Exclusión mutua centralizada y Exclusión mutua distribuida
¿Cuál es mejor?

Yoel Berant Elorza, Rol: 201604519-8
Diego Valderas Uribe, Rol: 201673549-6

Qué se hizo y cómo se hizo

En esta ocasión se desarrolló un sistema distribuido cuya finalidad es simular un servicio de biblioteca en el que los “libros” se guardarán como archivos distribuidos en distintas entidades.

El sistema consta de 3 tipos de agentes “agentes”:

- DataNode: Son los nodos encargados de repartirse las partes o “chunks” de los archivos que se suben. Cada vez que se quiera particionar y repartir un archivo, uno de los datanodes realiza una propuesta de distribución. En el caso de que el sistema sea “**excluyente distribuído**”, el resto de los datanodes se encargan de aprobar o rechazar las propuestas de distribución. En total son tres.

- NameNode: Es el nodo que toma registro de la ubicación de los chunks mediante un archivo “log” en el que se registra en qué datanode se encuentra cada chunk de cada libro subido. El namenode posee registro de esta información gracias a que cada vez que los datanodes se reparten un archivo, uno de los datanodes (el mismo que realizó la propuesta de distribución) le envía esta al namenode para que la ingrese al log; el ingreso a dicho registro y qué información es escrita en este estará monitorizada para que no se ingrese información de más de un archivo al mismo tiempo y la información no se corrompa. Para lograr esta funcionalidad se ocuparon dos enfoques; uno centralizado donde el namenode elige qué nodo ingresará; y otro distribuidos donde la elección de quién ingresa se decide entre los datanodes mediante el algoritmo de Ricart & Agrawala.
  Además, es la entidad que acepta o rechaza la propuesta de distribución en el caso de que el sistema sea “**excluyente centralizado**” (el caso opuesto a que sí es excluyente distribuído).

- Cliente: Este agente posee dos roles, el primero consiste en ser la entidad que sube los diferentes libros al sistema, poniendo estos a disposición de otros clientes interesados. El segundo rol corresponde a ser quien solicite la descarga de algún archivo entre los textos disponibles.

Para poder estudiar el comportamiento de los dos algoritmos implementados (exclusión mutua centralizada y exclusión mutua distribuida), se llevará a cabo un experimento en cada uno de estos con el fin de poder comparar y contrastar sus resultados.

El experimento consiste en, desde un cliente, subir el archivo “El maravilloso Mago de Oz” al sistema de la biblioteca. Se eligió este libro pues de los que se consiguieron, era el más pesado (más de 5 MB). Este proceso implica el envío de mensajes entre los distintos nodos del sistema a modo de coordinar la escritura en el registro, el veredicto final sobre la decisión de la propuesta de distribución y el envío de chunks. Aquí es que se pueden evidenciar las diferencias entre los algoritmos: para la exclusión mutua en el algoritmo

distribuido es necesario enviar mensajes entre los nodos para decidir si se puede acceder al log mientras que en su versión centralizada es el NameNode quien decide el acceso sin necesitar envío de mensajes. De todas formas, en caso de que se rechace una propuesta, tanto por un datanode como por el namenode, el datanode que realizó la propuesta tendrá que realizar una totalmente nueva

Para cada algoritmo el experimento es realizado cuatro veces. En cada una de estas se mide el tiempo que demora el algoritmo en subir el archivo, cuántos mensajes envía el datanode que realiza la propuesta (la que tiene una probabilidad de aprobación del 70%), cuántos mensajes envió el namenode y finalmente el total de mensajes enviados entre los datanodes y el namenode, sin considerar al cliente.

Se espera que mediante los resultados se evidencie que el mejor rendimiento corresponde al algoritmo centralizado ya que este realiza una menor cantidad de envíos de mensajes para lograr la coordinación, implicando con esto una menor cantidad de tiempo para llevar a cabo la tarea. Por otro lado, el algoritmo distribuido, al tener que llegar a un consenso con los otros nodos, debe entablar conversaciones con estos aumentando la cantidad de mensajes a enviar y por consiguiente un mayor tiempo de ejecución.

Resultados

A continuación se presentan dos tablas con los resultados correspondientes a los cuatro experimentos llevados a cabo en cada uno de los algoritmos. Las métricas son: tiempo de subida, mensajes enviados por el datanode encargado de realizar la propuesta, mensajes enviados por el namenode y mensajes enviados totales (enviados por los tres datanodes y el namenode, sin contar al cliente):

Resultados en algoritmo centralizado:

Iteración	Tiempo de Subida [milisegundos]	Mensajes enviados por DataNode de la propuesta	Mensajes enviados por NameNode	Mensajes Totales (datanodes y namenode)
1	133	27	2	33
2	149	27	2	33
3	144	49	3	56
4	180	27	2	33

Resultados en algoritmo distribuido:

Iteración	Tiempo de Subida [milisegundos]	Mensajes DataNode de la propuesta	Mensajes NameNode	Mensajes Totales
1	215	137	1	148
2	154	49	1	56
3	193	49	1	56
4	166	49	1	56

Para obtener los valores referentes a los mensajes enviados se emplearon contadores que aumentaban en una unidad cada vez que se enviaba un mensaje. Para obtener el tiempo de subida se hizo uso de la librería de go “Time”, donde se obtiene el tiempo al momento de empezar la ejecución y una vez esta es terminada, se procede a restar ambos tiempos para así obtener cuánto tardó la ejecución.

Análisis

Se observa claramente que la cantidad de mensajes enviados es mucho más alta en el caso del algoritmo distribuido que en el centralizado.

En cada algoritmo, en una de las 4 iteraciones, la cantidad de mensajes totales fue mucho mayor que en el resto. Con respecto al resto de iteraciones, la cantidad de mensajes enviados por el datanode que realiza la propuesta es cerca del doble en el caso distribuído que en el centralizado.

Con respecto a la cantidad de mensajes enviados desde el namenode, esta vez es mayor en el caso del algoritmo centralizado, aunque en ambos casos es bastante baja (sólo 3 mensajes o menos).

Con respecto al tiempo de subida, el algoritmo distribuido tiende a ser ligeramente mayor, si bien existe solapamiento

Discusión

Como se esperaba, el número de mensajes enviados fue mucho mayor en el caso distribuído, pues las propuestas de repartición de archivos tuvieron que ser aprobadas y, por tanto, enviadas a los otros dos datanodes en lugar de sólo al namenode, implicando a su vez más mensajes de respuesta sobre si dicha propuesta es aprobada (desde los otros datanodes, en este caso). No es de extrañar entonces que en el caso distribuído, el número de mensajes enviados desde el datanode que tomó la decisión es cerca del doble que en el caso centralizado.
En el caso de los mensajes enviados desde el namenode, tampoco es de extrañar que sea mayor en el caso centralizado, pues es en esta versión del algoritmo el namenode toma un mayor rol al ser la entidad que debe aprobar o rechazar la propuesta hecha. Por otro lado, la cantidad de mensajes en ambos casos sigue siendo bastante baja, lo cual es comprensible pues en la subida de archivos el namenode no necesita enviar más mensajes que un “OK”, o rechazar/aprobar una propuesta recibida en el caso centralizado.

Se puede ver también que en cada cuarteto de iteraciones hubo una en la que la cantidad de mensajes fué mayor. En estas ejecuciones cláramente sucedió que una propuesta de la distribución fue rechazada, por lo que se tuvo que crear y enviar otra, implicando más mensajes.

Considerando que en las dos versiones del algoritmo el porcentaje de aprobación fue de 70%, la probabilidad de que una propuesta se rechace en el caso centralizado es de 30%, pues sólo el namenode decide según esta probabilidad, y de 51% en el caso distribuido, pues es requisito que los otros dos datanodes aprueben, cada uno con probabilidad de 70% (por lo que para calcular la probabilidad de rechazo, solo se tiene que descartar el caso de que ambos aprueben, es decir: 100*(1-(0.7*0.7))=51%)

Era esperable que el tiempo de ejecución sea relativamente mayor en el caso de la versión distribuida, sin embargo, debido a la instancia y a la carga entregada al nodo en ambos escenarios (centralizado y distribuido), sumando al orden de magnitud en que fue medido cuánto tardó una ejecución, los algoritmos no difieren entre sí significativamente en cuanto a tiempo nos referimos. En un escenario realista donde millones de nodos buscan descargar información de gran tamaño distribuido en una mayor cantidad de nodos, el impacto que tiene la implementación de un algoritmo u otra es significativo. Es por esto que al momento de tener que decidir cuál de los algoritmos ocupar se requiere un estudio profundo del sistema y qué se quiere lograr con este, de modo que la opción escogida entregue el mayor valor posible al sistema.

Conclusión

A través de un simple experimento se demostró que, en el caso de esta implementación, la mejor versión del algoritmo es la centralizada, pues implica una cantidad menor de mensajes enviados (lo cual minimiza el riesgo de overhead) y un tiempo ejecución relativamente menor (aunque despreciable en ambos casos, considerando la magnitud del experimento).

Sin embargo, hay que dejar en claro que este fue un caso bastante particular, pues se creó un sistema distribuído bastante simple (5 entidades, contando al cliente) en menos de dos semanas. En los casos más “laborales”, los sistemas de almacenamiento distribuidos constan de cientos, quizás miles de servidores que se reparten gigabytes de archivos y se comunican entre sí de manera constante. Es en estos casos no se podría asegurar que un sistema basado en exclusión mutua centralizada es mejor que en exclusión mutua distribuida sólo basándonos en tiempos de respuesta y tráfico de mensajes. Por ejemplo, otra métrica muy valorada en sistemas distribuidos es la tolerancia a los errores. En un sistema centralizado como el implementado, es más difícil tolerar un error pues si el namenode falla, no queda ningún otro nodo que pueda asumir su rol de mantener el registro de los archivos. Por lo tanto, la versión centralizada, si bien puede parecer “más eficiente”, presenta el problema del “único punto de falla” y por eso no se puede decir que es mejor que la distribuida en todos los casos y vice-versa.

También es necesario recalcar que todas estas conclusiones se hicieron en base a la funcionalidad de subida de archivos del sistema y no la descarga. Esto se debe a que no hay mucha diferencia (medible) entre el algoritmo centralizado y el distribuído en cuanto a la descarga.