SISTEMAS DISTRIBUIDOS 2020
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

Autores :

Jorge Galleguillos M. - 201473545-6

Pablo Villalón C. - 201473584-7

Distribución de archivos utilizando un Algoritmo Centralizado y un Algoritmo Distribuido

¿Qué se hizo y cómo se hizo?

Se realizó la implementación de un sistema de biblioteca donde los clientes pueden consultar por algún libro en específico para descargar, o bien, subir algún libro que deseen agregar al sistema de biblioteca. Los libros que serán subidos al sistema se dividirán en diferentes partes de 250[kB] cada uno y serán distribuidos en diferentes servidores. Se utilizaron 4 máquinas virtuales, las cuales simulan el comportamiento de las diferentes servidores que conforman el sistema distribuido: 1 máquina para el Namenode, encargado de mantener un archivo que posee la distribución de los archivos dentro del sistema, 3 máquinas para los Datanodes, los cuales tendrán almacenadas las diferentes partes de un libro según se les indique y 1 máquina para el Cliente (Encargado de subir o descargar algún libro), la cuál puede ser la misma que las 4 máquinas mencionadas anteriormente.

Para esto, el sistema fue implementado bajo dos métodos diferentes para realizar la distribución de los archivos dentro de un sistema distribuido, utilizando un algoritmo centralizado y un algoritmo distribuido. Posteriormente se calcularon los tiempos que tardaron ambos algoritmos en ejecutar el traspaso de mensajes. A continuación, se detalla lo realizado en cada uno de ellos.

Antes que todo, cabe mencionar que todo el traspaso de mensajes realizado entre los distintos Servidores/Datanodes/Namenode/Cliente se realiza utilizando ProtocolBuffer.

- **Algoritmo Centralizado/Subir Libro:** Se utilizaron 3 Datanodes encargados de almacenar las distintas partes de los libros subidos, 1 de estos será el encargado de, en un principio, recibir todas las partes del libro a subir, luego generará una propuesta, considerando una distribución equitativa entre todos los Datanodes asumiendo que todos se encuentran disponibles, se enviará un mensaje por cada sub-propuesta al Namenode. (el conjunto de estas corresponde a la propuesta completa) Cada uno de estos corresponde a que parte del archivo irá a cada servidor, de esta forma el Namenode evaluará si es posible de realizar dicha distribución. Para esto, consultará a todos los Datanodes si es que están disponibles para almacenar los archivos. Se consideraron 3 tipos de posibles fallos: el primero considera la capacidad de los Datanodes de tal forma que tenga el espacio necesario para almacenar la cantidad de partes que el Namenode le propone, el segundo considera un factor aleatorio de fallo del 10% y el tercero considera que el Datanode no se encuentre caído al momento de la solicitud. Los Datanodes le responden al Namenode con un mensaje indicando su capacidad actual y un “flag” que representa si se produjo un error aleatorio, de esta manera el Namenode puede saber si el Datanode se encuentra

disponible o no para recibir los archivos necesarios. Una vez que el Namenode posee la respuesta de parte de todos los Datanodes, verifica si alguno de estos no puede recibir la cantidad de archivos propuestos inicialmente, si este es el caso, genera una nueva propuesta en base a los Datanodes disponibles y se la envía de vuelta al Datanode principal, enviando un mensaje por cada sub-propuesta, si no existen problemas, le responde con la misma propuesta inicial. Finalmente, el Datanode principal al obtener la propuesta del Namenode se encarga de distribuir las partes del libro a los Datanodes restantes según corresponda.

- **Algoritmo Distribuido/Subir Libro:** Al igual que en el algoritmo anterior, el Cliente primeramente se contacta con un Datanode en específico (nunca cambia) al cual le envía un mensaje por cada parte (Chunk) del libro a subir. Una vez el Datanode recibe las partes del libro, este procede a realizar la misma propuesta inicial creada para el algoritmo anterior, para consultar con el resto de los datanodes (además de si mismo) si se encuentran disponibles para recibir el archivo. En caso de que alguno de estos falle, se procede a generar una nueva propuesta y a consultar con el resto de los datanodes disponibles la misma consulta anterior. Con la nueva propuesta definida, esta se envía al namenode para que este lo añada a su archivo log. Posteriormente se procede a repartir las partes en los datanodes presentes en la propuesta.

- **Algoritmo Centralizado - Distribuido/Descargar Libro:** Para ambos algoritmos, el Cliente se comunica con el Namenode para solicitar las partes del libro que sea descargar, este le envía un mensaje indicando que libro necesita y el Namenode responde con un mensaje por cada Chunk con su ubicación respectiva. Una vez con esta información, el Cliente le manda un mensaje por cada Chunk solicitado al Datanode que corresponda, este último le envía un mensaje con el contenido de cada Chunk solicitado. Finalmente el Cliente con todos los Chunks obtenidoa procede a juntarlos y obtiene el libro solicitado.

Resultados

A continuación, se presentan los resultados obtenidos utilizando 3 libros diferentes con ambos algoritmos, subidos por 1 cliente a la vez:

Tamaño Archivo	Cantidad Chunks	Algoritmo	N° Mensajes	Tiempo
5.4 [MB]	23	Distribuido	84	35.8 [s]

Resultados obtenidos utilizando el libro *War and Peace (Kindle with Images)*: obtenido de: https://www.gutenberg.org/ebooks/2600

Tamaño Archivo	Cantidad Chunks	Algoritmo	N° Mensajes	Tiempo
5.4 [MB]	23	Centralizado	104	36.4 [s]

Resultados obtenidos utilizando el libro *War and Peace (Kindle with Images)*: obtenido de: https://www.gutenberg.org/ebooks/2600

Tamaño Archivo	Cantidad Chunks	Algoritmo	N° Mensajes	Tiempo
561 [kB]	3	Distribuido	16	9.4 [s]

Resultados obtenidos utilizando el libro *A Christmas Carol (EPUB con imagenes)*: obtenido de : https://www.gutenberg.org/ebooks/46

Tamaño Archivo	Cantidad Chunks	Algoritmo	N° Mensajes	Tiempo
561 [kB]	3	Centralizado	17	10.2 [s]

Resultados obtenidos utilizando el libro *A Christmas Carol (EPUB con imagenes)*: obtenido de : https://www.gutenberg.org/ebooks/46

Tamaño Archivo	Cantidad Chunks	Algoritmo	N° Mensajes	Tiempo
1.0 [MB]	5	Distribuido	24	12.2 [s]

Resultados obtenidos utilizando el libro: *Emma (PDF)* obtenido de: https://www.gutenberg.org/ebooks/158

Tamaño Archivo	Cantidad Chunks	Algoritmo	N° Mensajes	Tiempo
1.0 [MB]	5	Centralizado	26	11.7 [s]

Resultados obtenidos utilizando el libro: *Emma (PDF)* obtenido de: https://www.gutenberg.org/ebooks/158

Análisis

Es posible notar que para los dos primeros casos, el algoritmo centralizado tardó más en ejecutarse que el algoritmo distribuido, también hubo una mayor cantidad de mensajes transmitidos en el centralizado en todos los casos. El último caso se dió que el algoritmo centralizado tardó menos en ejecutarse pese a haber mandado más mensajes que en el distribuido.

Discusión

Las diferencias de tiempo producidas entre un algoritmo y otro se deben principalmente a la cantidad de mensajes que participaron durante el proceso, esto queda claro en los dos primeros casos, si bien, no existieron diferencias de tiempo muy importantes entre ambos algoritmos debido a que se ejecutó un cliente durante cada proceso y no se produjo un cuello de botella en el Datanode que recibe los archivos a subir.

En promedio, al utilizar el algoritmo distribuido, se utiliza un número cercano al 80% de los mensajes que se usan al utilizar un algoritmo centralizado. Esto se debe a que en el modo distribuido, al momento de chequear la propuesta, el datanode con el que se comunica principalmente el cliente no recibe mensajes desde el datanode. Mientras que con archivos pequeños, las diferencias en la cantidad de mensajes son pequeñas, estas son más notorias con archivos más grandes.

Conclusión

En este informe se estudió las diferencias entre dos tipos de algoritmos para la distribución de archivos en un repositorio de libros distribuido. Basado en los resultados obtenidos después de realizar la experimentación con tres libros diferentes, se puede concluir que utilizar un algoritmo distribuido trae consigo ventajas en términos de un menor número de mensajes enviados, además de un menor tiempo de ejecución, en comparación al algoritmo centralizado.