Análisis de Datos Geoespaciales de Tweets usando Hadoop y HBase

Junior Usca Huacasi

Escuela Profesional de Ciencia de la Computación

Universidad Nacional de San Agustn

A thesis submitted for the degree of

Licenciado en Ciencia de la Computación

Arequipa 2017

This thesis is dedicated to
someone
for some special reason

Acknowledgements

plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle.

Abstract

La gran cantidad de datos que producen las redes sociales como twitter, permiten a las empresas poder realizar operaciones como: almacenamiento, administración, y manipulación de datos. Para realizar este análisis de datos se requiere de sistemas distribuidos los cuales permitiran trabajar con cientos de miles de datos. Hadoop es un framework ideal para estos trabajos que funcionan sobre computadoras de bajo costo, pero existe el problema de tratar los datos en raw, por lo que se requiere un base de datos que pueda almacenar gran cantidad de datos en tiempo real como es HBase, una base de datos NoSQL orientada a columnas o familia de columnas. Se propone adaptar hbase como una base de datos geográfica, almacenando las geolocalizaciones de cada usuario y realizando el pre procesamiento como mean, median y midpoint, para optimizar algunos de los algoritmos más comunes en SIG(Sistema de Información Geográfico). Aprovechando el procesamiento MapReduce de Hadoop para realizar el KNN(K nearest neighbor), que es más rápido por los previos procesamientos y una adecuada configuración de hadoop y hbase.

Contents

List of Figures

Chapter 1

Introducción

Los datos estan creciendo de manera exponencial, un ejemplo de ello son las redes sociales como twitter que tiene 256 millones de usuarios es decir el 80 porciento de las personas en el mundo [6], generando al día cientos de miles de datos.

Esta gran cantidad de datos les es util a las empresas para poder gestionar sus negocios y saber en que lugar se encuentran sus seguidores, un sistema que realiza el almacenamiento, manipulación y administración de los datos es GIS (Sistema de información geográfico). Este sistema integra, almacena, edita, analiza, comparte y muestra información geográfica. Las aplicaciones SIG son herramientas que permiten a los usuarios crear consultas interactivas (búsquedas creadas por el usuario), analizar información espacial, editar datos en mapas y presentar los resultados de todas estas operaciones [8].

Para poder realizar el análisis de datos de Twitter en un espacio Geo-Espacial usaremos Apache Hadoop, este es un framework que permite el procesamiento distribuido de grandes conjuntos de datos a través de clusters de computadoras usando modelos de programación [2].

Hadoop tiene un modelo de programación que es MapReduce, el cual es un modelo de programación que procesa y genera grandes conjuntos de datos con un algoritmo paralelo distribuido en un cluster [1]. Un programa MapReduce se compone de un método Map que realiza el filtrado y la clasificación y un método Reduce que realiza una operación de resumen.

Apache HBase necesita acceso aleatorio, en tiempo real de lectura o escritura a gran cantidad de datos. El objetivo de este proyecto es el alojamiento de tablas muy

grandes de miles de millones de filas por millones de columnas sobre clusters de hardware [3]. Apache HBase es una base de datos open-source, distribuida, no relacional modelada después de Bigtable de Google. Así como Bigtable aprovecha el almacenamiento de datos distribuidos proporcionado por el sistema de archivos de Google, Apache HBase proporciona capacidades similares a Bigtable en la parte superior de Hadoop y HDFS.

En este paper se utilizará HBase para almacenar datos espaciales, como se vio HBase trabaja sobre Hadoop por lo que para el proceso de MapReduce, se usará el centro medio, mediano y geométrico, finalmente se obtendrán estos puntos y se hara el proceso de MapReduce para KNN.

Chapter 2

Trabajos Relacionados

La gran cantidad de datos que existen en el internet, van creciendo en cientos de miles de gigas, por lo que un análisis de estos datos sería muy importante y de gran ayuda para las empresas en la toma de decisiones respecto a sus clientes. Para lo cual existen herramientas para su procesamiento, pero este debe ser de manera distribuida puesto que sería la mejor manera de manejar gran cantidad de datos.

Una de las herramientas frecuentemente utilizadas para el procesamiento distribuido es Hadoop, el cual acelera el procesamiento de datos usando MapReduce, estes es es un modelo de programación basado en una clave y un valor, y una implementación asociada para el procesamiento de conjuntos de grandes cantidades de datos [7].

Las uniones espaciales en MapReduce se estudian en [9]. Los autores presentan el algoritmo SJMR (Spatial Join with MapReduce) que incluye un algoritmo de barrido de planos basado en bandas, esta función de partición espacial esta basada en mosaicos y tecnologías de evitación de duplicación para realizar ensamblaje espacial en MapReduce. La evaluación del rendimiento del algoritmo SJMR sobre los conjuntos de datos del mundo real muestra la aplicabilidad de MapReduce para aplicaciones espaciales intensivas de datos en pequeños grupos.

Estas investigaciones recientes de manejo y análisis de datos espaciales son manejadas por el MapReduce, claro que con limitantes para el análisis de datos, por lo que se propone usar una base de datos que pueda almacenar grandes cantidades de datos, como es HBase que corre sobre Hadoop.

Chapter 3

Marco Teórico

3.1 Bases de Datos Geo-espaciales

3.1.1 PostGis Replicado

PostGIS es una extensión de PostgreSQL que agrega soporte para datos geoespaciales y análisis [5] Es un proyecto de código abierto bajo la Open Source Geospatial Foundation (OSGeo) e independiente del proyecto PostgreSQL. A pesar de otro nombre similar, OSGeo es distinto y no está relacionado con OGC y OSM. Sin embargo, cada una de estas organizaciones contribuye al mismo ecosistema de software libre de código abierto para Geospatial (FOSS4G), que es también el nombre de una conferencia anual organizada por OSGeo [4] PostGIS utiliza las funciones simples estándar de OGC para objetos GIS5, así como las funciones que operan sobre ellos y los metadatos compatibles con OGC en los sistemas de referencia espacial (SRS). El índice espacial PostGIS es un árbol R sobre GiST (árbol de búsqueda generalizado) [5]

3.1.2

Chapter 4

Metodología

La metodlogía a utilizar, se realizará calculando la media geográfica, el punto medio y la mediana. Se usa un nuevo algoritmo para el cálculo de la mediana geométrica que inicia la iteración con un punto inicial y reduce los pasos de iteración total. Estos tres indicadores geográficos son estimadores importantes para resumir los patrones de distribución de localización en SIG. Por ejemplo, podría ayudar a estimar el espacio de actividad de una persona con mayor precisión.

4.1 Media Geográfica

La idea principal de la media geográfica es calcular un punto medio de latitud y longitud para todas las ubicaciones. La ecuación 1 muestra los pasos básicos de cálculo.

$$\begin{aligned} Lat &= \sum_{i=1}^{n} lat_i/n \\ Lon &= \sum_{i=1}^{n} lon_i/n \end{aligned} \qquad (1)$$

4.2 Punto medio geográfico

El punto medio geográfico (también conocido como el centro geográfico, o centro de gravedad) es la coordenada media de un conjunto de puntos en una tierra esférica. Inicialmente, la latitud y la longitud de cada localización se convierten en coordenadas cartesianas tridimensionales después las cambiamos a radianes. Luego calculamos la media aritmética ponderada de las coordenadas cartesianas de todas las ubicaciones

(se utiliza 1 como peso por defecto). Después de eso, la coordenada promedio tridimensional se cambia de nuevo a la latitud y longitud en grados.

4.3 Mediana Geográfica

Para calcular la mediana geográfica de un conjunto de puntos, necesitamos encontrar un punto que minimice la distancia total a todos los demás puntos. Un problema es encontrar el punto óptimo. La implementación de este algoritmo es un poco más compleja que la media geográfica y el punto medio geográfico, donde el punto óptimo se aproxima iterativamente.

Input: Location set $S = \{(lat_1, lon_1), ..., (lat_n, lon_n)\}$
Output: Coordinates of the geographic median
1: Let $CurrentPoint$ be the geographic midpoint
2: $MinimumDistance =$
$totalDistances(CurrentPoint, S)$
3: **for** $i = 1$ to n **do**
4: $distance = totalDistances(location_i, S)$
5: **if** $(distance < MinimumDistance)$ **then**
6: $CurrentPoint = location_i$
7: $MinimumDistance = distance$
8: **end if**
9: **end for**

Figure 4.1: Algoritmo de la Mediana [10]

En el algoritmo anterior el CurrentPoint será el punto medio geográfico que es el punto inicial, y dado la MinimumDistance, es la suma de todas las distancias del CurrentPoint hasta todos los demás puntos, para encontrar un punto inicial se cuenta la distancia total de cada lugar a otros, si uno de estos tiene una distancia menor que el CurrentPoint, entonces reemplazamos el CurretPoint por MinimumDistance.

Para calcular la distancia se usa la Ley de Cosenos, si las distancias entre cada par de puntos son pequeñas, entonces se debe usar la distancia entre dos puntos para reemplazar el arco circular.

$$distance = \arcsin(\sin(lat_1) * \sin(lat_2) + \cos(lat_1) * \cos(lat_2) * \cos(lon_2 - lon_1)) \quad (2)$$

4.4 KNN

KNN es un método para clasificar objetos, se basan en las distancias más cercanas como la Distancia Euclideana y la Distancia de Manhattan. KKN es un módulo importante en las redes sociales, que ayuda a encontrar a los usuarios más cercanos.

El siguiente algoritmo muestra el proceso de MapReduce en Hadoop. Para realizar la función Map usa uno de los valores que se ha procesado previamente (punto medio, la media y la mediana) estrayéndose el punto para su búsqueda, luego se calcula la distancia a cada punto y se envía los k puntos superiores, luego se aplica la función Reduce, la cual enviará los vecinos más cercanos a HBase.

function: Map(k,v)
1: $p = context.getPoint()$
2: **for** each cell c in value.rawCells() **do**
3: **if** column family is "coordinates" **then**
4: **if** qualifier is "minipoint" **then**
5: $rowKey = c.row()$
6: $location = c.value()$
7: $distance = calculateDistance(location, p)$
8: **end if**
9: **end if**
10: **end for**
11: emit (1, $rowKey$ + "," + $distance$)
function: Combine, Reduce(k,v)
12: $K = context.getK()$
13: **for** (each v_i in v) **do**
14: $(rowKey_i, distance_i) = v_i.split(,)$
15: **end for**
16: Sort all users by distances, and choose K smallest distance locations to emit;

Figure 4.2: Algoritmo de MapReduce para Knn [10]

Chapter 5

Experimentos Previos

Los experimentos se realizaron con un dataset de 22 GB de entrada, en un cluster de 4 nodos, cada nodo es una computadora Intel core i7 de 8GB de RAM, con sistema operativo Ubuntu 14.04, Hadoop 2.7 y HBase 1.2.

Realizamos una configuración adecuada para este cluster con lo siguiente:

- HBase maneja el pre-splitting de regiones automáticamente, creando una tabla con muchas regiones. Existe un problema con la pre-división al calcular los puntos de división para la tabla, se puede utilizar la RegionSplitter, la cual crea los puntos de división, utilizando un algoritmo de división HexStringSplit o UniformSPlit.

  Para Hadoop se realiza el cambio del tamaño de los bloques para que sea de 256MB, por defecto son de 64MB; este cambio se realiza por que se tiene que procesar grandes cantidades de datos, Hadoop al momento de realizar el MapReduce crea tantos map como bloques de datos existen, una gran cantidad de bloques de datos de tamaño pequeño genera un cuello de botella con muchos maps y una cantidad muy pequeña de bloques de datos no es bien aprovechada por el cluster.

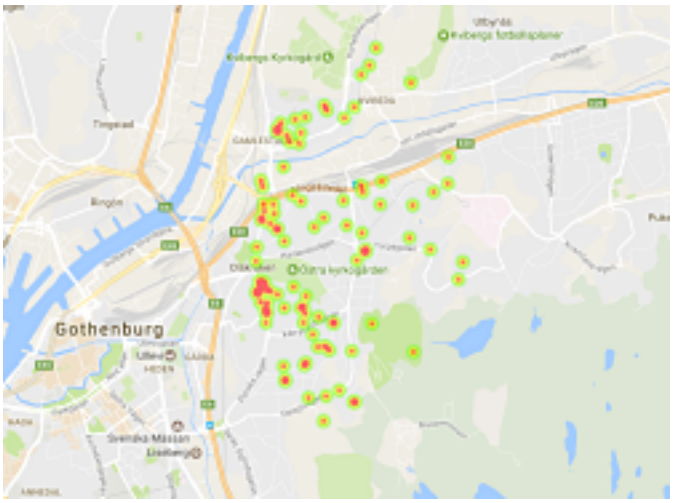
- Se modifica también las regiones en HBase con un tamaño de 256MB, siendo por defecto 64MB, esto se realiza debido a que HBase contiene Familias de Columnas las cuales van a ir incrementándose según la cantidad de datos que se

vayan insertando, al ser una gran cantidad de datos las que se manejan en redes sociales, debemos tener un mayor tamaño de almacenamiento de los mismos.

- Gran parte del procesamiento tanto en Hadoop como en HBase se realiza en memoria para hacer procesamientos de mayor tamaño se tiene que incrementar la memoria del JVM(Java Virtual Machine) a 1GB.

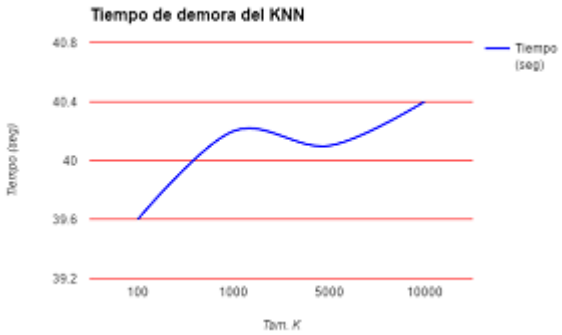
Luego de realizar estas configuraciones se grafica a los k usuarios más cercanos en un mapa de calor.

Se realizaron las pruebas usando 22GB de datos, para procesar los KNN con K= 10,50, 100 y 1000, con un tiempo de . Esto aplicando el punto medio



También se realizaron pruebas con la mediana geográifca, obteniendo puntos más adecuados para un análisis espacial.

La contribución preliminar que llega esta tesis es desarrollar un rápido sistema de análisis de datos espaciales utilizando el algoritmo de Mapreduce KNN Optimizado.



Chapter 6

Conclusiones

En este paper se realiza algunos algoritmos comunes para un Sistema de Información Geográfica para un manejo de Análisis de Datos Espaciales. Se uso HBase para almacenar datos espaciales, y se aprovechó MapReduce de Hadoop para el procesamiento de tweets, los cuales contienen las posiciones de los usuarios.

Se aceleró el proceso del algoritmo KNN, almacenando el cálculo del centro de la mediana, punto medio y la media. Para ser un análisis espacial más eficiente se configuró Hadoop y HBase para optimizar el proceso de distribución, almacenamiento y cálculo.

En nuestras pruebas se obtuvo tiempos significativos al momento de realizar el cálculo del algoritmo knn.

Como trabajos futuros se espera que se realicen más algoritmos orientados a análisis de datos espaciales.

Bibliography

[1] Ahmed; Alghamdi Rami; Mokbel Mohamed F. Alarabi, Louai; Eldawy. Tareeg: A mapreduce-based web service for extracting spatial data from openstreetmap. 2014.

[2] Apache. Hadoop.

[3] Apache. Hbase.

[4] FOSS4G. Osgeo: the open source geospatial foundation. foss4g.

[5] PostGIS. Postgis project steering.

[6] Business Twitter. Twitter.

[7] Jizhong; Tu Bibo; Dai Jiao; Zhou Wei; Song Xuan Wang, Kai; Han. Accelerating spatial data processing with mapreduce. 2010.

[8] Wikipedia. Geographic information system.

[9] Jizhong; Liu Zhiyong; Wang Kai; Xu Zhiyong Zhang, Shubin; Han. Sjmr: Parallelizing spatial join with mapreduce on clusters. 2009.