Osama Tanveer
21801147
CS 353 - Project 1

Experiment 1

Fig. 1. N = 2048 constant.

For the first experiment, the number of bytes read/write in system call is set constant, as 2048. The number of characters produced and consumed by the producer and consumer programs are varied as multiples of 10, starting from 1 000 upto 1 000 000. In both modes, it can be seen that the execution time increases. From the trend, it is plausible to say that the trend is linear, but this can be further clarified by taking more values of M, in order to produce more accurate results. There is a difference in execution time of normal and tapped mode, as indicated by Fig. 1. This might be the overhead of context switching. In tapped mode, after the execution of the first child, the parent process has to read M bytes and write M bytes via system calls. This consumes clock cycles. Moreover, the main process might be scheduled by the CPU to execute later; the second child, executing concurrently, also experiences the same scheduling by the CPU. The parent process has to read from one pipe and write to another, which might take multiple clock cycles. The second child process, even though, takes the input directly from the pipe. The total time required for the parent process to read from one pipe and write to the other pipe is a bottleneck to the overall time required to read the whole input for the child process. For the normal mode, the output of the first child is directly fed to the second child through one pipe. The parent does not have the responsibility to read from one pipe and write to the other pipe.

Experiment 2

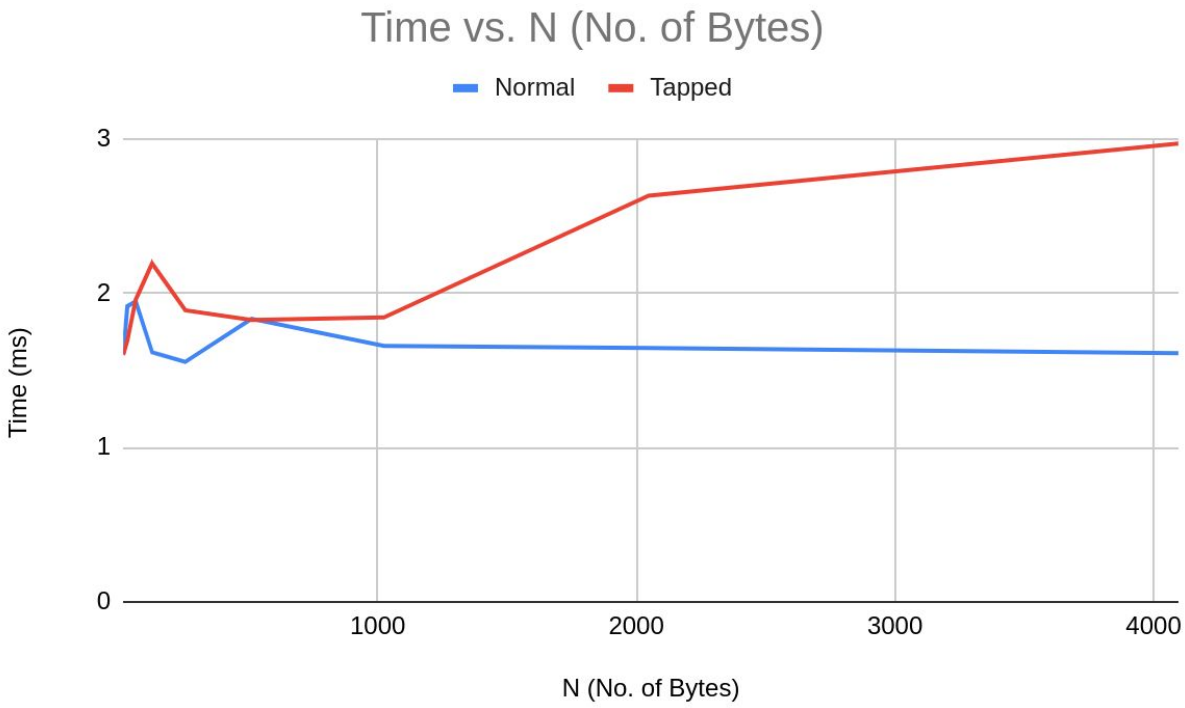

Fig. 2. M = 1000 constant.

For the second experiment, the value of the number of characters produced and consumed by producer and consumer programs respectively is kept constant. The number of bytes to read is varied. For the normal mode, the trend is constant. This is because the number of bytes read and written are independent of the value of bytes. This means that the output of the first child's execution is fed directly to the second child through proper I/O redirection. The initial outlier points might be the result of program-independent factors, such as other system processes being executed by the CPU. The general trend, however, is constant. In the case of tapped mode, the execution time is greater than the normal mode. The increased execution time can be explained by multiple factors. Firstly, the parent process reads from the pipe N bytes a time. However, if the number of remaining bytes is less than N, then a number less than N bytes are read, but the same system call (read) is executed to retrieve these bytes. Secondly, as mentioned earlier, the parent process copies the data from one pipe to another. Due to context switching between multiple processes, the parent and child processes may have to wait in queues multiple times during their execution. Since the processes are executing concurrently, context switching causes a time overhead. Lastly, as mentioned before, copying from one pipe to another might also cause a time overhead.

Conclusion

Drawing from Fig. 1 and Fig. 2, it is plausible to conclude that the normal mode execution is better in terms of execution time constraints. This is because this mode does not involve the additional overhead of the parent process reading and writing from one pipe to another. Dedicating one pipe for each child in this case causes increased execution times due to the reasons provided above.