

75.59-Técnicas de programación concurrente I

Trabajo Práctico n. 1:



Alumno	Padrón
Anca, Jorge	82399

Facultad de Ingeniería - Universidad de Buenos Aires

Objetivo	3
Introducción	3
Pipes	3
Unnamed Pipes	3
FIFO o Named Pipes	3
Memorias Compartida	3
Compilación y ejecución	4
Ejercicio 1	4
Explicación del algoritmo.	5
Visualizaciones	5
Problemas del código:	7
Ejercicio 2	7
Explicación del algoritmo	7
Visualizaciones	7
Problemas del código:	8
Dificultades encontradas	9
Conclusiones	9

Objetivo

El objetivo del tp es practicar los métodos de comunicación entre procesos de Memoria Compartida y FIFO.

Introducción

A continuación haré una breve descripción de conceptos teóricos involucrados en el presente tp.

Pipes

Unnamed Pipes

Los Pipes son archivos especiales que pueden almacenar una limitada cantidad de datos de la forma "first in first out". En Linux este tamaño en bytes está definido por la constante PIPE_SIZE que usualmente es igual a PAGE_SIZE. El buffer está definido por la constante PIPE_BUF y establece el número de bytes que pueden ser escritos en el Pipe de forma atómica (dicho valor es 4096).

El sistema operativo provee la sincronización entre los procesos que escriben y que leen. Por default si un proceso quiere escribir en un Pipe que este lleno, el sistema lo bloquea hasta que el Pipe pueda recibir información. Asimismo si un proceso intenta leer datos de un Pipe vacío, el sistema operativo lo bloquea hasta que haya datos en el mismo. Lo mismo ocurre si un proceso abre un Pipe para lectura y ninguno para escritura.

Los Pipes en general no son full duplex salvo en algunas versiones de Linux. Por lo tanto cuando se crea un Pipe con la syscall pipe esta devuelve un par de file descriptors, uno para la lectura y otro para la escritura.

FIFO o Named Pipes

Los FIFO son pipes que tiene una entrada de directorio por lo que se le puede asignar permisos de acceso a los procesos y se permite que proceso no relacionados se comuniquen mediante el pipe. A pesar de que los FIFO tienen una entrada de directorio, los datos escritos en los mismos son pasador y almacenados por el kernel y no directamente escritos en el sistema de archivos.

Memorias Compartida

La memoria compartida, permite a varios procesos compartir una dirección virtual de memoria. Este método es sumamente rápido pero no provee sistemas de sincronización por lo que se deben implementar.

El proceso es el siguiente:

1. Un proceso crea o aloca un segmento de memoria compartida con cierto tamaño y permisos de acceso
2. El proceso luego mapea el segmento de memoria a su espacio de direcciones
3. Otros procesos si son permitidos pueden acceder a la memoria compartida y mapear la misma a un espacio de direcciones. Para cada proceso la memoria compartida es como cualquier otro segmento de su direcciones de memoria
4. Cuada cada proceso termina de usar la memoria compartida puede hacer hacer un detach
5. El proceso creador es responsable de eliminar la memoria compartida cuando esta no es más requerida

Compilación y ejecución

Se adjunto un archivo Makefile que compila el código y genera los dos ejecutables:

-make all

Ejecuciones:

- ./procesadorSharedMem <tam_imagen> <numero_canaras>
- ./procesadorFifo <tam_imagen> <numero_canaras>

Ejercicio 1

Para resolver el problema dividi el código en tres clases:

- Imagen: genera la imagen a partir del ancho y del alto de la imagen, permite la serialización y deserialización de una imagen y el procesamiento.
Cada imagen está simulada por una matriz de enteros entre 0 y 255. la carga de la imagen se simula generando una matriz aleatoria.
El procesamiento se simula mediante la asignación de otros valores a la matriz de píxeles y un sleep de un número variables de segundos.
La serialización de las imágenes implica generar un string a partir de el ancho, alto y valores de las matriz de píxeles separados por coma.
Por último ña deserialiazación , recibe un string de valores separados por coma y reconstruye la imagen.
- ProcesadorImágenes: encargado de aplanar las imágenes
El aplanado se logra promediando los pixeles de las imágenes recibidas como parámetro
- Logger: encargado de crear el archivo de log y escribir en el mismo eventos

Para activar la escritura en el archivo log se debe cambiar la variable del main a:

```
bool logToFile=true;
```

y luego recopilar el código.

Explicación del algoritmo.

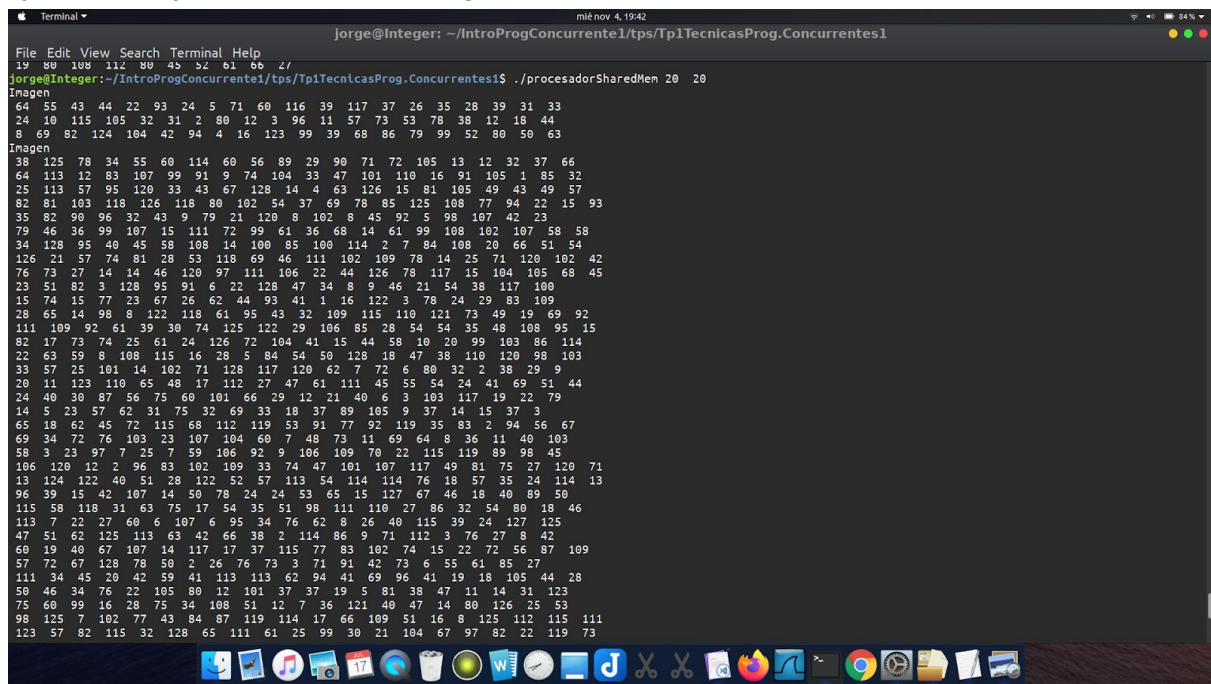
El proceso principal, crea el espacio de memoria compartido y genera hijos según la cantidad de cámaras. Cada hijo genera una imagen aleatoria simulando la carga de una imagen. Luego se la procesa y espera que el proceso padre setee la variable status como Ready para luego setearla como NotReady y escribir en el segmento de memoria compartido con la imagen serializada. Luego el proceso, desmapea el segmento de memoria compartido y termina su ejecución.

El proceso padre, espera la finalización de los procesos hijos con la syscall wait. cada vez termina un proceso, lee la memoria compartida la imagen serializada y setea la variable status en Ready para permitir que otro proceso hijo escriba en la memoria compartida. El padre luego deserializa la imagen y guarda la imagen en un vector.

Cuando terminan todos los procesos hijos, el padre realiza el aplanado de las imágenes-

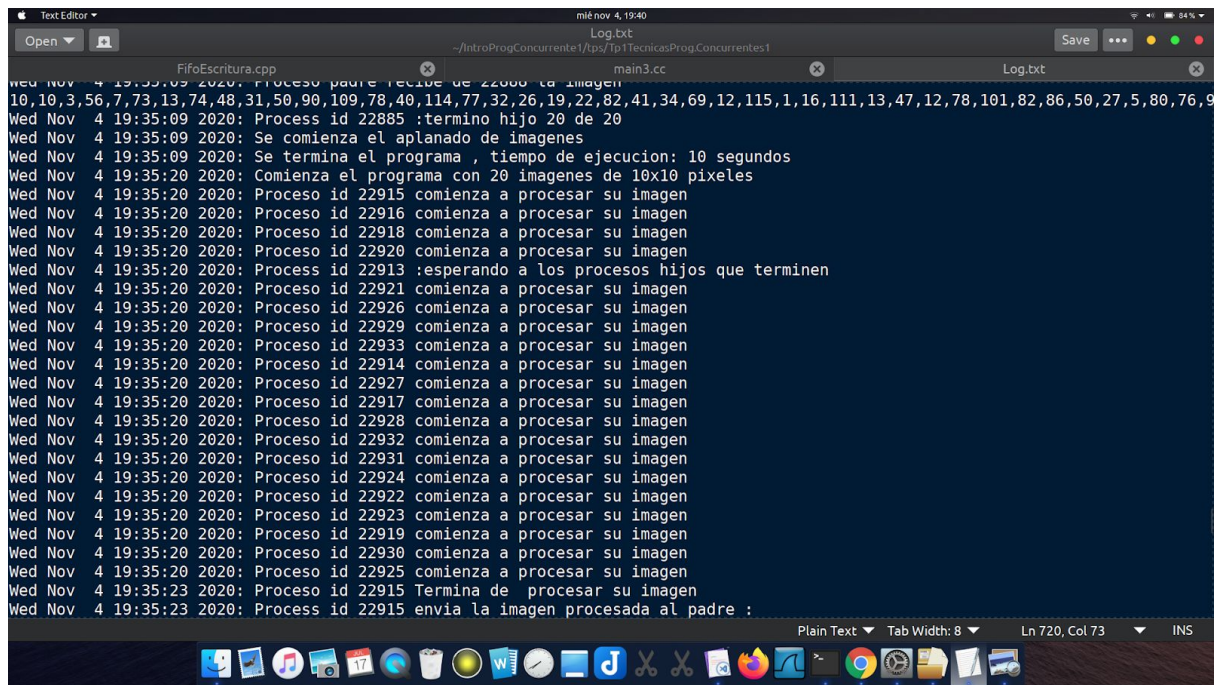
Visualizaciones

Ejemplo de ejecución de un 20 imágenes de 20 por 20 píxeles:



```
File Edit View Search Terminal Help
jorge@Integer: ~/IntroProgConcurrente1/tps/Tip1TecnicasProg.Concurentes1
jorge@Integer:~/IntroProgConcurrente1/tps/Tip1TecnicasProg.Concurentes1$ ./procesadorSharedMen 20 20
Imagen
64 55 43 44 22 93 24 5 71 60 116 39 117 37 26 35 28 39 31 33
24 10 115 105 32 31 2 80 12 3 96 11 57 73 53 78 38 12 18 44
8 69 82 124 104 42 94 4 16 123 99 39 68 86 79 99 52 80 50 63
Imagen
38 125 78 34 55 60 114 60 56 89 29 90 71 72 105 13 12 32 37 66
64 113 12 83 107 99 91 9 74 104 33 47 101 110 16 91 105 1 85 32
25 113 57 95 120 33 43 67 128 14 4 63 126 15 81 105 49 43 49 57
82 81 183 118 126 118 80 102 54 37 69 78 85 125 108 77 94 22 15 93
35 82 90 96 32 43 9 79 21 120 8 102 8 45 92 5 98 107 42 23
79 46 36 99 107 15 111 72 99 61 36 68 14 61 99 108 102 107 58 58
34 128 95 40 45 58 108 14 100 85 100 114 2 7 84 108 20 66 51 54
126 21 57 74 81 28 53 118 69 46 111 102 109 78 14 25 71 120 102 42
76 73 27 14 14 46 120 97 111 106 22 44 126 78 117 15 104 105 68 45
23 51 82 3 128 95 91 6 22 128 47 34 8 9 46 21 54 38 117 100
15 74 15 77 23 67 26 62 44 93 41 1 16 122 3 78 24 29 83 109
28 65 14 98 8 122 118 61 95 43 32 109 115 110 121 73 49 19 69 92
111 109 92 61 39 30 74 125 122 29 106 85 28 54 54 35 48 108 95 15
82 17 73 74 25 61 24 126 72 104 41 15 44 58 10 20 99 103 86 114
22 63 59 8 108 115 16 28 5 84 54 50 128 18 47 38 110 120 98 103
33 57 25 101 14 102 71 128 117 120 62 7 72 6 80 32 2 38 29 9
20 11 123 110 65 48 17 112 27 47 61 111 45 55 54 24 41 69 51 44
24 40 38 87 56 75 68 101 66 29 12 21 40 6 3 103 117 19 22 79
14 5 23 57 62 31 75 32 69 33 18 37 89 105 9 37 14 15 37 3
65 18 62 45 72 115 68 112 119 53 91 77 92 119 35 83 2 94 56 67
69 34 72 76 103 23 107 104 60 7 48 73 11 69 64 8 36 11 40 103
58 3 23 97 7 25 7 59 106 92 9 106 109 70 22 115 119 89 98 45
106 120 12 2 96 83 102 109 33 74 47 101 107 117 49 81 75 27 120 71
13 124 122 40 51 28 122 52 57 113 54 114 114 76 18 57 35 24 114 13
96 39 15 42 107 14 50 78 24 24 53 65 15 127 67 46 18 40 89 50
115 58 118 31 63 75 17 54 35 51 98 111 110 27 86 32 54 80 18 46
113 7 22 27 60 6 107 6 95 34 76 62 8 26 40 115 39 24 127 125
47 51 62 125 113 63 42 66 38 2 114 86 9 71 112 3 76 27 8 42
60 19 40 67 107 14 117 17 37 115 77 83 102 74 15 22 72 56 87 109
57 72 67 128 78 50 2 26 76 73 3 71 91 42 73 6 55 61 85 27
111 34 45 20 42 59 41 113 113 62 94 41 69 96 41 19 18 105 44 28
50 46 34 76 22 105 80 12 101 37 37 19 5 81 38 47 11 14 31 123
75 60 99 16 28 75 34 108 51 12 7 36 121 40 47 14 80 126 25 53
98 125 7 102 77 43 84 87 119 114 17 66 109 51 16 8 125 112 115 111
123 57 82 115 32 128 65 111 61 25 99 30 21 104 67 97 82 22 119 73
```

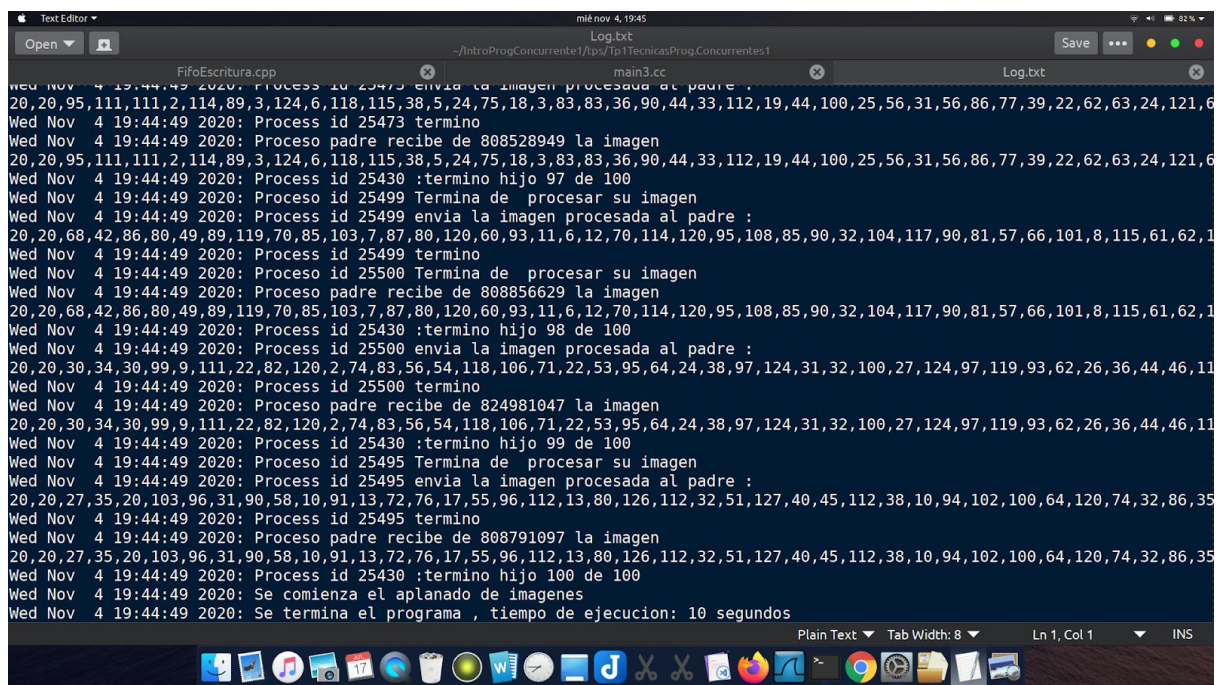
Ejemplo de archivo log.



```
Wed Nov 4 19:35:09 2020: Proceso padre recibe de 22885 la imagen
10,10,3,56,7,73,13,74,48,31,50,90,109,78,40,114,77,32,26,19,22,82,41,34,69,12,115,1,16,111,13,47,12,78,101,82,86,50,27,5,80,76,9
Wed Nov 4 19:35:09 2020: Process id 22885 :termino hijo 20 de 20
Wed Nov 4 19:35:09 2020: Se comienza el aplanado de imagenes
Wed Nov 4 19:35:09 2020: Se termina el programa , tiempo de ejecucion: 10 segundos
Wed Nov 4 19:35:20 2020: Comienza el programa con 20 imagenes de 10x10 pixeles
Wed Nov 4 19:35:20 2020: Proceso id 22915 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22916 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22918 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22920 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Process id 22913 :esperando a los procesos hijos que terminen
Wed Nov 4 19:35:20 2020: Proceso id 22921 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22926 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22929 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22933 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22914 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22927 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22917 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22928 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22932 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22931 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22924 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22922 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22923 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22919 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22930 comienza a procesar su imagen
Wed Nov 4 19:35:20 2020: Proceso id 22925 comienza a procesar su imagen
Wed Nov 4 19:35:23 2020: Proceso id 22915 Termina de procesar su imagen
Wed Nov 4 19:35:23 2020: Process id 22915 envia la imagen procesada al padre :
```

El procesamiento duro de las 20 imagenes duró 8 segundos.

Si en vez de 20 imágenes ingresamos 100 , el tiempo de procesamiento es de 10 segundos como muestra la siguiente imagen del archivo log.



```
Wed Nov 4 19:44:49 2020: Proceso id 25473 envia la imagen procesada al padre :
20,20,95,111,111,2,114,89,3,124,6,118,115,38,5,24,75,18,3,83,83,36,90,44,33,112,19,44,100,25,56,31,56,86,77,39,22,62,63,24,121,6
Wed Nov 4 19:44:49 2020: Process id 25473 termino
Wed Nov 4 19:44:49 2020: Proceso padre recibe de 808528949 la imagen
20,20,95,111,111,2,114,89,3,124,6,118,115,38,5,24,75,18,3,83,83,36,90,44,33,112,19,44,100,25,56,31,56,86,77,39,22,62,63,24,121,6
Wed Nov 4 19:44:49 2020: Process id 25430 :termino hijo 97 de 100
Wed Nov 4 19:44:49 2020: Proceso id 25499 Termina de procesar su imagen
Wed Nov 4 19:44:49 2020: Process id 25499 envia la imagen procesada al padre :
20,20,68,42,86,80,49,89,119,70,85,103,7,87,80,120,60,93,11,6,12,70,114,120,95,108,85,90,32,104,117,90,81,57,66,101,8,115,61,62,1
Wed Nov 4 19:44:49 2020: Process id 25499 termino
Wed Nov 4 19:44:49 2020: Proceso id 25500 Termina de procesar su imagen
Wed Nov 4 19:44:49 2020: Proceso padre recibe de 808856629 la imagen
20,20,68,42,86,80,49,89,119,70,85,103,7,87,80,120,60,93,11,6,12,70,114,120,95,108,85,90,32,104,117,90,81,57,66,101,8,115,61,62,1
Wed Nov 4 19:44:49 2020: Process id 25430 :termino hijo 98 de 100
Wed Nov 4 19:44:49 2020: Process id 25500 envia la imagen procesada al padre :
20,20,30,34,30,99,9,111,22,82,120,2,74,83,56,54,118,106,71,22,53,95,64,24,38,97,124,31,32,100,27,124,97,119,93,62,26,36,44,46,11
Wed Nov 4 19:44:49 2020: Process id 25500 termino
Wed Nov 4 19:44:49 2020: Proceso padre recibe de 824981047 la imagen
20,20,30,34,30,99,9,111,22,82,120,2,74,83,56,54,118,106,71,22,53,95,64,24,38,97,124,31,32,100,27,124,97,119,93,62,26,36,44,46,11
Wed Nov 4 19:44:49 2020: Process id 25430 :termino hijo 99 de 100
Wed Nov 4 19:44:49 2020: Proceso id 25495 Termina de procesar su imagen
Wed Nov 4 19:44:49 2020: Process id 25495 envia la imagen procesada al padre :
20,20,27,35,20,103,96,31,90,58,10,91,13,72,76,17,55,96,112,13,80,126,112,32,51,127,40,45,112,38,10,94,102,100,64,120,74,32,86,35
Wed Nov 4 19:44:49 2020: Process id 25495 termino
Wed Nov 4 19:44:49 2020: Proceso padre recibe de 808791097 la imagen
20,20,27,35,20,103,96,31,90,58,10,91,13,72,76,17,55,96,112,13,80,126,112,32,51,127,40,45,112,38,10,94,102,100,64,120,74,32,86,35
Wed Nov 4 19:44:49 2020: Process id 25430 :termino hijo 100 de 100
Wed Nov 4 19:44:49 2020: Se comienza el aplanado de imagenes
Wed Nov 4 19:44:49 2020: Se termina el programa , tiempo de ejecucion: 10 segundos
```

Esto última ilustra la rapidez del procesamiento paralelo utilizando memoria compartida para comunicar datos entre procesos.

Problemas del código:

Hay una limitación del tamaño de las imágenes a 30 pixels por 30 píxeles debido a la limitación de mil bytes de la memoria compartida. Esto se resolvería con la implementación de un método que escriba en la memoria en un ciclo y otro que lea la memoria en ciclo.

Ejercicio 2

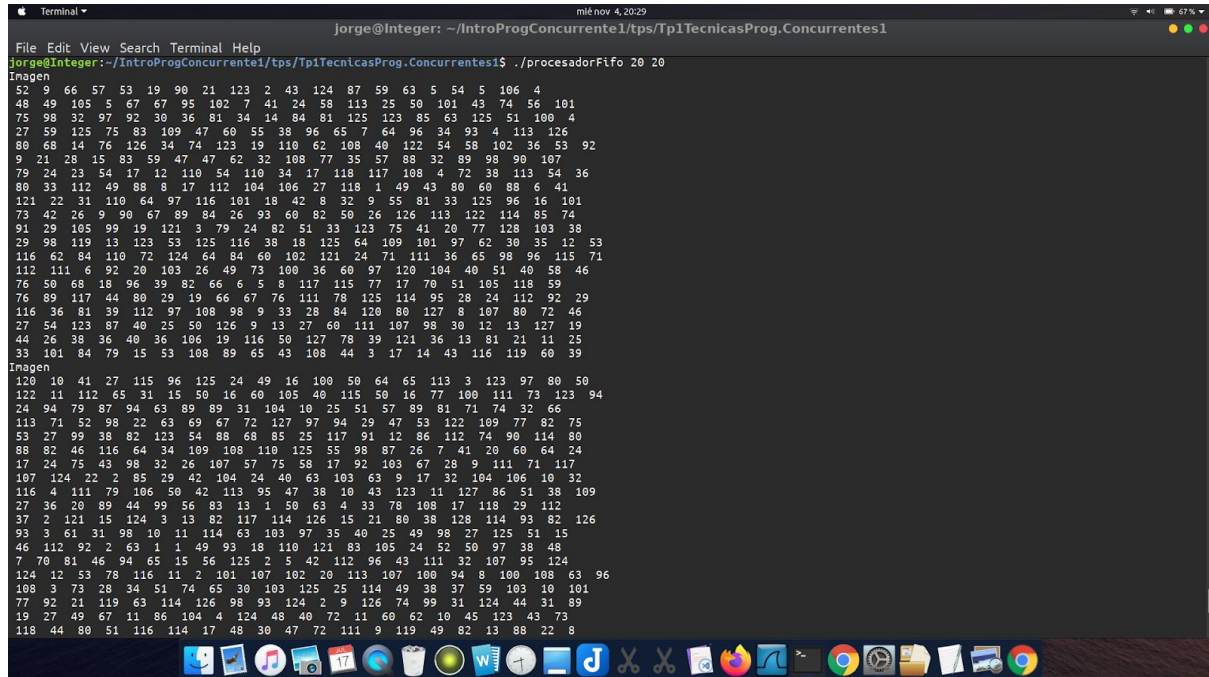
Se reutilizaron las clases del ejercicio 1.

Explicación del algoritmo

El algoritmo es similar al caso anterior con la diferencia que se usa un pipe denominado o Fifo en vez de memoria compartida. Por lo mientras que no se supere el buffer del pipe, no se requiere sincronizar los procesos. Por lo tanto no se utiliza la variable status como el caso anterior.

Visualizaciones

Ejemplo de ejecución de un 20 imágenes de 20 por 20 píxeles:



```
Terminal
jorge@Integer: ~/IntroProgConcurrente1/tps/Tp1TecnicasProg.Concurentes1
File Edit View Search Terminal Help
jorge@Integer:~/IntroProgConcurrente1/tps/Tp1TecnicasProg.Concurentes1$ ./procesadorFifo 20 20
Imagen
52 9 66 57 53 19 90 21 123 2 43 124 87 59 63 5 54 5 106 4
48 49 105 5 67 67 95 102 7 41 24 58 113 25 50 101 43 74 56 101
75 98 32 97 92 30 36 81 34 14 84 81 125 123 85 63 125 51 100 4
27 59 125 75 83 109 47 60 55 38 96 65 7 64 96 34 93 4 113 126
80 68 14 76 126 34 74 123 19 110 62 108 40 122 54 58 102 36 53 92
9 21 28 15 83 59 47 47 62 32 108 77 35 57 88 32 89 98 90 107
79 24 23 54 17 12 110 54 110 34 17 118 117 108 4 72 38 113 54 36
80 33 112 49 88 8 17 112 104 106 27 118 1 49 43 80 60 88 6 41
121 22 31 110 64 97 116 101 18 42 8 32 9 55 81 33 125 96 16 101
73 42 26 9 90 67 89 84 26 93 60 82 50 26 126 113 122 114 85 74
91 29 105 99 19 121 3 79 24 82 51 33 123 75 41 20 77 128 103 38
29 98 119 13 123 53 125 116 30 48 125 64 109 101 97 62 30 35 12 53
116 62 84 110 72 124 64 84 60 102 121 24 71 111 36 65 98 96 115 71
112 111 6 92 20 103 26 49 73 100 36 60 97 120 104 40 51 40 58 46
76 50 68 18 96 39 82 66 6 5 8 117 115 77 17 70 51 105 118 59
76 89 117 44 80 29 19 66 67 76 111 78 125 114 95 28 24 112 92 29
116 36 81 39 112 97 108 98 9 33 28 84 120 80 127 8 107 80 72 46
27 54 123 87 40 25 50 126 9 13 27 60 111 107 98 30 12 13 127 19
44 26 38 36 40 36 106 19 116 50 127 78 39 121 36 13 81 21 11 25
33 101 84 79 15 53 108 89 65 43 108 44 3 17 14 43 116 119 60 39
Imagen
120 10 41 27 115 96 125 24 49 16 100 50 64 65 113 3 123 97 80 50
122 11 112 65 31 15 50 16 60 105 40 115 50 16 77 100 111 73 123 94
24 94 79 87 94 63 89 89 31 104 10 25 51 57 89 81 71 74 32 66
113 71 52 98 22 63 69 67 72 127 97 94 29 47 53 122 109 77 82 75
53 27 99 38 82 123 54 88 68 85 25 117 91 12 86 112 74 90 114 80
88 82 46 116 64 34 109 108 110 125 55 98 87 26 7 41 20 60 64 24
17 24 75 43 98 32 26 107 57 75 58 17 92 103 67 28 9 111 71 117
107 124 22 2 85 29 42 104 24 40 63 103 63 9 17 32 104 106 10 32
116 4 111 79 106 50 42 113 95 47 38 10 43 123 11 127 86 51 38 109
27 36 20 89 44 99 56 83 13 1 50 63 4 33 78 108 17 118 29 112
37 2 121 15 124 3 13 82 117 114 126 15 21 80 38 128 114 93 82 126
93 3 61 31 98 10 11 114 63 103 97 35 40 25 49 98 27 125 51 15
46 112 92 2 63 1 1 49 93 10 110 121 83 105 24 52 50 97 33 48
7 70 81 46 94 65 15 56 125 2 5 42 112 96 43 111 32 107 95 124
124 12 53 78 116 11 2 101 107 102 20 113 107 100 94 8 100 108 63 96
108 3 73 28 34 51 74 65 30 103 125 25 114 49 38 37 59 103 10 101
77 92 21 119 63 114 126 98 93 124 2 9 126 74 99 31 124 44 31 89
19 27 49 67 11 86 104 4 124 48 40 72 11 60 62 10 45 123 43 73
118 44 80 51 116 114 17 48 30 47 72 111 9 119 49 82 13 88 22 8
```

El procesamiento duro de las 20 imágenes duró 9 segundos.

The screenshot displays a Windows desktop with a Visual Studio Code editor window open. The editor shows a C++ program named 'FifoEscritura.cpp' with a main function that uses threads to process images. The output window at the bottom shows the execution results, including the IDs of the processes and the time taken for each step. The program successfully demonstrates parallel image processing using threads and a queue.

Text Editor - File Edit View Extensions Search Run and Debug Test Explorer Output Console Run and Debug Console Log.txt

Open [Icons] - /IntroProgConcurrente1/fps/Tip1TecnicasProgConcurrentes1 Save [Icons] 67%

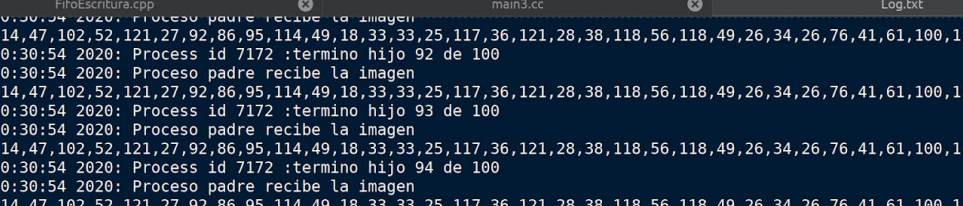
FifoEscritura.cpp [X] main.cc [X] Log.txt [X]

```

Wed Nov 4 20:29:09 2020: Proceso id 7918 termina de procesar su imagen
Wed Nov 4 20:29:09 2020: Process id 7918 envia la imagen procesada al padre :
20,20,95,67,6,47,64,38,71,86,41,91,83,85,89,54,97,103,16,86,112,14,49,50,72,74,69,16,121,80,110,23,57,77,89,62,59,24,34,1,45,74
Wed Nov 4 20:29:09 2020: Proceso padre recibe la imagen
20,20,95,67,6,47,64,38,71,86,41,91,83,85,89,54,97,103,16,86,112,14,49,50,72,74,69,16,121,80,110,23,57,77,89,62,59,24,34,1,45,74
Wed Nov 4 20:29:09 2020: Proceso id 7927 Termina de procesar su imagen
Wed Nov 4 20:29:09 2020: Process id 7927 envia la imagen procesada al padre :
20,20,96,80,104,29,40,76,20,81,24,49,100,84,110,115,40,121,55,48,81,92,85,5,46,69,109,55,73,126,88,35,93,119,50,68,83,25,79,102
Wed Nov 4 20:29:09 2020: Proceso id 7928 Termina de procesar su imagen
Wed Nov 4 20:29:09 2020: Process id 7928 envia la imagen procesada al padre :
20,20,122,97,48,7,53,6,99,106,17,7,22,26,73,1,86,48,46,51,83,4,76,78,75,46,104,36,35,115,76,91,17,69,123,127,75,47,4,109,88,21,
Wed Nov 4 20:29:09 2020: Process id 7172 :termino hijo 15 de 20
Wed Nov 4 20:29:09 2020: Proceso padre recibe la imagen
20,20,96,80,104,29,40,76,20,81,24,49,100,84,110,115,40,121,55,48,81,92,85,5,46,69,109,55,73,126,88,35,93,119,50,68,83,25,79,102
Wed Nov 4 20:29:09 2020: Process id 7172 :termino hijo 16 de 20
Wed Nov 4 20:29:09 2020: Proceso padre recibe la imagen
20,20,96,80,104,29,40,76,20,81,24,49,100,84,110,115,40,121,55,48,81,92,85,5,46,69,109,55,73,126,88,35,93,119,50,68,83,25,79,102
Wed Nov 4 20:29:09 2020: Process id 7172 :termino hijo 17 de 20
Wed Nov 4 20:29:09 2020: Proceso padre recibe la imagen
20,20,96,80,104,29,40,76,20,81,24,49,100,84,110,115,40,121,55,48,81,92,85,5,46,69,109,55,73,126,88,35,93,119,50,68,83,25,79,102
Wed Nov 4 20:29:09 2020: Process id 7172 :termino hijo 18 de 20
Wed Nov 4 20:29:09 2020: Proceso padre recibe la imagen
20,20,96,80,104,29,40,76,20,81,24,49,100,84,110,115,40,121,55,48,81,92,85,5,46,69,109,55,73,126,88,35,93,119,50,68,83,25,79,102
Wed Nov 4 20:29:09 2020: Process id 7172 :termino hijo 19 de 20
Wed Nov 4 20:29:09 2020: Proceso padre recibe la imagen
20,20,96,80,104,29,40,76,20,81,24,49,100,84,110,115,40,121,55,48,81,92,85,5,46,69,109,55,73,126,88,35,93,119,50,68,83,25,79,102
Wed Nov 4 20:29:09 2020: Process id 7172 :termino hijo 20 de 20
Wed Nov 4 20:29:09 2020: Se comienza el aplanado de imagenes
Wed Nov 4 20:29:09 2020: Se termina el programa , tiempo de ejecucion: 9 segundos
  
```

Plain Text Tab Width: 8 Ln 1, Col 1 INS

[Taskbar icons: File Explorer, Edge, VS Code, etc.]



The screenshot shows a Windows desktop with two windows open. The background window is a Windows Explorer window showing the contents of the 'C:\Program Files\Google\Chrome\Application' folder, listing various system files and folders. The foreground window is a Visual Studio Code editor window with the file 'main3.cc' open. The code in the editor is a C++ program that simulates a concurrent image processing task. It features a parent process that forks 100 child processes, each of which processes a portion of an image and then returns control to the parent. The output of the program is visible in the terminal at the bottom of the editor window, showing the execution of the code and the completion of the image processing task.

```
Text Editor v8.0.0
Open
Log.txt
~/fntroProgConcurrente1/tps/Tip1/TechnicasProgConcurrentes1
Save
FifoEscritura.cpp
main3.cc
Log.txt

Wed Nov 4 20:30:54 2020: Proceso padre recibe la imagen
20,20,36,89,14,47,102,52,121,27,92,86,95,114,49,18,33,33,25,117,36,121,28,38,118,56,118,49,26,34,26,76,41,61,100,117,43,74,104,9
Wed Nov 4 20:30:54 2020: Process id 7172 :termino hijo 92 de 100
Wed Nov 4 20:30:54 2020: Proceso padre recibe la imagen
20,20,36,89,14,47,102,52,121,27,92,86,95,114,49,18,33,33,25,117,36,121,28,38,118,56,118,49,26,34,26,76,41,61,100,117,43,74,104,9
Wed Nov 4 20:30:54 2020: Process id 7172 :termino hijo 93 de 100
Wed Nov 4 20:30:54 2020: Proceso padre recibe la imagen
20,20,36,89,14,47,102,52,121,27,92,86,95,114,49,18,33,33,25,117,36,121,28,38,118,56,118,49,26,34,26,76,41,61,100,117,43,74,104,9
Wed Nov 4 20:30:54 2020: Process id 7172 :termino hijo 94 de 100
Wed Nov 4 20:30:54 2020: Proceso padre recibe la imagen
20,20,36,89,14,47,102,52,121,27,92,86,95,114,49,18,33,33,25,117,36,121,28,38,118,56,118,49,26,34,26,76,41,61,100,117,43,74,104,9
Wed Nov 4 20:30:54 2020: Proceso padre recibe la imagen
20,20,36,89,14,47,102,52,121,27,92,86,95,114,49,18,33,33,25,117,36,121,28,38,118,56,118,49,26,34,26,76,41,61,100,117,43,74,104,9
Wed Nov 4 20:30:54 2020: Process id 7172 :termino hijo 96 de 100
Wed Nov 4 20:30:54 2020: Proceso padre recibe la imagen
20,20,36,89,14,47,102,52,121,27,92,86,95,114,49,18,33,33,25,117,36,121,28,38,118,56,118,49,26,34,26,76,41,61,100,117,43,74,104,9
Wed Nov 4 20:30:54 2020: Process id 7172 :termino hijo 97 de 100
Wed Nov 4 20:30:54 2020: Proceso padre recibe la imagen
20,20,36,89,14,47,102,52,121,27,92,86,95,114,49,18,33,33,25,117,36,121,28,38,118,56,118,49,26,34,26,76,41,61,100,117,43,74,104,9
Wed Nov 4 20:30:54 2020: Process id 7172 :termino hijo 98 de 100
Wed Nov 4 20:30:54 2020: Proceso padre recibe la imagen
20,20,36,89,14,47,102,52,121,27,92,86,95,114,49,18,33,33,25,117,36,121,28,38,118,56,118,49,26,34,26,76,41,61,100,117,43,74,104,9
Wed Nov 4 20:30:54 2020: Process id 7172 :termino hijo 99 de 100
Wed Nov 4 20:30:54 2020: Proceso padre recibe la imagen
20,20,36,89,14,47,102,52,121,27,92,86,95,114,49,18,33,33,25,117,36,121,28,38,118,56,118,49,26,34,26,76,41,61,100,117,43,74,104,9
Wed Nov 4 20:30:54 2020: Process id 7172 :termino hijo 100 de 100
Wed Nov 4 20:30:54 2020: Se comienza el aplanado de imagenes
Wed Nov 4 20:30:54 2020: Se termina el programa , tiempo de ejecucion: 10 segundos
```


un método que escriba en la memoria en un ciclo primero el tamaño de la imagen serializada en bytes y luego enviar la imagen en fragmentos. El proceso padre debería leer del pipe la imagen serializada con un método que ejecute un ciclo.

Asimismo se debería implementar un mecanismo de sincronización para evitar que un proceso escriba en el pipe cuando otro no haya terminado de enviar la imagen serializada

Dificultades encontradas

En primer lugar tuve que recordar cómo codificar en c++ ya que hace más de una año que utilizaba el lenguaje. Luego me di cuenta tarde de las limitaciones de tamaño de la imagen que imponía no haber implementado funciones que envíen y escriban de a fragmentos. Cuando me di cuenta de esto último pude implementarlo pero detecte fugas de memoria con Valgrind que no puede resolver a tiempo por lo no incorporé las mejoras al código.

Conclusiones

Pude observar la rapidez de el uso de memoria compartida junto con el uso de la concurrencia para ejecutar algoritmos que requieren un cierto tiempo de ejecución. También pude comprobar que es más sencillo utilizar pipes por sobre memoria compartida porque provee un mecanismo de sincronización del tipo first in first out. Sin embargo esto es cierto si no se el tamaño de los datos que se quieran enviar no supere el del buffer del pipe. Si ocurriese esto se debería implementar un mecanismo de sincronización como se mencionó en el punto anterior.