CSE 344 – HW5 Report

Mehmet Acar

How I solved this problem?

This code is a multi-threaded file copying program written in C. It uses a producer-consumer model to efficiently copy files from a source directory to a destination directory. The program takes command-line arguments to specify the buffer size, the number of consumer threads, the source directory, and the destination directory.

Let's go through the code and understand its functionality:

1. The necessary header files are included: `stdio.h`, `stdlib.h`, `pthread.h`, `dirent.h`, `string.h`, `unistd.h`, `fcntl.h`, `sys/types.h`, `sys/stat.h`, `sys/time.h`, and `sys/wait.h`.
2. The code defines a constant `MAX_PATH_LENGTH` with a value of 1024. This represents the maximum length of the file path.
3. A structure `FileInfo` is defined to hold information about each file. It contains the file descriptors for the source and destination files, as well as the file paths.
4. Global variables are declared:
    - `buffer` is a pointer to an array of `FileInfo` structures, used as the buffer for the producer-consumer model.
    - `buffer_size` stores the size of the buffer.
    - `buffer_count` keeps track of the number of items in the buffer.
    - `buffer_in` and `buffer_out` are indices for inserting and removing items from the buffer.
    - `done` is a flag that indicates when the producer has finished.
    - `total_bytes_copied` keeps track of the total number of bytes copied.
    - `num_files_copied` stores the total number of regular files copied.
    - `num_directories_copied` stores the total number of directories copied.
    - `num_fifos_copied` stores the total number of fifos copied.
    - `buffer_mutex` is a mutex used to control access to the buffer.
    - `output_mutex` is a mutex used to control access to the output.

- `buffer_not_empty` is a condition variable used to signal when the buffer is not empty.
- `buffer_not_full` is a condition variable used to signal when the buffer is not full.

5. The code defines a signal handler function `handle_signal`, which is called when the program receives a signal (SIGINT, SIGTSTP, or SIGQUIT). In this function, the necessary clean-up tasks are performed, including freeing memory, destroying mutexes and condition variables, and exiting the program.
6. The function `copy_file` is responsible for copying the contents of a file from the source to the destination. It reads data from the source file descriptor and writes it to the destination file descriptor. The number of bytes copied is tracked, and the file descriptors are closed after copying.
7. The function `add_buffer` is a recursive function that traverses the source directory and adds files and directories to the buffer for copying. It uses the `readdir` function to iterate over the entries in the directory. For each entry, it creates the source and destination file paths and checks if the entry is a directory or a regular file/FIFO. If it's a directory, the function calls itself recursively. If it's a regular file/FIFO, it opens the source and destination files, creates the `FileInfo` structure, and adds it to the buffer.
8. The producer thread function `producer` is responsible for calling `add_buffer` with the source and destination directories. It recursively adds files and directories to the buffer. After adding all the files, it sets the `done` flag and signals that the buffer is not empty.
9. The consumer thread function `consumer` runs in an infinite loop and continuously consumes items from the buffer. It waits until the buffer is not empty, retrieves a `FileInfo` structure from the buffer, and calls the `copy_file` function to copy the file. After copying, it prints the completion status and
10. The `main` function is the entry point of the program. It first checks the command-line arguments to ensure that the required arguments are provided: buffer size, number of consumer threads, source directory, and destination directory.
11. It initializes the buffer size and the number of consumer threads based on the command-line arguments. It also initializes the buffer, mutexes, and condition variables.

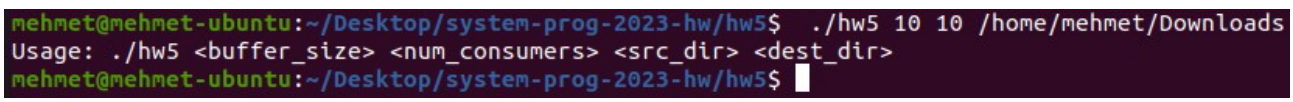
12. The program sets up signal handlers for SIGINT, SIGTSTP, and SIGQUIT to handle termination signals gracefully.
13. It creates a producer thread and passes the source and destination directories as arguments.
14. It creates multiple consumer threads based on the specified number. Each consumer thread will consume files from the buffer and copy them.
15. The main thread waits for the producer and consumer threads to complete using `pthread_join`.
16. After all threads have completed, it calculates the total time taken by measuring the start and end time using `gettimeofday`.
17. It displays the total time taken, the number of files copied, and the total number of bytes copied.
18. Finally, the program cleans up by destroying mutexes and condition variables, freeing the buffer memory, and returning 0.

Overall, this program demonstrates a multithreaded approach to efficiently copy files using a producer-consumer model. The producer thread recursively traverses the source directory and adds files to the buffer, while the consumer threads consume files from the buffer and copy them. The program ensures synchronization and coordination between the producer and consumer threads using mutexes and condition variables.

Note: It's important to handle errors properly in a production-ready code. This code includes some basic error handling, such as checking the return values of system calls and printing error messages when necessary. However, there might be room for improvement and additional error handling based on specific requirements and use cases.

Test Cases and Results

1- If user enters wrong command line



2- If user enters wrong buffer size

```
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$ ./hw5 -5 10 /home/mehmet/Downloads /home/mehmet/Documents/fileCopy
Buffer size must be a positive integer
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$
```

3- If user enters wrong consumer number

```
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$ ./hw5 5 -10 /home/mehmet/Downloads /home/mehmet/Documents/fileCopy
Number of consumers must be a positive integer
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$
```

4- Copy file from source path to the destination path

```
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$ valgrind --leak-check=full ./hw5 10 10 /home/mehmet/Documents/test /home/mehmet/Documents/fileCopy
==48197== Memcheck, a memory error detector
==48197== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==48197== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==48197== Command: ./hw5 10 10 /home/mehmet/Documents/test /home/mehmet/Documents/fileCopy
==48197==
Copied file: /home/mehmet/Documents/test/output
Completed: /home/mehmet/Documents/fileCopy/output
Copied file: /home/mehmet/Documents/test/folder1/output
Completed: /home/mehmet/Documents/fileCopy/folder1/output
Copied file: /home/mehmet/Documents/test/folder1/output (3rd copy)
Completed: /home/mehmet/Documents/fileCopy/folder1/output (3rd copy)
Copied file: /home/mehmet/Documents/test/folder1/output (4th copy)
Completed: /home/mehmet/Documents/fileCopy/folder1/output (4th copy)
Copied file: /home/mehmet/Documents/test/output (9th copy)
Completed: /home/mehmet/Documents/fileCopy/output (9th copy)
Copied file: /home/mehmet/Documents/test/output (8th copy)
Completed: /home/mehmet/Documents/fileCopy/output (8th copy)
Copied file: /home/mehmet/Documents/test/output (copy)
Completed: /home/mehmet/Documents/fileCopy/output (copy)
Copied file: /home/mehmet/Documents/test/output (another copy)
Completed: /home/mehmet/Documents/fileCopy/output (another copy)
Copied file: /home/mehmet/Documents/test/output (3rd copy)
Completed: /home/mehmet/Documents/fileCopy/output (3rd copy)
Copied file: /home/mehmet/Documents/test/fifo2
Completed: /home/mehmet/Documents/fileCopy/fifo2
Copied file: /home/mehmet/Documents/test/output (6th copy)
Completed: /home/mehmet/Documents/fileCopy/output (6th copy)
Copied file: /home/mehmet/Documents/test/output (7th copy)
Completed: /home/mehmet/Documents/fileCopy/output (7th copy)
Copied file: /home/mehmet/Documents/test/fifo1
Completed: /home/mehmet/Documents/fileCopy/fifo1
Copied file: /home/mehmet/Documents/test/fifo3
Completed: /home/mehmet/Documents/fileCopy/fifo3
Copied file: /home/mehmet/Documents/test/output (4th copy)
Completed: /home/mehmet/Documents/fileCopy/output (4th copy)
Copied file: /home/mehmet/Documents/test/output (5th copy)
Completed: /home/mehmet/Documents/fileCopy/output (5th copy)

Total time taken: 31.23 seconds
Number of files copied: 13
Number of directories copied: 1
Number of fifos copied: 3
Total bytes copied: 1299912536
==48197==
==48197== HEAP SUMMARY:
==48197==     in use at exit: 1,654 bytes in 4 blocks
==48197==   total heap usage: 21 allocs, 17 frees, 91,998 bytes allocated
==48197==
==48197== LEAK SUMMARY:
==48197==    definitely lost: 0 bytes in 0 blocks
==48197==    indirectly lost: 0 bytes in 0 blocks
==48197==      possibly lost: 0 bytes in 0 blocks
==48197==    still reachable: 1,654 bytes in 4 blocks
```

```
==48197==          suppressed: 0 bytes in 0 blocks
==48197== Reachable blocks (those to which a pointer was found) are not shown.
==48197== To see them, rerun with: --leak-check=full --show-leak-kinds=all
==48197==
==48197== For lists of detected and suppressed errors, rerun with: -s
==48197== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$
```

It does not give memory leak error.

5- If user enters “Ctrl C” at the middle of the program

```
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$ valgrind --leak-check=full ./hw5 10 10 /home/mehmet/Downloads /home/mehmet/Documents/fileCopy
==53994== Memcheck, a memory error detector
==53994== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==53994== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==53994== Command: ./hw5 10 10 /home/mehmet/Downloads /home/mehmet/Documents/fileCopy
==53994==
Copied file: /home/mehmet/Downloads/fifo_segnum.h
Completed: /home/mehmet/Documents/fileCopy/fifo_segnum.h
Copied file: /home/mehmet/Downloads/Mehmet_Acar_1801042095.zip
Completed: /home/mehmet/Documents/fileCopy/Mehmet_Acar_1801042095.zip
Copied file: /home/mehmet/Downloads/Hw4_test_files.zip
Completed: /home/mehmet/Documents/fileCopy/Hw4_test_files.zip
Copied file: /home/mehmet/Downloads/1801042095 (2).zip
Completed: /home/mehmet/Documents/fileCopy/1801042095 (2).zip
Copied file: /home/mehmet/Downloads/output(1)
Completed: /home/mehmet/Documents/fileCopy/output(1)
Copied file: /home/mehmet/Downloads/wyoos-49a2efa4bf490ce6152deb82071fce30a5d76aca.zip
Completed: /home/mehmet/Documents/fileCopy/wyoos-49a2efa4bf490ce6152deb82071fce30a5d76aca.zip
Copied file: /home/mehmet/Downloads/hw3 5.pdf
Completed: /home/mehmet/Documents/fileCopy/hw3 5.pdf
Copied file: /home/mehmet/Downloads/final_project.pdf
Completed: /home/mehmet/Documents/fileCopy/final_project.pdf
^CCopied file: /home/mehmet/Downloads/output2
Completed: /home/mehmet/Documents/fileCopy/output2

Received SIGINT signal
==53994==
==53994== HEAP SUMMARY:
==53994==     in use at exit: 0 bytes in 0 blocks
==53994==   total heap usage: 20 allocs, 20 frees, 59,182 bytes allocated
==53994==
==53994== All heap blocks were freed -- no leaks are possible
==53994==
==53994== For lists of detected and suppressed errors, rerun with: -s
==53994== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$
```

It does not give memory leak error.

Buffer Size – Consumer Number Combination

According to the test results, when I copied large directories, only increasing the buffer size did not decrease the program’s total time clearly. But if I also increase consumer number, program’s total time is decreasing clearly.

```
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$ ./hw5 10 10 /home/mehmet/Downloads /home/mehmet/Documents/fileCopy
```

```
Total time taken: 12.65 seconds
Number of files copied: 135
Number of directories copied: 2
Number of fifos copied: 1
Total bytes copied: 609944117
```

```
mehmet@mehmet-ubuntu:~/Desktop/system-prog-2023-hw/hw5$ ./hw5 1000 1000 /home/mehmet/Downloads /home/mehmet/Documents/fileCopy
```

```
Total time taken: 9.13 seconds
Number of files copied: 134
Number of directories copied: 2
Number of fifos copied: 1
Total bytes copied: 553029589
```