Natalia Herrera,
natalia.herrera@sansano.usm.cl,
201673501-1

Sistemas Distribuidos
Laboratorio 2

Andrés Shehadeh
andres.shehadeh@sansano.usm.cl,
201673560-7

Desarrollo del Problema

Se implementó un sistema distribuido para un biblioteca que considera 4 máquinas: 3 DataNodes que que acumulan secciones (o chunks) de libros, y 1 NameNode que lleva el registro de qué chunks están en qué máquina. Los clientes pueden subir libros seccionados a este sistema o descargar las secciones y luego unirlas, acciones que pueden reaizar tanto desde los DataNodes como desde el NameNode.
La implementación de este sistema se hizo por medio de gRPC para comunicar a las máquinas con archivos .proto distintos: uno en que se consideraba a los DataNodes como servidores (por ejemplo para subir un libro al sistema) y otro en que se consideraba al NameNode como servidor (por ejemplo cuando el cliente quiere solicitar el log de los libros).

Resultados

Se ejecutó el programa 5 veces de forma centralizada y 5 veces de forma distribuida subiendo y descargando siempre el mismo libro (Drácula). Cabe destacar que al inicio de cada ejecución se borran los chunks de los libros que quedan en las máquinas de ejecuciones anteriores. Se midieron los tiempos y cantidad de mensajes para la descarga en ambos casos porque, a pesar de que no hay diferencias en el código para bajar un libro, podría darse alguna diferencia en la distribución de los chunks que cause que la descarga tome más tiempo o más mensajes para alguno de los algoritmos. Los resultados obtenidos fueron los siguientes:

a) Algoritmo Distribuido

Upload [ms]	Download [ms]
147.326047	325.991903
209.080806	505.149605
176.795491	293.03127
170.331768	408.359081
144.300497	349.873167

Table 1: Tiempos de subida y bajada para el libro Drácula con el algoritmo distribuido.

Con respecto al tiempo que se tarda en subir un libro y actualizar o crear el log, son 169.5669218ms en promedio. Para descargar el libro se tarda, en promedio, 376.4810052ms.

	Upload	Download
Cliente - DataNode	14	14
NameNode - DataNode	4	
DataNode - DataNode	78	
Cliente - NameNode		4

Table 2: Mensajes intercambiados entre los distintos tipos de máquinas en el proceso de subir y descargar el libro Drácula con el algoritmo distribuido.

Por otro lado, la cantidad de mensajes que se intercambian es un total de 96 mensajes para subir el libro y 18 mensajes para descargarlo, considerando tanto requests como replies.

b) Algoritmo Centralizado

Upload [ms]	Download [ms]
265.470747	352.949681
218.950773	339.914906
214.093504	342.719114
164.064697	434.174389
160.438152	456.998088

Table 3: Tiempos de subida y bajada para el libro Drácula con el algoritmo centralizado.

Con respecto al tiempo que se tarda en subir un libro y actualizar o crear el log, son 204.6035746ms en promedio. Para descargar el libro se tarda, en promedio, 385.3512356ms.

	Upload	Download
Cliente - DataNode	14	14
NameNode - DataNode	32	
DataNode - DataNode	14	
Cliente - NameNode		4

Table 4: Mensajes intercambiados entre los distintos tipos de máquinas en el proceso de subir y descargar el libro Drácula con el algoritmo centralizado.

Por otro lado, la cantidad de mensajes que se intercambian es un total de 60 mensajes para subir el libro y 18 mensajes para descargarlo, considerando tanto requests como replies.

Análisis

Empezando por los tiempos, se observa que el tiempo de subida (o upload) promedio es de aproximadamente 170ms con el algoritmo distribuido, cerca de un 80% del tiempo que toma subir en promedio un libro con el algoritmo centralizado, lo que corresponde a aproximadamente 205ms. Estos 35ms de diferencia se pueden atribuir a que el algoritmo distribuido no necesita que los DataNodes (encargados de cargar los chunks del libro en las máquinas) interactúen con el NameNode para consultar sobre posibles distribuciones de los chunks, ya que los DataNodes lo resuelven entre ellos.
Los tiempos de bajada (o download) solo presentan una pequeña diferencia de menos de 10ms de diferencia que se puede ignorar o atribuir a la variabilidad del uso de los recursos del computador, dado que no hay ninguna diferencia de implementación en el programa para la descarga de los libros.
Con respecto a la cantidad de mensajes intercambiados entre las máquinas, para el algoritmo distribuido son un total de 96 mensajes, mientras que para el algoritmo centralizado son 60, cerca de 2/3 de los mensajes del algoritmo distribuido. Esto se puede deber a que, como se puede ver en las tablas 2 y 4, el algoritmo distribuido implica que cada DataNode se tiene que comunicar con todos los otros al momento de trabajar las propuestas, mientras que en el algoritmo centralizado solo se tienen que comunicar con el NameNode, que se encarga de tomar decisiones y luego de transmitirlas al resto. Es importante notar que, aunque el algoritmo distribuido realiza más envíos de mensajes, estos mensajes también se reparten entre

más máquinas, por lo que el uso de recursos y las interrupciones no son necesariamente mayores.
Como ya se mencionó, no hay ninguna diferencia en el programa para las descargas, por lo que la cantidad de mensajes se mantiene igual en ambos casos.

Conclusiones

En conclusión, el algoritmo distribuido parece ser la mejor opción en términos de tiempo y, aunque la cantidad de mensajes intercambiados sea mayor, la cantidad de nodos que manejan estos mensajes también lo es, por lo que la carga de trabajo no es necesariamente mucho mayor. Por lo demás, si el problema escalara en magnitud, es decir, más DataNodes, el algoritmo centralizado empezaría a sufrir por cuellos de botella al tener solo un NameNode y aún más mensajes. Mientras, el algoritmo distribuido, mientras más DataNodes tenga, aunque aumente la cantidad de mensajes, los nodos siguen manejando más controladamente la carga.
Es importante mencionar que, aunque la diferencia en tiempos entre ambos algoritmos no sea tan considerable, estando en el orden de los 10ms, al escalar el problema, es de esperar que esta diferencia aumente considerablemente, reforzando así la idea de que, en ese caso, es mejor el algoritmo distribuido. Por otro lado, si lo que importa es la cantidad de mensajes más que el tiempo que tarda en ejecutarse el programa, entonces la mejor opción es el algoritmo centralizado.