

# **CSE 344 System Programming**

## **Homework 5**

**Kahraman Arda KILIÇ**

**1901042648**

## 1-Producer & Consumer

This project was about the producer-consumer problem. For this reason, one producer and a number of consumer threads, determined by the user input, were created. The tasks of each are as follows:

Producer:

- Takes the source and destination folders as arguments. It opens these folders using the `opendir()` function. If the destination folder doesn't exist, it creates it using the `mkdir()` function.
- Checks if the folders are successfully opened, and returns NULL if any folder fails to open.
- Reads each file and folder in the source folder using the `readdir()` function. It retrieves the stats of each entry using the `stat()` function and stores them in the `fileStat` structure.
- Determines whether the read entry is a regular file or a directory based on its `stat` values. It performs the copying operation only on regular files. Files within a source directory are also recursively copied by calling the function again.
- If the entry is a regular file, it opens the source and destination file descriptors. If a file with the same name already exists in the destination folder, it is truncated. This operation is performed using `open(dest_filepath, O_WRONLY | O_CREAT | O_TRUNC)`. It checks for errors in file opening and returns NULL if any occur.
- If the source and destination files are successfully opened, it puts their file descriptors into a Task object. Then it puts this task into the `taskQueue` array. In this project, the buffer is represented by the `taskQueue`.
- Access to the `taskQueue` (buffer) is managed using two semaphores. One is `semEmpty` (checks if the buffer is empty), and the other is `semFull` (checks if the buffer is full). These two semaphores ensure synchronization between the producer and consumer for the `taskQueue` (buffer).
- If the producer thread has finished recursively processing all directories and there are no more files to read, it sets the "done" flag to 1 and notifies the consumers.
- At the end of the function, the opened folders are closed.

Consumer:

- Runs in an infinite while loop. After a task is added to the taskQueue (buffer) by the producer, a consumer thread from the thread pool is assigned to that task.
- While waiting for a new task, the consumer continuously checks the "done" flag.
- If the "done" flag is set to 1 and there is a task in the taskQueue (buffer), the consumer takes and completes the task, and then exits the while loop by returning NULL.
- If the "done" flag is set to 1 and there are no tasks in the taskQueue (buffer), the consumer directly exits the while loop by returning NULL.
- After retrieving a task from the taskQueue (buffer), the consumer reads the source file using the read() function. It then writes each byte value it reads to the destination file using the write() function.
- During the read and write operations, the read byte values are accumulated, and at the end of the program, they are printed as statistics.
- After each copying operation, a message is printed to the standard output indicating the number of bytes copied from which file to which file.
- All opened files (file descriptors opened by the producer) are closed by the consumer at the end.

## 2-Thread Pool

For the consumer threads, a thread pool implementation was required. Therefore, I created a pthread\* array called "consumers." I allocated memory for this array based on the value provided by the user. Then, using pthread\_create(), I made each consumer active within the main(). The consumer threads execute the consumerFunction() function. After each task is added by the producer, an idle thread without a task in the pool is awakened, and the given task is performed. The synchronization of the threads within the thread pool is achieved using "pthread\_cond\_t condQueue." Each consumer thread waits in a sleep state using pthread\_cond\_wait() as long as taskCount == 0 and done == 0. When a task is added to the taskQueue (buffer) by the producer, an asleep thread is awakened using pthread\_cond\_signal() to perform the task. If the program encounters any error or if the done flag is set to 1, all sleeping threads are awakened using pthread\_cond\_broadcast().

### 3-Handling Race Conditions

To prevent race conditions between threads, `pthread_mutex_t` was used. Two different mutexes were used to ensure synchronization for both the taskQueue (buffer) and the standard output. Before accessing the taskQueue, `lock()` is applied using `mutexQueue`, then necessary operations are performed (such as retrieving a task, decreasing/increasing the task count, deleting a task, adding a task, etc.), and `mutexQueue` is unlocked using `unlock()`. This process ensures synchronization among threads while accessing the taskQueue.

Additionally, both consumer and producer threads print to the standard output using `printf()`. To prevent potential race conditions that may arise during printing, `lock()` and `unlock()` operations were performed using `mutexSTDOUT`.

### 4- Buffer

In this project, the buffer is referred to as the taskQueue. It is an array of Task objects. Each Task object contains the source file descriptor, destination file descriptor, source filename, and destination filename. In the producer, when file information is read from the source folder, a new Task is created and added to the taskQueue. Then, consumers retrieve the first element from the taskQueue (FIFO) and perform the given task. The buffer operates based on the First In - First Out (FIFO) principle.

```
typedef struct Task{
    int source_fd;
    int dest_fd;
    char * source_filename;
    char * dest_filename;
}Task;
```

## 5-Signal Handling

In this project, only the SIGINT signal is handled. The SIGCHLD signal is not handled since there are no child processes involved. The handling of the SIGINT signal is as follows:

- The user sends the SIGINT signal to the program.
- The program receives the SIGINT signal and goes to the sigIntHandler() function.
- In this function, the "done" flag is first set to 1, and pthread\_cond\_broadcast() is used to wake up the sleeping threads in the thread pool and wait for them to shut themselves down.
- If a consumer thread is currently in the process of copying a file when the SIGINT signal is received, that thread continues its operation. After completing its task, it closes itself.

## 6- Experiments with different buffer size and consumer threads

Buffer Size = 5

Consumer Threads = 3

Copied files = 7

Total time for copying = 4.74 seconds

```
ardaklil@ardaklil:~/Desktop$ valgrind --leak-check=full --show-leak-kinds=all --track-origins=yes --verbose --log-file=levelgrind-outs.txt ./main 5 3 /home/ardaklil/Desktop/TestFolderMS/folderA/e-Belge.pdf to /home/ardaklil/Desktop/TestFolderMSDest/folderA/e-Belge.pdf
193918 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/folderA/e-Belge.pdf to /home/ardaklil/Desktop/TestFolderMSDest/folderA/e-Belge.pdf
182817 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/folderA/foto.jpeg to /home/ardaklil/Desktop/TestFolderMSDest/folderA/foto.jpeg
0 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/folderA/10mb.txt to /home/ardaklil/Desktop/TestFolderMSDest/folderA/10mb.txt
Joined to consumers[0] thread
10866395 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/folderA/10mb.txt to /home/ardaklil/Desktop/TestFolderMSDest/folderA/10mb.txt
310488389 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/nesenliyerinde.npt to /home/ardaklil/Desktop/TestFolderMSDest/nesenliyerinde.npt
102 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/testA.c to /home/ardaklil/Desktop/TestFolderMSDest/testA.c
Joined to consumers[1] thread
Joined to consumers[2] thread
Number of files copied:7
Total time for copying:4.74558 seconds
Total bytes copied:411526741
Total number of regular files copied:7
Total number of directories copied:1
```

Buffer Size = 5

Consumer Threads = 5

Copied files = 7

Total time for copying = 5.12 seconds

```
ardaklil@ardaklil:~/Desktop$ valgrind --leak-check=full --show-leak-kinds=all --track-origins=yes --verbose --log-file=levelgrind-outs.txt ./main 5 5 /home/ardaklil/Desktop/TestFolderMS/folderA/10mb.txt to /home/ardaklil/Desktop/TestFolderMSDest/folderA/10mb.txt
0 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/folderA/10mb.txt to /home/ardaklil/Desktop/TestFolderMSDest/folderA/10mb.txt
Joined to producer thread
102 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/testA.c to /home/ardaklil/Desktop/TestFolderMSDest/testA.c
102 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/testB to /home/ardaklil/Desktop/TestFolderMSDest/testB
182817 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/folderA/foto.jpeg to /home/ardaklil/Desktop/TestFolderMSDest/folderA/foto.jpeg
Joined to consumers[0] thread
Joined to consumers[1] thread
193918 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/folderA/e-Belge.pdf to /home/ardaklil/Desktop/TestFolderMSDest/folderA/e-Belge.pdf
10866395 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/folderA/10mb.txt to /home/ardaklil/Desktop/TestFolderMSDest/folderA/10mb.txt
310488389 bytes are copied from /home/ardaklil/Desktop/TestFolderMS/nesenliyerinde.npt to /home/ardaklil/Desktop/TestFolderMSDest/nesenliyerinde.npt
Joined to consumers[2] thread
Joined to consumers[3] thread
Joined to consumers[4] thread
Number of files copied:7
Total time for copying:5.12737 seconds
Total bytes copied:411527788
Total number of regular files copied:7
Total number of directories copied:1
```

Buffer Size = 3

Consumer Threads = 5

Copied files = 7

Total time for copying = 4.81 seconds

```
ardaklil@ardaklil:~$ valgrind --leak-check=full --show-leak-kinds=all --track-origins=yes --verbose --log-file=valgrind-out.txt ./main 3 5 /home/ardaklil/Desktop/TestFolderM5 /home/ardaklil/Desktop/TestFolderM5Dest
0 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/10mb.txt to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/10mb.txt
102 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/testA.c to /home/ardaklil/Desktop/TestFolderM5Dest/testA.c
Joined to producer thread
903 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/testB to /home/ardaklil/Desktop/TestFolderM5Dest/testB
182017 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/foto.jpeg to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/foto.jpeg
193018 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/e-Belge.pdf to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/e-Belge.pdf
100663295 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/10mb.txt to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/10mb.txt
Joined to consumers[0] thread
Joined to consumers[1] thread
Joined to consumers[2] thread
Joined to consumers[3] thread
Joined to consumers[4] thread
Number of files copied:7
Total time for copying:4.819119 seconds
Total bytes copied:41357783
Total number of regular files copied:7
Total number of directories copied:1
```

Buffer Size = 7

Consumer Threads = 7

Copied files = 7

Total time for copying = 4.68 seconds (Best case)

```
ardaklil@ardaklil:~$ valgrind --leak-check=full --show-leak-kinds=all --track-origins=yes --verbose --log-file=valgrind-out.txt ./main 7 7 /home/ardaklil/Desktop/TestFolderM5 /home/ardaklil/Desktop/TestFolderM5Dest
0 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/10mb.txt to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/10mb.txt
102 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/testA.c to /home/ardaklil/Desktop/TestFolderM5Dest/testA.c
902 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/testB to /home/ardaklil/Desktop/TestFolderM5Dest/testB
Joined to producer thread
193018 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/e-Belge.pdf to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/e-Belge.pdf
182017 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/foto.jpeg to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/foto.jpeg
Joined to consumers[0] thread
Joined to consumers[1] thread
Joined to consumers[2] thread
Joined to consumers[3] thread
Joined to consumers[4] thread
Joined to consumers[5] thread
Joined to consumers[6] thread
Number of files copied:7
Total time for copying:4.688779 seconds
Total bytes copied:41357783
Total number of regular files copied:7
Total number of directories copied:1
```

Buffer Size = 10

Consumer Threads = 10

Copied files = 7

Total time for copying = 4.91 seconds

```
ardaklil@ardaklil:~$ valgrind --leak-check=full --show-leak-kinds=all --track-origins=yes --verbose --log-file=valgrind-out.txt ./main 10 10 /home/ardaklil/Desktop/TestFolderM5 /home/ardaklil/Desktop/TestFolderM5Dest
0 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/10mb.txt to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/10mb.txt
102 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/testA.c to /home/ardaklil/Desktop/TestFolderM5Dest/testA.c
902 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/testB to /home/ardaklil/Desktop/TestFolderM5Dest/testB
Joined to producer thread
Joined to consumers[0] thread
182017 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/foto.jpeg to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/foto.jpeg
193018 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/e-Belge.pdf to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/e-Belge.pdf
100663295 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/FolderA/10mb.txt to /home/ardaklil/Desktop/TestFolderM5Dest/FolderA/10mb.txt
19048399 bytes are copied from /home/ardaklil/Desktop/TestFolderM5/Nesentzyerinde.mp4 to /home/ardaklil/Desktop/TestFolderM5Dest/Nesentzyerinde.mp4
Joined to consumers[1] thread
Joined to consumers[2] thread
Joined to consumers[3] thread
Joined to consumers[4] thread
Joined to consumers[5] thread
Joined to consumers[6] thread
Joined to consumers[7] thread
Joined to consumers[8] thread
Joined to consumers[9] thread
Number of files copied:7
Total time for copying:4.910990 seconds
Total bytes copied:41357783
Total number of regular files copied:7
Total number of directories copied:1
```

Buffer Size = 20

Consumer Threads = 20

Copied files = 7

Total time for copying = 5.92 seconds (Worst case)

```
ardak@ardak:~/Desktop$ valgrind --leak-check=full --show-leak-kinds=all --track-origins=yes --verbose --log-file=leavalgrind-out.txt ./main 20 20 /home/ardak/Desktop/TestFolderM5 /home/ardak/Desktop/TestFolderM5Dest
0 bytes are copied from /home/ardak/Desktop/TestFolderM5/FolderA/10mb.txt to /home/ardak/Desktop/TestFolderM5Dest/FolderA/10mb.txt
102 bytes are copied from /home/ardak/Desktop/TestFolderM5/testA.c to /home/ardak/Desktop/TestFolderM5Dest/testA.c
902 bytes are copied from /home/ardak/Desktop/TestFolderM5/testB to /home/ardak/Desktop/TestFolderM5Dest/testB
Joined to producer thread
182017 bytes are copied from /home/ardak/Desktop/TestFolderM5/FolderA/foto.jpeg to /home/ardak/Desktop/TestFolderM5Dest/FolderA/foto.jpeg
129218 bytes are copied from /home/ardak/Desktop/TestFolderM5/FolderA/e belge.pdf to /home/ardak/Desktop/TestFolderM5Dest/FolderA/e belge.pdf
Joined to consumers[0] thread
Joined to consumers[1] thread
Joined to consumers[2] thread
Joined to consumers[3] thread
Joined to consumers[4] thread
Joined to consumers[5] thread
Joined to consumers[6] thread
Joined to consumers[7] thread
Joined to consumers[8] thread
10062295 bytes are copied from /home/ardak/Desktop/TestFolderM5/FolderA/100mb.txt to /home/ardak/Desktop/TestFolderM5Dest/FolderA/100mb.txt
31848389 bytes are copied from /home/ardak/Desktop/TestFolderM5/HeseniYerinde.npt to /home/ardak/Desktop/TestFolderM5Dest/HeseniYerinde.npt
Joined to consumers[9] thread
Joined to consumers[10] thread
Joined to consumers[11] thread
Joined to consumers[12] thread
Joined to consumers[13] thread
Joined to consumers[14] thread
Joined to consumers[15] thread
Joined to consumers[16] thread
Joined to consumers[17] thread
Joined to consumers[18] thread
Joined to consumers[19] thread
Number of files copied:7
Total time for copying:5.92595 seconds
Total bytes copied:411527703
Total number of regular files copied:7
Total number of directories copied:1
```

## 7-Per-process limit on the number of open file descriptors

Each process has a limited number of file opening rights at the same time. If this number is reached, a new file cannot be opened without closing an already open file. If the `open()` function tries to open a file and the file descriptor limit has been reached, the file cannot be opened and the `errno` value is set to `EMFILE`. By using this `errno` value, it can be determined within the program whether there is an error due to reaching the limit.

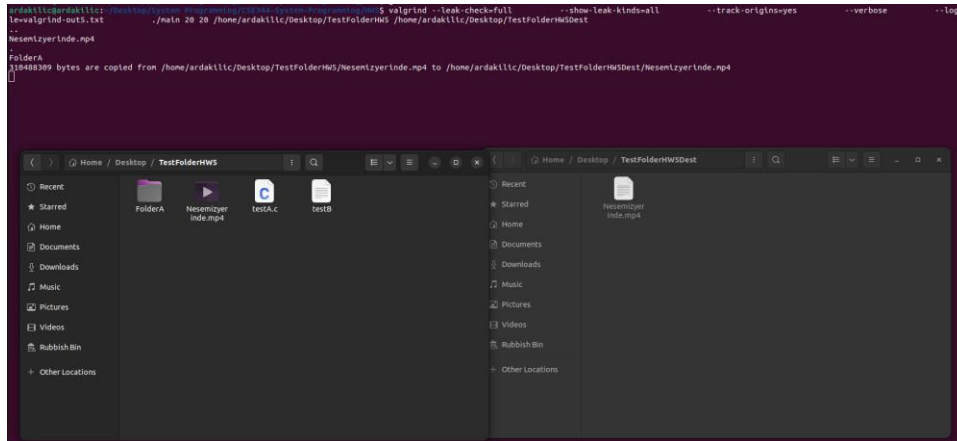
```
source_fd = open(source_filepath, O_RDONLY, 0666);
dest_fd = open(dest_filepath, O_WRONLY | O_CREAT | O_TRUNC, 0666);

/* If there is an error at opening file, then print it. */
if(source_fd == -1 || dest_fd == -1){
    if(errno == EMFILE){
        perror("Exceeded the per-process limit on open file descriptors.\n");
    }
    else{
        pthread_mutex_lock(&mutexSTDOUT);
        printf("Error at opening file\n");
        pthread_mutex_unlock(&mutexSTDOUT);
    }
    pthread_mutex_lock(&mutexQueue);
    done = 1; // Set done flag
    pthread_mutex_unlock(&mutexQueue);
    pthread_cond_broadcast(&condQueue);
    closedir(SOURCE_DIR);
    closedir(DEST_DIR);
    return NULL;
}
```

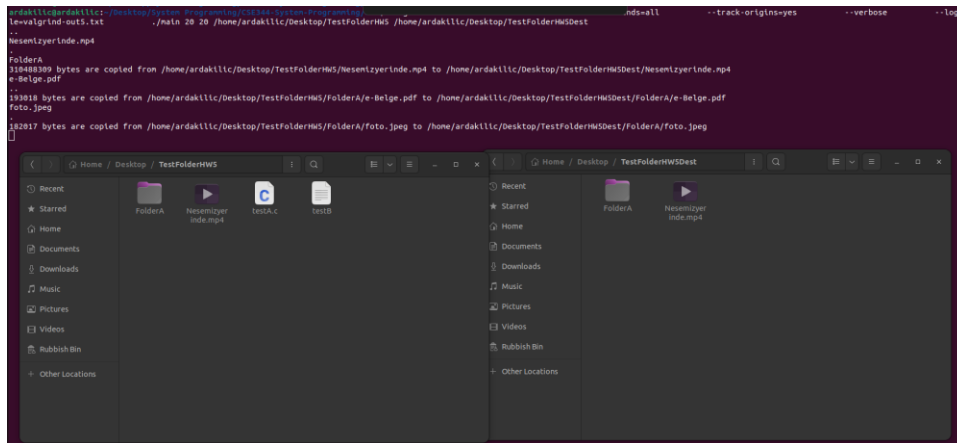
## 8- Tests

The program is started. Copy process will be from /Desktop/TestFolderHW5 to /Desktop/TestFolderHW5Dest folder. (Sleep operation is used only for taking screenshots before program termination. It is removed afterwards.)

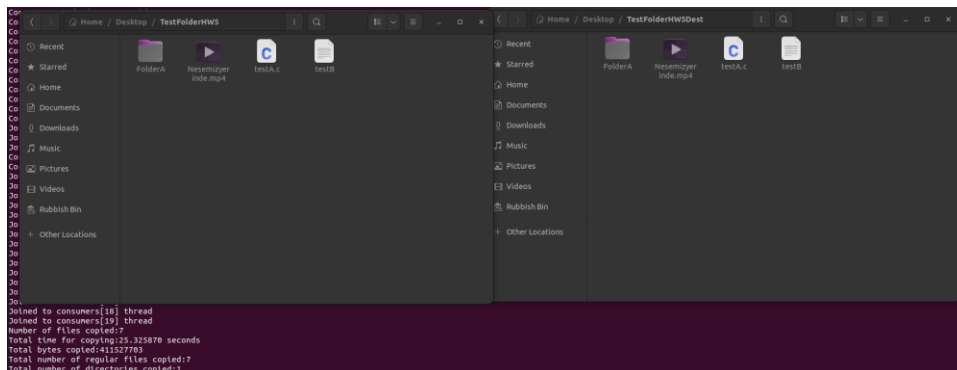
The first file is being written to the destination file.



The directory is created in the destination folder.



The program is finished. Both folders have 411 MB at the end of the program.





The SIGINT signal is sent to the program during its execution. Only one file is written to the destination folder. During this test, sleep() is used to prevent producer thread to set done flag before SIGINT handler.

```
ardakilc@ardakilc:~$ valgrind --leak-check=full --track-origins=yes --verbose --show-leak-kinds=all ./main 20 20 /home/ardakilc/Desktop/TestFolderHWS /home/ardakilc/Desktop/TestFolderHWSDest\nesentzyerInde.mp4\n\nFolderHWS\n^C10488389 bytes copied from /home/ardakilc/Desktop/TestFolderHWS/nisentzyerInde.mp4 to /home/ardakilc/Desktop/TestFolderHWSDest/nisentzyerInde.mp4\njoined to producer thread\njoined to consumers[0] thread\njoined to consumers[1] thread\njoined to consumers[2] thread\njoined to consumers[3] thread\njoined to consumers[4] thread\njoined to consumers[5] thread\njoined to consumers[6] thread\njoined to consumers[7] thread
```