CS342 – Operating Systems

Spring – 2021

Project 1

Mustafa Yaşar

21702808

Section: 1

Sample commands that are used to test the normal mode and their results.

make

./isp 1000 1

$isp ls

Output: consumer consumer.c isp isp.c Makefile producer producer.c

Total time = 0.002286 seconds

$isp cp isp.c ispCopy.c

Output: Total time = 0.002752 seconds

$isp ls -l | sort

Output:

-rw-rw-r-- 1 mustafa mustafa 12012 Şub 23 22:33 isp.c

-rw-rw-r-- 1 mustafa mustafa 12012 Şub 23 23:19 ispCopy.c

-rw-rw-r-- 1 mustafa mustafa 163 Şub 23 21:44 consumer.c

-rw-rw-r-- 1 mustafa mustafa 230 Şub 14 10:47 Makefile

-rw-rw-r-- 1 mustafa mustafa 436 Şub 23 21:25 producer.c

-rwxrwxr-x 1 mustafa mustafa 19416 Şub 23 23:18 consumer

-rwxrwxr-x 1 mustafa mustafa 19832 Şub 23 23:18 producer

-rwxrwxr-x 1 mustafa mustafa 28152 Şub 23 23:18 isp

total 104

Total time = 0.007162 seconds

$ ps aux

Output: { Since the output is very long, I only add the total time }

Total time = 0.020728 seconds

$ ps aux | sort

Output: { Since the output is very long, I only add the total time }

Total time = 0.028575

Sample commands that are used to test the tapped mode and their results.

./isp 1000 2

Isp$ ls

Output:

consumer consumer.c isp isp.c ispCopy.c Makefile producer producer.c

Total time = 0.002492 seconds

$isp cp isp.c ispCopy2.c

Output:

Total time = 0.002465 seconds

$isp ls -l | sort

Output:

-rw-rw-r-- 1 mustafa mustafa 12012 Şub 23 22:33 isp.c

-rw-rw-r-- 1 mustafa mustafa 12012 Şub 23 23:19 ispCopy.c

-rw-rw-r-- 1 mustafa mustafa 12012 Şub 23 23:23 ispCopy2.c

-rw-rw-r-- 1 mustafa mustafa 163 Şub 23 21:44 consumer.c

-rw-rw-r-- 1 mustafa mustafa 230 Şub 14 10:47 Makefile

-rw-rw-r-- 1 mustafa mustafa 436 Şub 23 21:25 producer.c

-rwxrwxr-x 1 mustafa mustafa 19416 Şub 23 23:18 consumer

-rwxrwxr-x 1 mustafa mustafa 19832 Şub 23 23:18 producer

-rwxrwxr-x 1 mustafa mustafa 28152 Şub 23 23:18 isp

total 116

character-count: 1000

read-call-count: 1

write-call-count: 1

Total time = 0.004550 seconds

$isp ps aux

Output:

Total time = 0.018373

$isp ps aux |sort

Output:

character-count: 33000

read-call-count: 33

write-call-count: 33

Total time = 0.029765 seconds

Experiments

Every Y coordinate in the line graphs are elapsed time while the program is run in seconds x10^(-3).

1st case, when the number of bytes to read/write in one system call is 256.

Mode 1:

M = 5000 -> Total time = 0.001931 sec

M = 10000 -> Total time = 0.002252 sec

M = 20000 -> Total time = 0.002520 sec

M = 40000 -> Total time = 0.002868 sec

M = 80000 -> Total time = 0.003765 sec

Mode 2:

M = 5000 -> Total time = 0.003489 sec

M = 10000 -> Total time = 0.003586 sec

M = 20000 -> Total time = 0.003983 sec

M = 40000 -> Total time = 0.004123 sec

M = 80000 -> Total time = 0.004504 sec

We can see that Mode 1 is significantly faster compared to Mode 2.

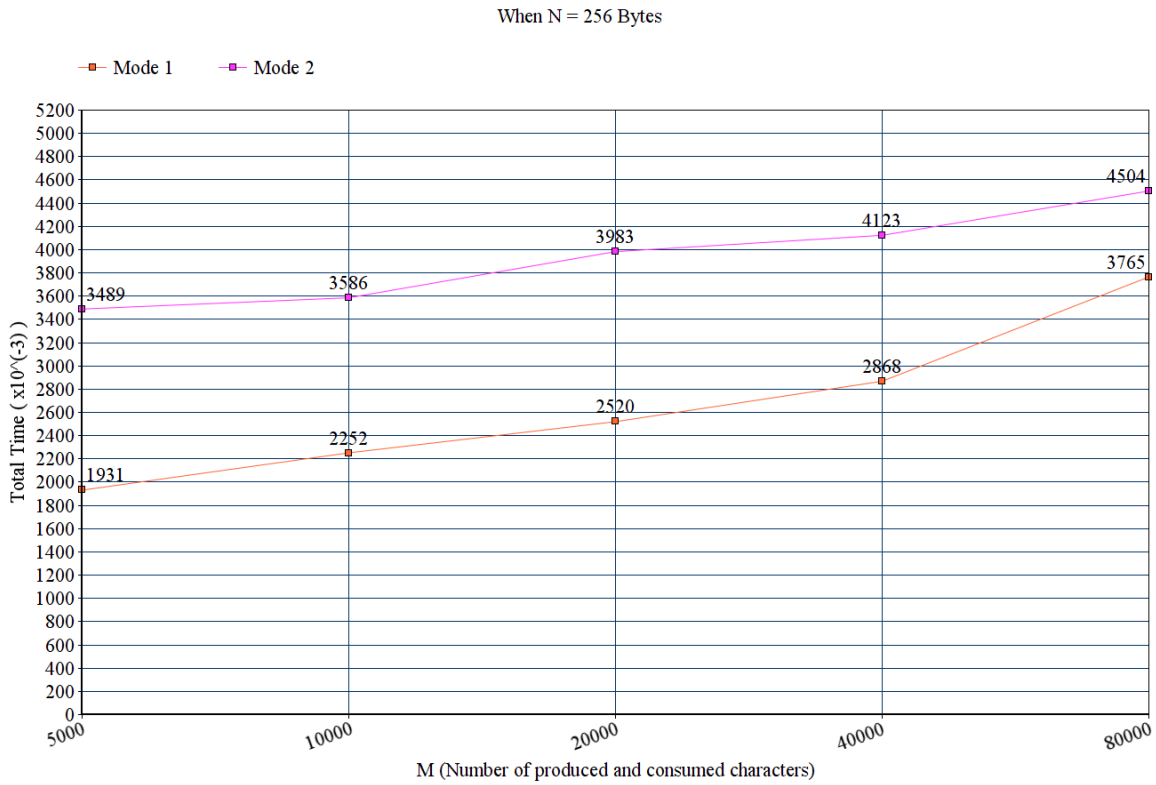

Figure 1 - Data Plot of 1st CASE

2nd case, when the number of bytes to read/write in one system call is 1024.

Mode 1:

M = 5000 -> Total time = 0.002027 sec

M = 10000 -> Total time = 0.002344 sec

M = 20000 -> Total time = 0.002487 sec

M = 40000 -> Total time = 0.002822 sec

M = 80000 -> Total time = 0.003575 sec

Mode 2:

M = 5000 -> Total time = 0.003452 sec

M = 10000 -> Total time = 0.003705 sec

M = 20000 -> Total time = 0.003790 sec

M = 40000 -> Total time = 0.004469 sec

M = 80000 -> Total time = 0.005185 sec

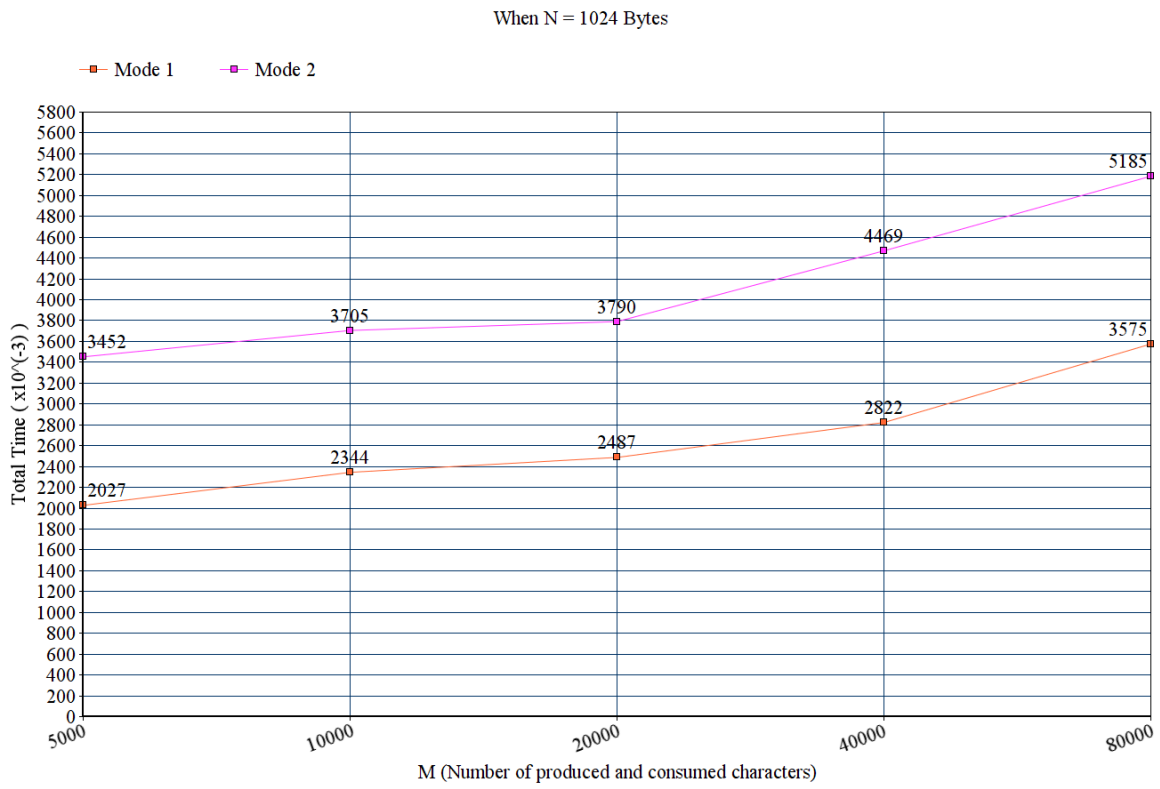

Figure 2 - Data Plot of 2nd CASE

3rd case, when the number of bytes to read/write in one system call is 2048.

Mode 1:

M = 5000 -> Total time = 0.001965 sec

M = 10000 -> Total time = 0.002268 sec

M = 20000 -> Total time = 0.002379 sec

M = 40000 -> Total time = 0.002702 sec

M = 80000 -> Total time = 0.003388 sec

Mode 2:

M = 5000 -> Total time = 0.003452 sec

M = 10000 -> Total time = 0.003499 sec

M = 20000 -> Total time = 0.003798 sec

M = 40000 -> Total time = 0.004540 sec

M = 80000 -> Total time = 0.004867 sec

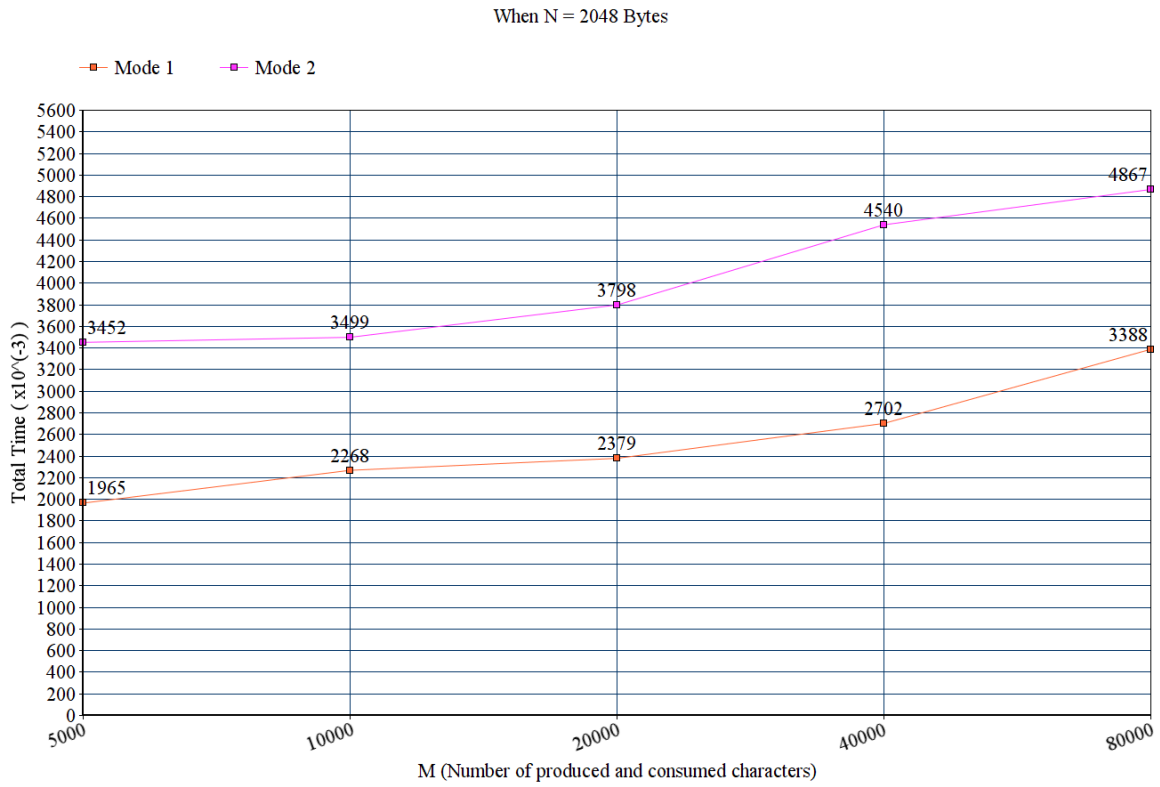

Figure 3 - Data Plot of 3rd CASE

4th case, we will adjust the number of bytes to read/write to 2 so that we can compare with the first 3 cases.

Mode2:

M = 5000 -> Total time = 0.005005 sec

M = 10000 -> Total time = 0.005673 sec

M = 20000 -> Total time = 0.007609 sec

M = 40000 -> Total time = 0.012410 sec

M = 80000 -> Total time = 0.018256 sec

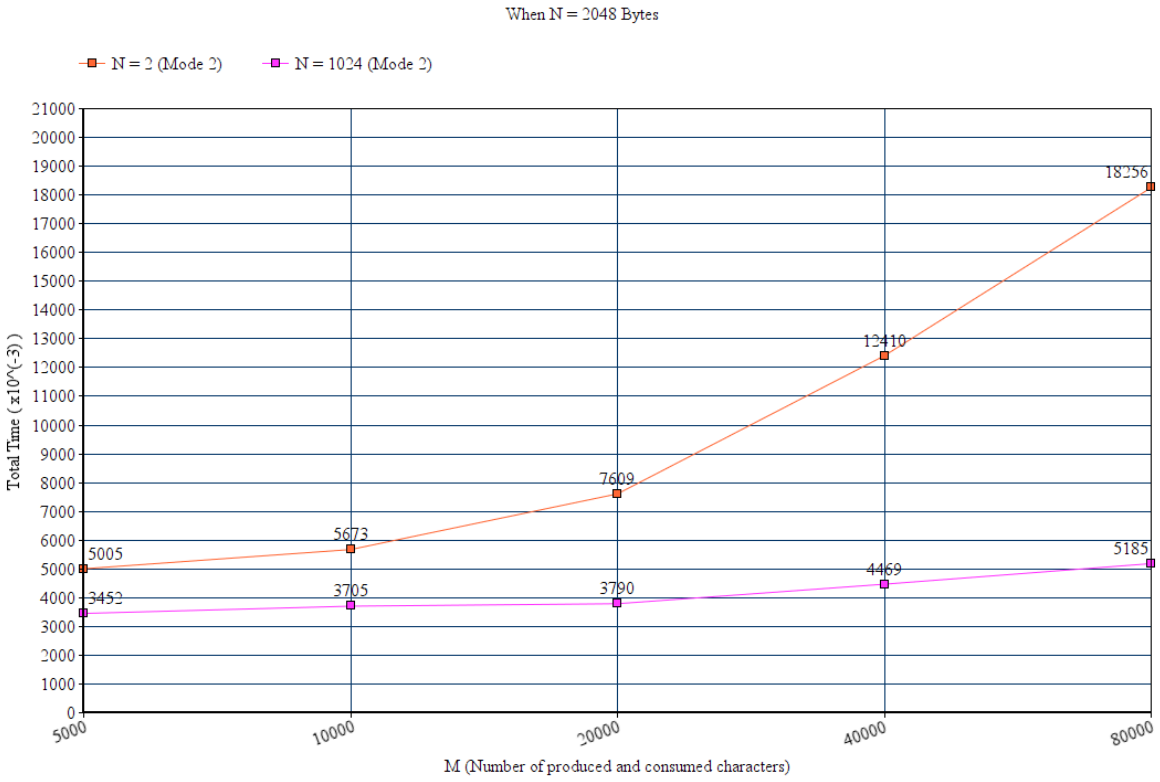

Figure 4 Data Plot of 4th CASE

Conclusion

In Normal Mode, the main process does not intercept, therefore, it is not on the data flow. Since we are not reading and writing byte by byte in the normal mode, changes in the number of bytes to read/write in one system call do not affect the result of the executions, as can be seen from the figures 1-2-3. However, in Tapped Mode, the main process in on the data flow. I.e., main process reads the output of the first child byte by byte from the pipe and write the result to the other pipe again byte by byte. When we increase the number of bytes that we read/write in one system call, the time elapsing for producing and consuming characters significantly decreases. Therefore, it can be beneficial in terms of performance to increase the number of bytes that is used in one system call.