Informe Segundo Proyecto Sistemas Operativos

Análisis Concurrente de un Log, usando una estrategia MapReduce

Autores

Sergio Andrés Mejía Tovar **Santiago Palacios Loaiza** **Julián David Parada Galvis**

Descripción de cada implementación

Procesos

Como primera opción se tuvo el uso de procesos para solucionar el problema planteado inicialmente.Una de las características principales de esta implementación es que por la naturaleza limitante de los procesos, no es posible comunicar los procesos entre sí, estos no comparten memoria y son independientes. Sin embargo, como la comunicación entre partes es fundamental para el algoritmo planteado y el buen funcionamiento del programa es obligatorio buscar una solución. La solución planteada fue la creación de archivos de texto plano, en los cuales las partes escriben o leen la información correspondiente y necesaria para el funcionamiento óptimo del programa. Finalmente los procesos son creados cada vez que se realiza una consulta y los archivos de texto depende de lo ingresado en consola por el usuario.

Hilos

Las segunda opción es mediante el uso de hilos. La ventaja principal de los hilos es que estos sí se pueden comunicar entre sí, debido a que comparten una memoria. En esta implementación la memoria dinámica compartida por los hilos hace de medio de comunicación evitando la creación de archivos de texto y haciendo así que el programa sea mucho más rápido. En este caso otra vez cada consulta activa la destruccion y creacion de nuevos hilos.

Hilos y Semáforos

Para finalizar tenemos la solución que usa tanto hilos como semáforos, esta implementación es muy similar a la anterior con hilos pero aprovechando todas la ventajas que ofrecen los semáforos, entre estas, la protección y el manejo de regiones de memoria crítica y el no uso de recursos de manera innecesaria con Busy Waiting. Otra característica de los semáforos que puede ser vista desde de el lado de las ventajas y de las desventajas es la independencia de los semáforos de la máquina, los semáforos son manejados por los desarrolladores y requieren ser manejados con cuidado para evitar interbloqueos. En el caso del proyecto presente se implementó una solución muy similar al algoritmo del productor-consumidor. En esta última implementación las consultas de una misma ejecución del programa usan los mismos hilos, esto es, no se eliminan y se crean hilos para cada consulta si no para cada ejecución del programa, esto requirió que toda la información de los hilos fue inicializada para mantener independientes los resultados de cada consulta.

Decisión de la tercera opción.

La decisión de usar el par de hilos-semáforos en lugar de usar procesos-pipes está respaldada por varias razones. La primera razón es que considerando que en la entrega anterior los hilos tuvieron un mejor desempeño que los procesos, es lógico pensar que mediante el uso de otras herramientas esta eficiencia se puede potenciar para alcanzar una eficiencia aún mayor. Otra razón importante es que los pipes son archivos, y a pesar de que no funcionan igual que los archivos de texto plano de la primera implementación de procesos y de que los pipes tienen un funcionamiento más óptimo, vemos una pequeña similitud que, en primera instancia, inclina la balanza a favor del uso de los semáforos. Una de las razones principales es la similitud del problema planteado con la solución que ofrece el algoritmo de Productor-Consumidor, clásico en las implementaciones de semáforos. El algoritmo implementado se muestra a continuación.

En la figura 1 se ve un esquema de la arquitectura del programa. A la parte izquierda se tiene el Master, la ejecución del programa en sí, a la derecha se tienen en círculos con la letra 'M' los Mappers, luego a su derecha se tienen los buffers Mappers-Reducer, donde hay un buffer por cada Reducer del programa, posteriormente, en la misma dirección y representado con círculos con la letra 'R' los Reducer, finalmente al fondo a la izquierda se tiene el buffer Reducers-Master. Para cada sección de buffers se utiliza el algoritmo de Productor-Consumidor. Los semáforos "m_write" y "r_write" sirven como controles de las regiones críticas, los buffers, también cuentan con los semáforos 'n' y 'e' que se utilizan de la manera clásica para controlar la cantidad de elementos en el buffer, teniendo en cuenta que los buffers tienen tamaños limitados y constantes. Como medida adicional y debido a que en esta implementación hay hilos que cumplen tanto la función de productor, como el rol de consumidor fue necesario el uso de variables por cada buffer de nombre "tam[i]" que junto a variables del código ayudan a decidir hasta qué momento un consumidor espera datos de los productores, ya que en el caso de este programa específico la información es finita y el hecho de que un consumidor se queda esperando datos para leer no es óptimo. Así los productores dejan datos que funcionan como banderas en los buffers,que a su vez, son leídos por los consumidores que llevan la cuenta de la cantidad de banderas para saber cuando dejar de leer en esos buffers.

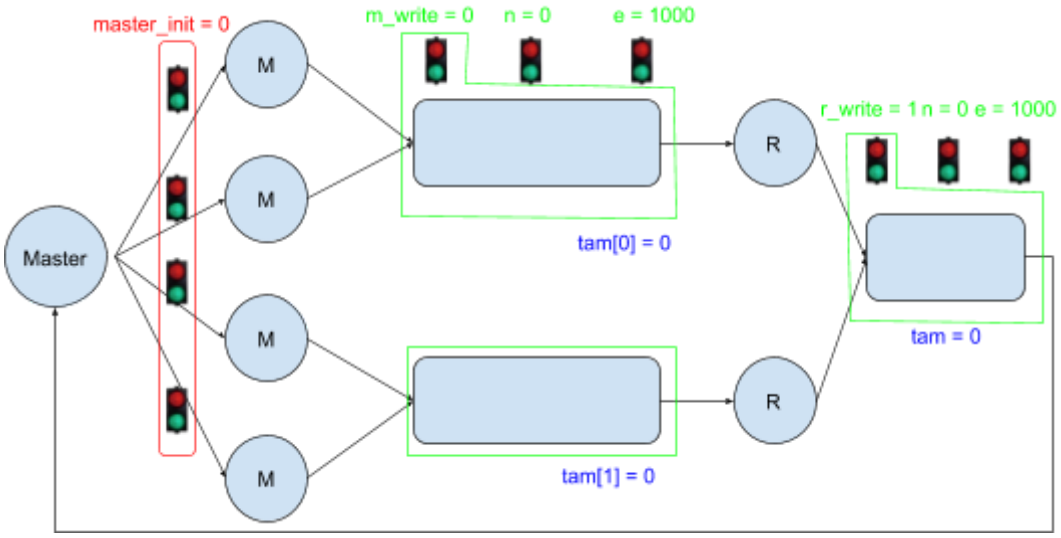


Figura 1. Esquema del diseño del programa con los semáforos y buffers usados.

Desempeño de los programas

La consulta realizada a modo de prueba en las tres implementaciones fue (5, >, 1) lo que significa todos los procesos que usaron más de un procesador en su ejecución. Para mantener las pruebas constantes en todas las implementaciones, todas estas se realizaron con 6 Mappers y 4 Reducers. Para el caso de la implementación de procesos el valor de intermedios debe ser igual a 0. Por último para mantener resultados fidedignos e intentar que representar mejor la información se realizaron tres veces consecutivas la misma prueba en cada implementación. Las consultas, los resultados, los promedios y la gráfica que generaliza la información se muestran a continuación:

- ➢ 1000 logs (se usa el archivo de la entrega anterior pues en esta no se disponía de uno con esta cantidad de logs)

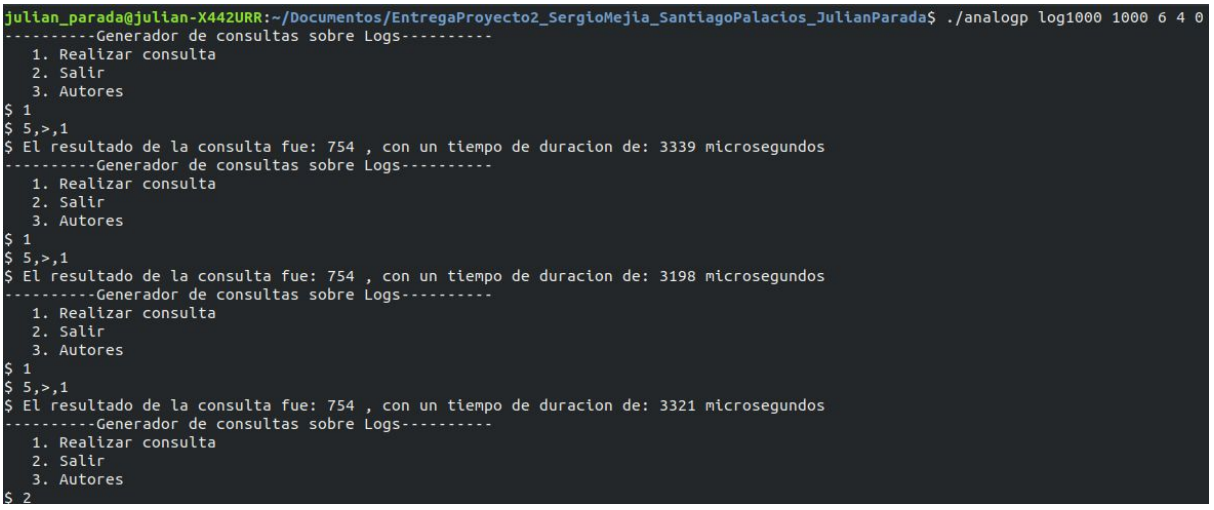


Imagen 1. Prueba de la consulta se obtiene un tiempo promedio con la implementación de procesos de 3286 microsegundos.

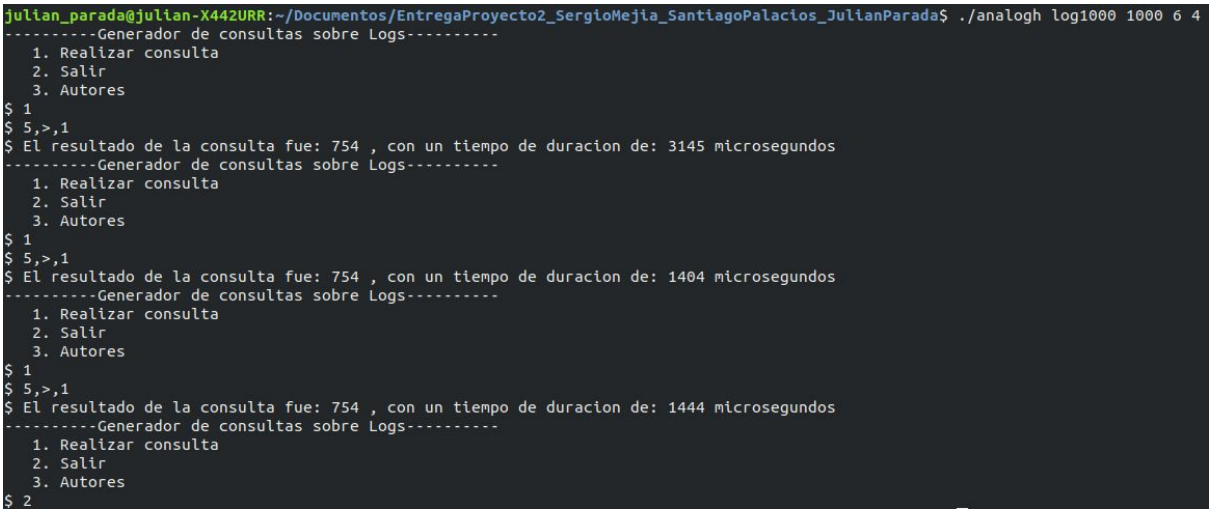


Imagen 2. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos de 1997 microsegundos.

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogh2 log1000 1000 6 4
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 754 , con un tiempo de duracion de: 2339 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 754 , con un tiempo de duracion de: 2165 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 754 , con un tiempo de duracion de: 1859 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
Mapper con ID 0 termina
Mapper con ID 1 termina
Mapper con ID 2 termina
Mapper con ID 3 termina
Mapper con ID 4 termina
Mapper con ID 5 termina
Reducer con ID 0 termina
Reducer con ID 1 termina
Reducer con ID 2 termina
Reducer con ID 3 termina
```

Imagen 3. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos y semáforos de 2121 microsegundos.

- ➢ 10000 logs

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogp log10000 10000 6 4 0
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 7228 , con un tiempo de duracion de: 11378 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 7228 , con un tiempo de duracion de: 11081 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 7228 , con un tiempo de duracion de: 10833 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
```

Imagen 4. Prueba de la consulta se obtiene un tiempo promedio con la implementación de procesos de 11097 microsegundo.

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogh log10000 10000 6 4
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 7228 , con un tiempo de duracion de: 11845 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 7228 , con un tiempo de duracion de: 11009 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 7228 , con un tiempo de duracion de: 10883 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
```

Imagen 5. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos de 11246 microsegundos.

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogh2 log10000 10000 6 4
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 7228 , con un tiempo de duracion de: 13376 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 7228 , con un tiempo de duracion de: 13283 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 7228 , con un tiempo de duracion de: 12639 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
Mapper con ID 0 termina
Mapper con ID 1 termina
Mapper con ID 2 termina
Mapper con ID 3 termina
Mapper con ID 4 termina
Mapper con ID 5 termina
Reducer con ID 0 termina
Reducer con ID 1 termina
Reducer con ID 2 termina
Reducer con ID 3 termina
```

Imagen 6. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos y semáforos de 13099 microsegundos.

➢ 20000 logs

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogp log20K 20000 6 4 0
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 10054 , con un tiempo de duracion de: 22208 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 10054 , con un tiempo de duracion de: 22449 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 10054 , con un tiempo de duracion de: 21651 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
```

Imagen 7. Prueba de la consulta se obtiene un tiempo promedio con la implementación de procesos de 22103 microsegundos.

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogh log20K 20000 6 4
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 10054 , con un tiempo de duracion de: 22633 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 10054 , con un tiempo de duracion de: 20338 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 10054 , con un tiempo de duracion de: 18407 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
```

Imagen 8. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos de 20459 microsegundos.

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogh2 log20K 20000 6 4
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 10054 , con un tiempo de duracion de: 21316 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 10054 , con un tiempo de duracion de: 21599 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 10054 , con un tiempo de duracion de: 23022 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
Mapper con ID 0 termina
Mapper con ID 1 termina
Mapper con ID 2 termina
Mapper con ID 3 termina
Mapper con ID 4 termina
Mapper con ID 5 termina
Reducer con ID 0 termina
Reducer con ID 1 termina
Reducer con ID 2 termina
Reducer con ID 3 termina
```

Imagen 9. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos y semáforos de 21979 microsegundos.

- ➢ 30000 logs

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogp log30000 30000 6 4 0
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 11971 , con un tiempo de duracion de: 41477 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 11971 , con un tiempo de duracion de: 39781 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 11971 , con un tiempo de duracion de: 39586 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
```

Imagen 10. Prueba de la consulta se obtiene un tiempo promedio con la implementación de procesos de : 40281 microsegundos.

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogh log30000 30000 6 4
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 11971 , con un tiempo de duracion de: 39712 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 11971 , con un tiempo de duracion de: 40258 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 11971 , con un tiempo de duracion de: 38536 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
```

Imagen 11. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos de 39502 microsegundos.

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogh2 log30000 30000 6 4
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 11971 , con un tiempo de duracion de: 24057 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 11971 , con un tiempo de duracion de: 26016 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 11971 , con un tiempo de duracion de: 26474 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
Mapper con ID 0 termina
Mapper con ID 1 termina
Mapper con ID 2 termina
Mapper con ID 3 termina
Mapper con ID 4 termina
Mapper con ID 5 termina
Reducer con ID 0 termina
Reducer con ID 1 termina
Reducer con ID 2 termina
Reducer con ID 3 termina
```

Imagen 12. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos y semáforos de 25516 microsegundos.

- 40000 logs

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogp log40K 40000 6 4 0
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 21554 , con un tiempo de duracion de: 81233 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 21554 , con un tiempo de duracion de: 82165 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 21554 , con un tiempo de duracion de: 83104 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
```

Imagen 13. Prueba de la consulta se obtiene un tiempo promedio con la implementación de procesos de 82167 microsegundos

```
julian_parada@julian-X442URR:~/Documentos/EntregaProyecto2_SergioMejia_SantiagoPalacios_JulianParada$ ./analogh log40K 40000 6 4
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 21554 , con un tiempo de duracion de: 89626 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 21554 , con un tiempo de duracion de: 115613 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 1
$ 5,>,1
$ El resultado de la consulta fue: 21554 , con un tiempo de duracion de: 83962 microsegundos
----------Generador de consultas sobre Logs----------
   1. Realizar consulta
   2. Salir
   3. Autores
$ 2
```

Imagen 14. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos de 96400 microsegundos.

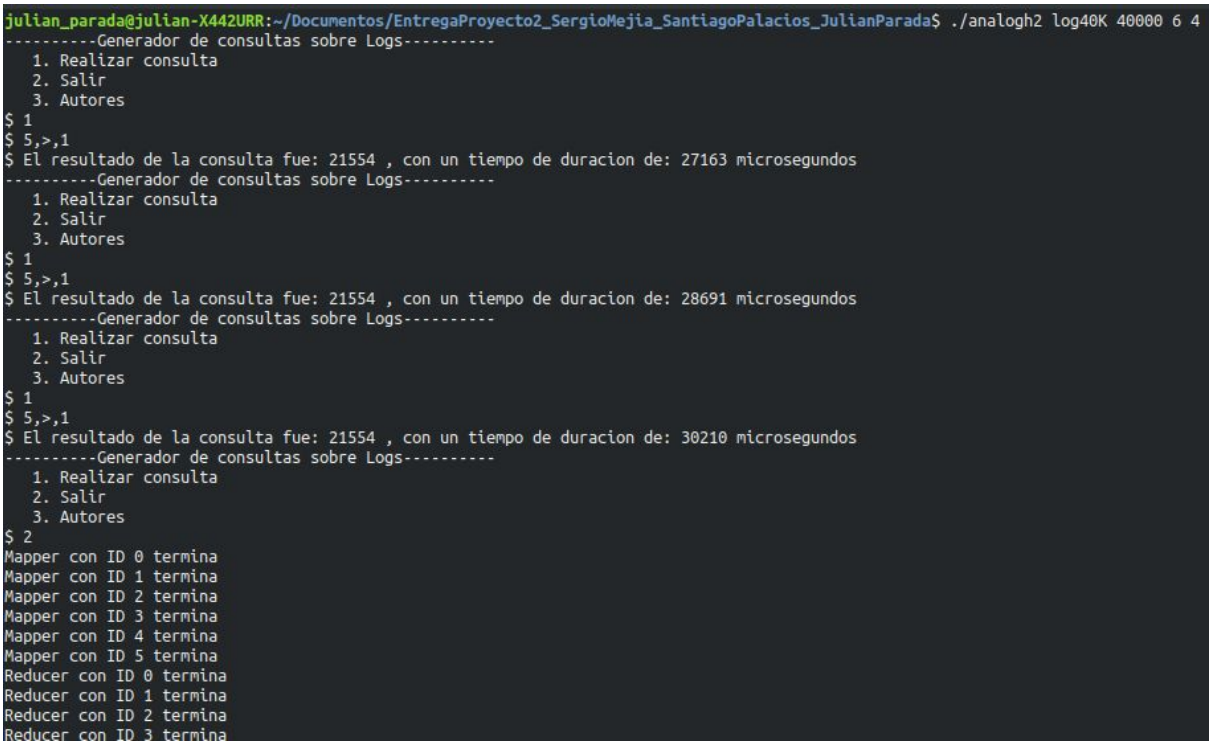
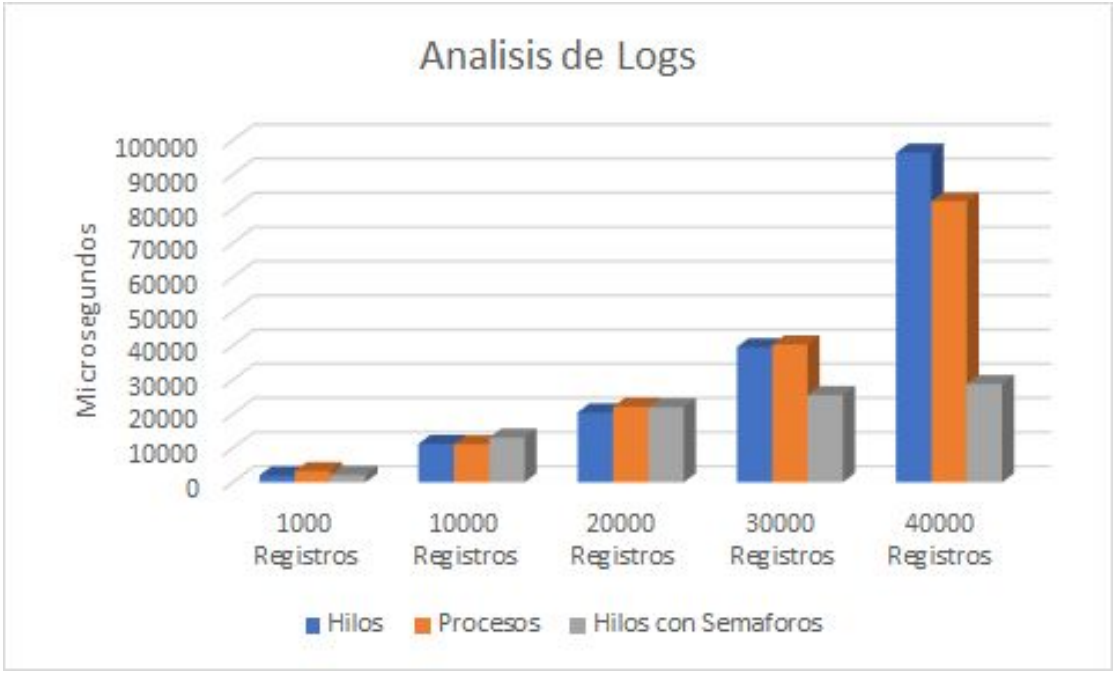


Imagen 14. Prueba de la consulta se obtiene un tiempo promedio con la implementación de hilos y semáforos de 28688 microsegundos.

	1000 Registros	10000 Registros	20000 Registros	30000 Registros	40000 Registros
Hilos (se crean y destruyen hilos para cada consulta). Comunicación con memoria compartida	1997 microsegundos	11246 microsegundos	20459 microsegundos	39502 microsegundos	96400 microsegundos
Procesos (los procesos se comunican a través de archivos, se crean y destruyen procesos en cada consulta)	3286 microsegundos	11097 microsegundos	22103 microsegundos	40281 microsegundos	82167 microsegundos
Hilos (los Mappers y Reducers se crean solo una vez, se comunican a través de buffers compartidos)	2121 microsegundos	13099 microsegundos	21979 microsegundos	25516 microsegundos	28688 microsegundos

De la tabla de resultados de desempeño podemos crear la siguiente gráfica:



Tras haber realizado las consultas de manera equitativa bajo condiciones relativamente controladas, podemos ver claramente en la gráfica como cambia el rendimiento de cada implementación. Para comenzar, como se esperaba, el par hilos-semáforos fue la implementación más efectiva, dando los resultados correctos en un tiempo mucho menor que sus implementaciones rivales. El uso de los semáforos para potenciar los hilos es mucho más notable cuando el tamaño de los logs comienza a crecer, de hecho hasta el log de 20000 registros el rendimiento es relativamente similar, con los hilos sin semáforos siendo incluso un poco más efectivos, sin embargo, tras pasar de esta cantidad de registros el uso de semáforos marca una diferencia muy significativa llegando a necesitar menos de la mitad del tiempo que requieren las otras implementaciones al enfrentarse al log de 40000 registros. Esta mejora en la eficiencia la podemos atribuir al uso del algoritmo del productor consumidor, donde por ejemplo, los Reducer no tienen que esperar que los Mappers terminen de escribir en los buffers para comenzar a leer de los mismos, disminuyendo asi el tiempo requerido por la implementación para terminar la consulta satisfactoriamente.

Una observación interesante tras estudiar los datos es que cuando el tamaño de registros alcanza el máximo analizado (40000), los hilos se vuelven menos efectivos que los procesos y no por la mínima diferencia que había aventajado los hilos hasta los logs anteriores, la posible explicación de este fenómeno puede estar en la misma característica que le da ventaja a los hilos en logs con menos registros y es el uso de la memoria dinámica. Es probable que frente a cantidades de registros tan altas el uso de memoria dinámica sea menos efectiva debido a que esta necesita copiar, eliminar, crear y escribir información por cada nuevo dato que se agregue, este proceso a gran escala puede terminar siendo más lento que la creación de archivos de texto plano pues estos no modifican mucho su arquitectura de datos tras leer más información. Esto también explicaría porque el dúo hilos-semáforos es más efectivo en grandes cantidades y es que debido al

algoritmo de productor-consumidor los buffers donde se almacenan los datos nunca son reestructurados si no la información de estos va cambiando.

Como un dato añadido, tras varias pruebas se hizo evidente un mínimo error en la ejecución del programa, este solo se presentaba con el log más grande, el error hacía que el programa arrojará resultados de consultas erróneos y siempre menores a los resultados correctos, además esto solo ocurría en la primera consulta de la ejecución. Tras análisis se llegó a la conclusión de que el error es debido a cómo se leen los datos, al parecer cuando se tiene un tamaño tan masivo de registros el programa comienza a trabajar sin haber leído por completo el log, para evitarlo sólo es necesario esperar unos segundos antes de realizar la consulta y asi se da tiempo suficiente para leer todos los registros.

Para concluir podemos decir con seguridad que el uso de hilos-semáforos es la implementación más efectiva, especialmente cuando el tamaño de los logs crece, debido al algoritmo del productor-consumidor, pero que sin embargo su desempeño no es muy notable cuando la cantidad de registros es mucho menor, donde las otras implementaciones mantienen un rendimiento casi par.