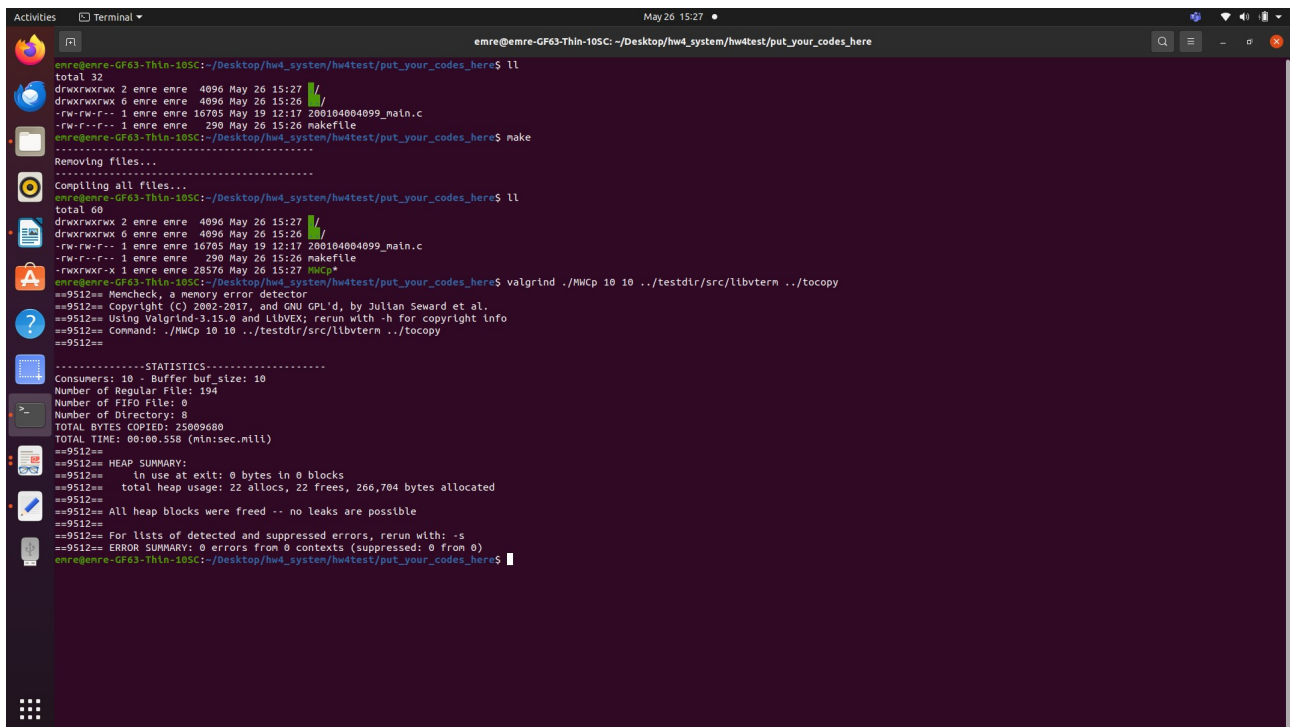
Barrier:

- It is used for controlling phases between “buffer size” much files copies. I mean it is a cycle that manager threads insert tasks to buffer and each worker thread copies exactly 1 file. After that, they come to the barrier.

2) Screenshots:

I put the uncut screenshots of the whole screen for all tests by showing the compilation process first and running the output file with proper arguments.

a) First Test:



```
emre@emre-GF63-Thin-105C: ~/Desktop/hw4_system/hw4test/put_your_codes_here$ ll
total 32
drwxrwxrwx 2 emre emre 4096 May 26 15:27 ./
drwxrwxrwx 6 emre emre 4096 May 26 15:26 ../
-rw-rw-r-- 1 emre emre 16705 May 19 12:17 200104004099_main.c
-rw-rw-r-- 1 emre emre 290 May 26 15:26 makefile
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ make
-----
Removing files...
-----
Compiling all files...
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ ll
total 60
drwxrwxrwx 2 emre emre 4096 May 26 15:27 ./
drwxrwxrwx 6 emre emre 4096 May 26 15:26 ../
-rw-rw-r-- 1 emre emre 16705 May 19 12:17 200104004099_main.c
-rw-rw-r-- 1 emre emre 290 May 26 15:26 makefile
-rwxrwxr-x 1 emre emre 28576 May 26 15:27 HWcpc*
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ valgrind ./HWcpc 10 10 ../testdir/src/libvterm ../tocopy
==9512== Memcheck, a memory error detector
==9512== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==9512== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==9512== Command: ./HWcpc 10 10 ../testdir/src/libvterm ../tocopy
==9512==
-----STATISTICS-----
Consumers: 10 - Buffer buf_size: 10
Number of Regular File: 194
Number of FIFO File: 0
Number of Directory: 8
TOTAL BYTES COPIED: 25009680
TOTAL TIME: 00:00.558 (min:sec.milli)
==9512==
==9512== HEAP SUMMARY:
==9512==      in use at exit: 0 bytes in 0 blocks
==9512==    total heap usage: 22 allocs, 22 frees, 266,704 bytes allocated
==9512==
==9512== All heap blocks were freed -- no leaks are possible
==9512==
==9512== For lists of detected and suppressed errors, rerun with: -s
==9512== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$
```

b) Second Test:

```
Activities Terminal May 26 15:28 emre@emre-GF63-Thin-105C: ~/Desktop/hw4_system/hw4test/put_your_codes_here

emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ ll
total 32
drwxrwxrwx 2 emre emre 4096 May 26 15:28 //
drwxrwxrwx 6 emre emre 4096 May 26 15:26 //
-rw-rw-r-- 1 emre emre 16705 May 19 12:17 200104004099_main.c
-rw-rw-r-- 1 emre emre 290 May 26 15:26 makefile
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ make
-----
Removing files...
-----
Compiling all files...
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ ll
total 60
drwxrwxrwx 2 emre emre 4096 May 26 15:28 //
drwxrwxrwx 6 emre emre 4096 May 26 15:26 //
-rw-rw-r-- 1 emre emre 16705 May 19 12:17 200104004099_main.c
-rw-rw-r-- 1 emre emre 290 May 26 15:26 makefile
-rwxrwxr-x 1 emre emre 28576 May 26 15:28 hwc*
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ ./hwc 10 4 ../testdir/src/libvterm/src ../toCopy
-----
STATISTICS-----
Consumers: 4 - Buffer buf_size: 10
Number of Regular File: 140
Number of FIFO File: 0
Number of Directory: 3
TOTAL BYTES COPIED: 24873082
TOTAL TIME: 00:00.058 (min:sec.mill)
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$
```

c) Third Test:

```
Activities Terminal May 26 15:29 emre@emre-GF63-Thin-105C: ~/Desktop/hw4_system/hw4test/put_your_codes_here

emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ ll
total 32
drwxrwxrwx 2 emre emre 4096 May 26 15:28 //
drwxrwxrwx 6 emre emre 4096 May 26 15:26 //
-rw-rw-r-- 1 emre emre 16705 May 19 12:17 200104004099_main.c
-rw-rw-r-- 1 emre emre 290 May 26 15:26 makefile
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ make
-----
Removing files...
-----
Compiling all files...
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ ll
total 60
drwxrwxrwx 2 emre emre 4096 May 26 15:29 //
drwxrwxrwx 6 emre emre 4096 May 26 15:26 //
-rw-rw-r-- 1 emre emre 16705 May 19 12:17 200104004099_main.c
-rw-rw-r-- 1 emre emre 290 May 26 15:26 makefile
-rwxrwxr-x 1 emre emre 28576 May 26 15:29 hwc*
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$ ./hwc 10 10 ../testdir ../toCopy
-----
STATISTICS-----
Consumers: 10 - Buffer buf_size: 10
Number of Regular File: 3116
Number of FIFO File: 1
Number of Directory: 152
TOTAL BYTES COPIED: 73520554
TOTAL TIME: 00:00.172 (min:sec.mill)
emre@emre-GF63-Thin-105C:~/Desktop/hw4_system/hw4test/put_your_codes_here$
```

