UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Departamento de Informática
Universidad Técnica Federico Santa María

Comparación entre algoritmos de coordinación en un sistema de biblioteca distribuida usando GO y gRPC.

Nicolás Durán 201673513-5
Matías Marchant 201673556-9

2 de Diciembre de 2020

1. Qué se hizo y Cómo se hizo

Se implementó una biblioteca como un sistema de archivos distribuidos con dos algoritmos de coordinación en lenguaje Go y se usó gRPC y Protocol Buffers para la comunicación entre nodos.

Para lograr el sistema se contó con 3 tipos de nodos:

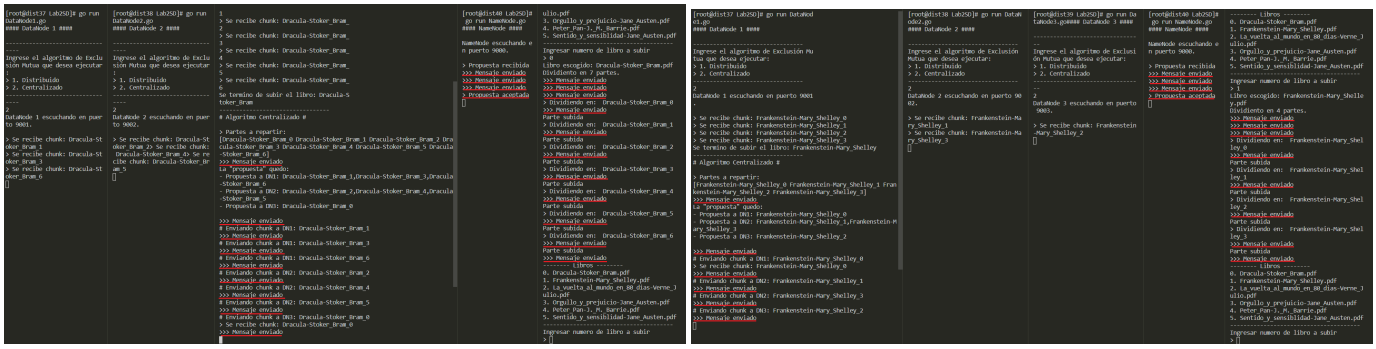
1. **Cliente:** Actúa como **Cliente Uploader** encargándose de separar los libros en chunks de tamaño 250 kB y cargarlos a la biblioteca comunicándose con un DataNode. También actúa como **Cliente Downloader** encargándose de reconstruir los libros de la biblioteca, comunicándose con el NameNode para conocer la ubicación de los chunks y así poder descargarlos de los DataNodes.
2. **DataNode:** Encargados de recibir chunks del Cliente Uploader, para después generar una propuesta de distribución de chunks entre DataNodes, la cual debe ser aprobada por los demás DataNodes si se está trabajando con el **algoritmo distribuido** o por el NameNode si se trabaja con el **algoritmo centralizado**. Una vez la propuesta es aceptada, es decir que cada DataNode funcionando reciba al menos un chunk, se registra la ubicación de todos los chunks en un registro log.txt del NameNode.
3. **NameNode:** Encargado de almacenar el registro log.txt que guarda la ubicación de las partes de cada libro, además de mostrar el listado de libros disponibles al Cliente Downloader. Si se trabaja con el **algoritmo centralizado**, el NameNode tiene papel de coordinador en caso de que más de un DataNode desee realizar modificaciones concurrentemente en el registro y también de generar propuestas de distribución si rechaza la propuesta recibida.

En el algoritmo distribuido, si dos datanodes desean modificar concurrentemente el log.txt del NameNode, se pidió el uso del algoritmo de Ricart y Agrawala, si bien se pidió, no lo implementamos, aún así se considerará en el análisis.

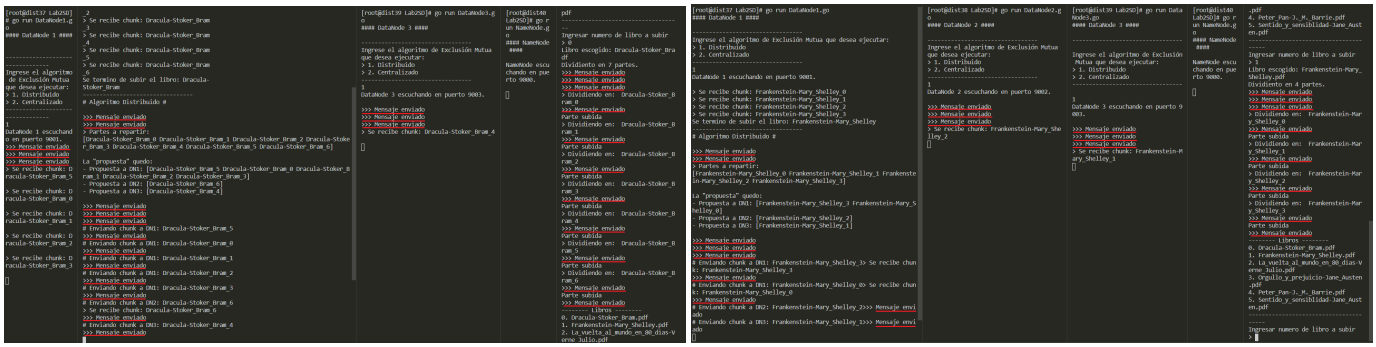
2. Resultados

Las métricas para obtener resultados son la cantidad de mensajes enviados y el tiempo que se demora en escribir en el archivo LOG. Para obtener la cantidad de mensajes enviados se imprime por pantalla los instantes en donde se envían mensajes (cuando se usa gRPC), y para los tiempos también se imprime por pantalla el valor de la demora en segundos. A continuación se muestran las capturas para cada caso.

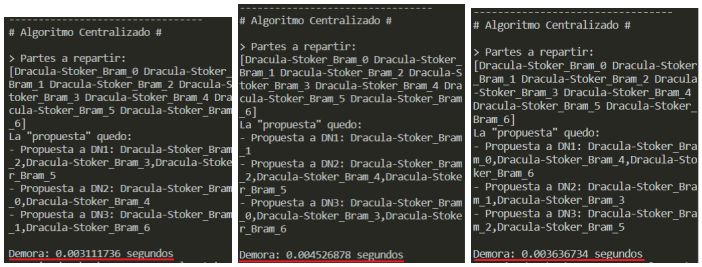
Cantidad de mensajes para algoritmo centralizado:



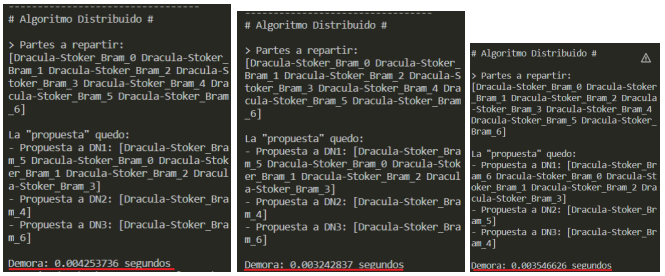
Cantidad de mensajes para algoritmo distribuido:



Tiempos para algoritmo centralizado:



Tiempos para algoritmo distribuido:



Se evaluó la carga de dos libros, el primero que se separa en 7 chunks, y el segundo que se separa en 4 chunks. Se presenta la siguiente tabla:

Mensajes	Libro 1 (7)	Libro 2 (4)
Distribuido	58	46
Centralizado	46	34

Cuadro 1: Cantidad de mensajes enviados según algoritmo y libro

Se evaluó el tiempo de creación y aprobación de propuesta de un libro, con 6 chunks, tres veces en cada algoritmo para obtener un tiempo promedio. Obteniéndose:

	Tiempo [s]
Distribuido	0.00375845
Centralizado	0.00368107

Cuadro 2: Tiempo promedio de creación y aprobación de propuesta según algoritmo

3. Análisis

Para cada algoritmo con **N** DataNodes, se realiza un ruteo general de los mensajes enviados en cada carga de libro separado en **i** chunks:

1. ClienteUploader envía mensajes KeepAlive a los N DataNodes para saber con cuál comunicarse $\implies$ **2N Mensajes**
2. Una vez conectado con un DataNode funcionando, envía i chunks y un mensaje final notificando que se terminó de enviar un libro $\implies$ **2i + 2 Mensajes**
3. Desde acá lo que sigue es diferente según el algoritmo
   a) Para el algoritmo distribuido, el DataNode que recibió los chunks, envía mensajes KeepAlive a los demás DataNodes para crear una propuesta correcta $\implies$ **2(N-1) Mensajes**

b) Envía la propuesta creada a los otros DataNode para recibir aprobación $\implies$ **2(N-1) Mensajes**

c) Cada DataNode que recibió una propuesta envía mensajes KeepAlive a los otros DataNode para revisar factibilidad de la propuesta y así no existan conflictos $\implies$ **2(N-1)N Mensajes**

d) El DataNode usa gRPC para escribir en el log $\implies$ **2 Mensajes**

---

a) Para el algoritmo centralizado, el DataNode que recibió los chunks, genera una propuesta y se la envía al NameNode $\implies$ **2 Mensajes**

b) El NameNode envía mensajes KeepAlive a los DataNodes involucrados en la propuesta para verificar conflictos $\implies$ **2N Mensajes**

c) Ya aprobada la propuesta por el NameNode, el DataNode con los chunks escribe en el log $\implies$ **2 Mensajes**

4. El DataNode con los chunks los reparte entre los demás y sí mismo también $\implies$ **2i Mensajes**

La fórmula obtenida para calcular la cantidad de mensajes en el algoritmo distribuido es: **MDistribuido = 2(N-1)(N+2) + 2N + 4i + 4**, y para el algoritmo centralizado es: **MCentralizado = 4N + 4i + 6**. En el algoritmo distribuido, si se hubiera implementado el algoritmo Ricart-Agrawala, entonces se agregarían **2(N-1) Mensajes** más antes de que el DataNode escriba en el log.

4. Discusión

Se esperaba que el algoritmo distribuido invirtiera más tiempo en la confección de propuestas debido a la mayor cantidad de mensajes enviados comparado al algoritmo centralizado, lo que se verifica con los resultados obtenidos.

Si bien a la vez de repartir chunks a todos los DataNodes se comete el error de repartirse chunks a sí mismo, el error se comete en la implementación de ambos algoritmos, por lo que no afecta en la comparación.

Para terminar la discusión, se aprecia un trade off entre tolerancia a fallas y ancho de banda usado/performance en los algoritmos. En el algoritmo distribuido los DataNodes se comunican con el NameNode sólo para escribir en el log, pudiendo aún confeccionar propuestas, a costo de una mayor cantidad de mensajes circulando en el sistema, mientras que en el algoritmo centralizado existe una cantidad mucho menor de mensajes transportados pero si el NameNode deja de funcionar, entonces no se puede continuar.

5. Conclusión

Para la implementación presentada anteriormente se obtienen las siguientes fórmulas para calcular los mensajes enviados por cada algoritmo: **MDistribuido = 2(N-1)(N+2) + 2N + 4i + 4**, **MCentralizado = 4N + 4i + 6**. Se comprueba que la cantidad de mensajes enviados en el algoritmo distribuido es mucho mayor comparado al algoritmo centralizado, lo que también permite concluir que el algoritmo distribuido toma más tiempo en aprobar propuestas que el algoritmo centralizado.