Big Data

Trabajo Práctico 2 - Grupo 1

De Santi, Matías(51051) - Homovc, Federico(50418) - Pereyra, Cristian(51190) - Pintos, Esteban(51048)

Introducción

El objetivo del Trabajo Práctico 2 era procesar un stream de datos de Twitter y en tiempo real generar estadísticas según hits de ciertas palabras claves que se encontraban en los mismos. De esta manera se podría analizar la mención de ciertos partidos políticos en Twitter durante al época electoral.

Flume

Se utilizó flume para poder detectar y encolar los tweets a medida que se iban depositando en un cierto directorio.

Desarrollo

A medida que los tweets iban llegando se iban colocando en una cola de mensajes ActiveMQ para ser luegos procesados. Para esto se creó un Productor de una cola de mensajes y se fueron encolando los tweets en formato json mientras iban llegando.

Debido a que nunca habíamos utilizando Flume, no fue tarea sencilla configurar el mismo integrándolo con ActiveMq aunque tampoco nos presentó grandes dificultades.

Problemas Encontrados

Si bien la configuración de ActiveMQ no fue algo trivial, no nos consumió una gran cantidad de tiempo. Tuvimos un problema por el cual no se encolaban los mensajes en la cola de mensajes pero fue rápidamente resuelto cambiando un parámetro al crear la misma.

Sin embargo, un problema que se nos presentó fue una excepción al momento de parsear el JSON del tweet. En muchos casos el mismo estaba incompleto y por lo tanto no podía ser procesado. Luego de hacer varios seguimientos del flujo de datos, nos dimos cuenta

que flume estaba parseando los tweets de manera incompleta. Es decir, obtenía cierta cantidad de caracteres y generaba un evento con estos y los restantes los utilizaba para generar un nuevo evento. Como no sabíamos por qué podía ocurrir esto recurrimos a la documentación de flume y allí encontramos que flume, por default, levanta 2048 caracteres. Dado que los tweets tienen en su mayoría más de 2048 caracteres, había problemas al procesar casi todos.

Para solucionar el problema simplemente agregamos la siguiente línea al archivo de configuración de flume: *a1.sources.r1.deserializer.maxLineLength = 9000*.

Storm

Se utilizó Storm para poder desencolar los mensajes y generar las estadísticas buscadas.

Desarrollo

A medida que se iban encolando los tweets mediante Flume en la cola de mensajes, utilizando Storm pudimos desencolarlos y generar las estadísticas buscadas. Para esto se creó una topología con Spouts y Bolt. Cada uno de los Spouts generaba un consumidor de la cola de mensajes y a medida que los mensajes iban llegando se emitían para poder procesarlos con el Bolt.

Luego en el Bolt utilizando el HashMap mencionado en la siguiente sección, se buscaron matches entre el texto recibido y los datos del hash y utilizando MySql se fueron insertando los resultados. El texto se obtuvo parseando el JSON que se emitía desde el Spout. En un principio la complejidad del algoritmo de búsqueda de matches era altísima, ya que había que recorrer por cada palabra del tweet, cada key del map y por cada uno de estos su lista de keywords, es por esto que cambiamos la forma en que se guardó la información, reduciendo la complejidad a la cantidad de palabras del tweet. El cambio que se realizó se menciona en el apartado de **HBase**

En un comienzo el parser de JSON se encontraba en el Spout y emitiamos únicamente el texto del tweet. Luego esto lo cambiamos debido a que en una futura implementación se podrían necesitar de otros campos del JSON, por esto decidimos emitir el JSON entero para poder parsear lo necesario en el Bolt.

Problemas Encontrados

Al igual que Flume, la configuración no fue algo trivial. Esto se debió más que nada a que no fue tarea sencilla poder loguear lo que iba sucediendo. Decidimos usar Log4J para poder loguear los eventos, pero fue un gran desafío poder encontrar los archivos donde se encontraban los logs. Esto se debió a que el log de Storm no es distribuido por lo tanto hubo que buscar los logs en el nodo donde se estaban ejecutando los workers.

Otro problema encontrado, dominado durante el desarrollo del trabajo práctico, fue entender cómo funcionaba Storm, qué servicios necesitábamos tener levantados para poder correr la topología, spouts y bolts.

HBase

En la tabla *itba_tp2_twitter_words* se encontraban lo diferentes partidos políticos como Row key y las keywords para cada uno.

Desarrollo

La primera implementación que se realizó fue un parseo de la tabla mencionada anteriormente y la generación de un HashMap. El mismo contenía los partidos políticos como key y una lista de keywords como valor.

La segunda implementación que se realizó fue guardar los datos de una manera diferente para realizar de una manera más eficiente la búsqueda de matches. Para esto se guardó como key cada uno de los diferentes keywords y como valor el partido político al que correspondía.

Luego el HashMap generado se lo asignaba a la Topología correspondiente.

Debido a que utilizamos HBase para el Trabajo Práctico 1, no hubo gran dificultad para parsear la información a pesar de que la misma se encontraba guardada de diferente manera.

MySql

Se utilizó MySql para poder volcar los datos generados en los Bolts de Storm y así poder luego generar las estadísticas.

Desarrollo

Para poder volcar los datos se configuró un ConnectionManager con JDBC y a través de PrepareStatements se fueron volcando los datos. Para esto se creó previamente la tabla *party* con las columnas *Name* y *Quantity*, donde en la primera se insertaron los diferentes partidos políticos y en la segunda la cantidad de matches. Para esto cuando una keyword matcheaba, si el nombre del partido político no existía, se lo insertaba como (Partido, 1), y luego se iba aumentando en 1 por cada match.

El desarrollo no trajo complicaciones debido a nuestros conocimientos tanto de SQL como de JDBC.

Tests

Para poder garantizar que la lógica de nuestra implementación era correcta, utilizamos tests con PowerMockito para validar el comportamiento de los métodos más relevantes. Esto nos permitió refactorizar, por ejemplo, la forma en la que se conectaba a la base de datos MySQL sin problema alguno.

Conclusiones

La implementación del trabajo práctico nos permitió conocer más en profundidad nuevas tecnologías de procesamiento de información en tiempo real. Pudimos entender el flow de la información y la importancia de cada tecnología que entra en juego. También pudimos ver que no es tarea sencilla la configuración e integración de cada una de las mismas.