Informe Tarea 2, Sistemas Distribuidos

Integrantes: Sebastián Gutiérrez-201773006-4

Carlos Rojas M.-201773013-7

Explicación de lo hecho

Centralizado

Para el caso centralizado se partirá pidiendo al usuario la dirección del libro (los cuales estarán en la misma carpeta del programa Go), luego este libro pasara a partirse en partes de 250kb cada uno para ser enviados a un datanode elegido al azar, en caso de estar caído se reintentara a otro nodo aleatorio, así al llegar estos chunks a un Datanode disponible este generara aleatoriamente una propuesta la cual será enviada al NameNode el cual es nuestro encargado de comprobar el los nodos relacionados a la propuestas estén en un estado correcto para recibir los chunks, así este le pregunta su estado, si este estado es diferente a ok (tiene más de 200000 chunks) o este no responde , lo que implica que esta caído, este descarta ese nodo y cambia la propuesta, así este retorna al Datanode una propuesta correcta y escribe esta propuesta en su Log, así el Datanode que esta administrando los chunks usa la nueva distribución y envía los chunks a los respectivos Datanodes, los cuales lo reciben y crean un archivo de formato “NombredelLibro—OffsetChunk” en el cual se escribió el respectivo Chunk del libro, así se terminara el proceso de subida de un libro.

Para la bajada el cliente este introduce el nombre del libro a descargar junto a su extensión, esto llamara al Unchunking para ese libro el cual le enviara un mensaje a Namenode preguntando por las direcciones del respectivo libro, el cual las retornara al cliente las cuales las usara para preguntarle a cada DataNode por el respectivo Chunk que este tiene , a lo que este busca los archivos con el nombre del libro y envía su contenido hacia el cliente el cual es el encargado de rearmarlo y crear un nuevo archivo con el nombre del archivo +D para denotar que es el archivo descargado.

Distribuido

Para el caso de distribuido es parecido el comienzo a centralizado, se le pedí al usuario el archivo el cual será partido en chunks de 250kb los cuales serán enviados a un datanode aleatorio el cual se verificara previamente que no este caído, así este datanode producirá una propuesta la cual será comparada con sus pares para así revisar que esta no contenga una dirección de un DataNode caído o que no se encuentre en condiciones, así al obtener una propuesta correcta este DataNode enviara los Chunks según lo dice la propuesta y se enviara la propuesta al NameNode para que este escriba esta propuesta usada en el Log , para este pasó se usó el algoritmo “Ricart y Agrawala” ya que se considera al Log como un componente critico del sistema, por lo que antes de enviar la propuesta al código se realizara una comprobación los demás DataNode y sus estados en busca de algún DataNode que también busque escribir en el Log así se usaran Timestamp para comparar en caso de que 2 Datanode que quiera escribir se encuentren y el que tenga un Timestamp anterior en el tiempo sea el que pueda escribir primero en el Log, así finalmente al tener esto escrito en el Log se terminara el proceso de subida.

Para la bajada el cliente este introduce el nombre del libro a descargar junto a su extensión, esto llamara al Unchunking para ese libro el cual le enviara un mensaje a Namenode preguntando por las direcciones del respectivo libro, y se hará de igual forma que en la distribución centralizada.

Resultados obtenidos

Según lo visto en la ejecución de las rutinas el que logra subir los chunks y realizar todas las acciones de forma más rápida fue la parte “distribuida” ya que esta no usa un intermediario para organizar sus propuestas , pero en el caso del NameNode al tener que realizar mas comprobaciones y enviar una mayor cantidad de mensajes este escribe mas lento en este que nuestro versión de Centralizada ya que este escribe directamente sin usar un algoritmo que controle la escritura sobre el componente critico de Log, finalmente cabe destacar que ambos fueron muy buenos a la hora de enfrentarse a problemas tales como caídas de nodos.

Conclusiones

En los casos observados se puede apreciar que para mayor velocidad es conveniente usar la versión distribuida y así también tenemos la seguridad de controlar de buena forma un archivo critico como lo es un Log que puede ser compartido en un momento del tiempo, así ambos tipos de distribución nos ayudan a lidiar con los problemas básicos cuando se trabaja con sistemas distribuidos como lo es la caída de alguno nodo, cabe destacar que si bien resuelve el problema a la hora de subir, cuando se desea descargar un archivo al tener un sistema distribuido se genera el problema de que si un nodo esta caído esos archivos serán inaccesibles.