Map Reduce Fest

Bloom Fiter - HBase

Me baje el tar.gz de hbase, version 0.94.7 y lo descomprimi dentro de la carpeta $HADOOP_HOME/hbase.
Posteriormente, cree la tabla que guarda la información de userId – reputación.

> hbase shell
> create 'user_table', 'data'
> exit

Luego, para que las librerias de HBase sean "vistas" por hadoop, se deben agregar al classpath, modificando el archivo *conf/hadoop-env.sh:*

export HBASE_HOME="/Users/marbarfa/Documents/Development/tools/hadoop-1.0.4/hbase"
export HADOOP_CLASSPATH="$HADOOP_CLASSPATH:$HADOOP_CLASSPATH/lib:$HBASE_HOME/hbase-0.94.7.jar:$HBASE_HOME/lib/zookeeper-3.4.5.jar:$HBASE_HOME/conf:$HBASE_HOME/lib:$HBASE_HOME/lib/guava-11.0.2.jar:$HBASE_HOME/lib/protobuf-java-2.4.0a.jar"

Posteriormente, ejecuto el Driver. El Driver lee el archivo *users.xml* pasando su ubiación como parámetro y crea por cada usuario, un registro en la tabla 'users_table', familia 'data', qualifier 'reputation', rowId = userId.
Además, el Driver también crea el BloomFilter de todos los usuarios con reputación mayor o igual a 1500.

El algoritmo MapReduce, lee posts de stack overflow y filtra incialmente con el bloomFilter todos aquellos comentarios que el usuario es mayor a 1500, luego, en caso de pasar por el BloomFilter, se consulta a la base de datos.
El BloomFilter ayuda a minimizar consultas a la base de datos.

Para verifiar si los datos han sido ingresados correctamente a la base de datos se puede realizar una simple consulta:

> hbase shell
> scan 'user_table'

En este caso, la fase de “entrenamiento” es importante ya que crea la tabla con los datos de los usuarios y crea el bloomFilter con las reputaciones de los usuarios, el cual será utilizado para minimizar las consultas a la base de datos.

Las consultas a HBase son costosas en comparación a una consulta a BloomFilter. Se realizó una ejecución de los mismos datos sin utilizar el BloomFilter obteniendo una diferencia en tiempos de ejecución, pero no considerable. Pienso que si puede ser significativo a medida que crecen los datos. La cantidad de datos que tiene la base son pocos por lo que las consultas a la misma no hacen una diferencia significativa, pero en un caso mas realista, esto no es así. Asumiendo que la tabla a consultar es mucho mas grande, la utilización de BloomFilter para reducir el número de consultas que se realizan a la base, puede repercutir en diferencias en tiempos de ejecución considerables.

Tiempo de ejecución utilizando BloomFilter: 108 segundos.
Tiempo de ejecución **no** utilizando BloomFilter: 127 segundos.