Desarrollo de Aplicaciones Big-Data 2024-2025. Proyecto final.

Sergio Hernandez
EHU-UPV
shernandez053@ikasle.ehu.eus

Yeray Li
EHU-UPV
yli003@ikasle.ehu.eus

Yeray Carretero
EHU-UPV
ycarretero001@ikasle.ehu.eus

Abstract

En este trabajo se analizarán y se expondrán distintos datos extraídos de un conjunto de datos sobre tipo de ocupación de viviendas en España junto con su consumo eléctrico. A su vez, se explicará cómo se han obtenido dichos datos mediante el algoritmo Map-Reduce. Por otro lado, se ha desarrollado una aplicación que calcula en tiempo real cuánto ha avanzado la estación internacional en un rango de tiempo de 30 segundos.

1 Introducción

El problema de la vivienda es una preocupación que va en aumento hoy en día. Cada día la vivienda es más inasequible tanto para los bolsillos de clase media como para los de clase baja, sobre todo en las grandes urbes donde la vivienda escasea y, debido a esto, muchas personas están siendo obligadas a abandonar la ciudad o compartir pisos entre 3 o más personas para poder seguir viviendo en la ciudad.

La situación en ciertos puntos de España está empezando a volverse crítica y cada vez es más necesario poner medidas para paliar esta problemática. Para ello, se necesita tener un gran conocimiento sobre cuáles son los mayores factores que están afectando a la gran escasez y precios elevados de la vivienda española.

En este trabajo se han propuesto 4 algoritmos de Map-Reduce para poder extraer datos interesantes que puedan ayudar a focalizar y entender mejor el problema de la vivienda en España, además de poder tomar mejores decisiones a la hora de legislar o proponer soluciones sobre este tema.

Por otro lado, se ha calculado la cantidad de kilómetros que se mueve la Estación Espacial Internacional(ISS). Para ello, se ha utilizado un entorno de procesamiento en tiempo real en el que se recogen los datos para después procesarlos y calcular su desplazamiento.

2 Datos

Los principales datos que se han utilizado para realizar los algoritmos de Map-Reduce han sido obtenidos del Ministerio para la Transformación Digital. Estos datos contienen 57.000 líneas de información sobre el estado de ocupación junto con el consumo eléctrico de municipios de más de mil habitantes de España. En estos datos aparece el código postal de los municipios junto a su nombre. En la siguiente columna aparece el tipo de dato que es, es decir, si habla sobre el tipo de la vivienda (viviendas vacías, de uso esporádico o totales) o el margen de electricidad. Por último, en la última columna aparecen el número de viviendas que el criterio del segundo tipo de dato, es decir, número de viviendas que consumen una franja de electricidad que normalmente los márgenes son de un tamaño de 1.000 Kw/h o número de viviendas que están vacías o que son de uso esporádico o vacacional. Las viviendas que se catalogaron de uso esporádico son aquellas que consumen entre 251 y 750 Kw/h.

Para la segunda parte del trabajo en la que se quiere calcular la traslación de la Estación Espacial Internacional cada 30 segundos, se utilizará la API de Notify API que es un proyecto de código abierto para solicitar cierta información de la NASA como las coordenadas de la estación ISS. Por tanto, el propósito de esta parte será calcular el desplazamiento de la ISS mediante las coordenadas proporcionadas por esta API.

3 Entorno de Ejecución

3.1 Cloudera

Cloudera es una empresa que ofrece una plataforma de software basada en Apache Hadoop y otras herramientas del ecosistema Big Data. Por tanto, es una plataforma beneficiosa para realizar el trabajo de procesar algoritmos de Map-Reduce y de procesamiento en tiempo continuo. Cloudera propor-

ciona muchas herramientas para procesar aplicaciones de Big Data:

- **Hadoop HDFS y MapReduce**
- **Apache Spark**
- **Apache Hive**
- Y otras muchas herramientas.

3.2 Hadoop MapReduce

Cloudera y Hadoop MapReduce están estrechamente relacionados, ya que Cloudera proporciona una plataforma que facilita el uso, la gestión y la escalabilidad de tecnologías de procesamiento de datos distribuidos como Hadoop. En particular, Hadoop MapReduce es uno de los componentes fundamentales incluidos dentro de la distribución de Cloudera, conocida como CDH (Cloudera Distribution including Apache Hadoop). MapReduce es el modelo de programación que permite procesar grandes volúmenes de datos mediante tareas distribuidas en paralelo, y Cloudera ofrece una infraestructura robusta para ejecutar estos procesos de manera eficiente y segura.

Nosotros hemos querido hacer de un uso más moderno de Hadoop MapReduce, por lo que hemos usado la estructura Context, a la vez que nos permitía escribir las clases de Mapper y Reducer en un mismo archivo, facilitando la estructura del proyecto. A su vez hemos automatizado el proceso de construcción del archivo JAR, de su ejecución y de la recolección de los datos. Esto nos permite ser más flexibles con el IDE (VIM) y no tener que depender de Eclipse a la hora de programar y de ejecutar el proyecto.

3.3 Apache Spark

Apache Spark es una plataforma de procesamiento distribuido en memoria, lo que significa que puede ejecutar operaciones sobre grandes conjuntos de datos mucho más rápido que tecnologías anteriores como Hadoop MapReduce, que escriben y leen en disco constantemente.

Aparte de ser mucho más rápido que Hadoop MapReduce, Apache Spark tiene la capacidad de poder procesar datos en tiempo real mediante la librería de Apache Streaming en Python.

3.4 Apache Flume

Apache Flume es una herramienta de código abierto diseñada para recoger, agregar y transportar grandes cantidades de datos desde múltiples fuentes hacia un sistema de almacenamiento como podría ser HDFS. Apache Flume actúa como un canal de ingesta de datos perfecto para transportar grandes cantidades de datos, como desde una página web, para después ser procesados por Hadoop MapReduce o Apache Spark.

El flujo de un agente de Flume funciona de la siguiente manera:

- **Source**: El lugar donde el agente recoge los datos.
- **Channel**: El intermediario que almacena temporalmente los datos hasta que estos se puedan mandar.
- **Sink**: El destino final de los datos, por ejemplo, HDFS, HBase...

4 Algoritmos

4.1 Map-Reduce

Map-Reduce es un algoritmo de programación diseñado para procesar y generar grandes cantidades de datos de forma distribuida y paralela. Este algoritmo se fundamenta en dos partes principales.

- **Map**: Cada nodo del sistema toma una línea del conjunto de datos y lo procesa para generar elementos clave-valor.
- **Reduce**: Todos los elementos clave-valor con la misma clave se agrupan, y se aplican funciones que resumen o combinan esos valores.

Para analizar el conjunto de datos, se han realizado los siguientes algoritmos.

4.1.1 Municipios con Mayor Porcentaje de Viviendas Vacías

Este algoritmo genera el porcentaje de viviendas vacías que tienen todos los municipios que se encuentran en nuestro conjunto de datos; después, se cogen los 10 pueblos más despoblados en porcentaje y se muestran en un gráfico para analizarlos.

Las partes de este algoritmo son las siguientes:

- **Map**: En cada elemento del conjunto de datos se utiliza el nombre del pueblo como clave. Para el valor, tan solo se seleccionan las lineas que contienen e número total de viviendas y el número de viviendas vacías, los demás se descartan. También se pasan a formato de texto y se le añade una "v" si el valor son las viviendas vacías o una "t" si son las totales.

- **Reduce**: En esta parte se divide el número de viviendas vacías de un mismo pueblo con el número de viviendas totales y se devuelve ese valor junto con el nombre del pueblo.

4.1.2 Municipios con el Mayor Número de Viviendas de Uso Ocasional

Este algoritmo calcula los diez municipios con mayor número de viviendas de uso ocasional. Hablamos en términos absolutos.

- **Map**: Cada proceso de mapper se encarga de seleccionar los 10 municipios con mayor número de viviendas y pasarlos al reducer. Cómo cada mapper no va a ver todos los registros lo que hace en realidad es seleccionar los 10 municipios de una forma "local".

- **Reduce**: El reducer recibe más de 10 municipios, así que su trabajo es de agrupar todas las salidas del mapper y realizar el filtrado final que se encargará de seleccionar el resultado definitivo.

Para poder hacer un ranking de municipios de manera cómoda nos ayudamos de la estructura de datos TreeMap, que se encuentra definida en Java SE 8.

4.1.3 Número de Viviendas Vacías en Cada Provincia

En este algoritmo se genera el número total de viviendas vacías por cada provincia. Las partes de este algoritmo son las siguientes:

- **Map**: Para la clave se utiliza los dos primeros números del código postal, que son los que definen la provincia a la que pertenecen, como clave. Después, si esa línea tiene el tipo de valor de las viviendas vacías, se pasa el número de casas vacías como valor. Si no, no se devuelve nada.

- **Reduce**: Se juntan todos los elementos clave-valor con misma clave que son los que pertenecen a una misma provincia y se suman todos sus valores que son el númeor de casas vacías por provincia.

4.1.4 Estimación de la Energía Total Consumida por Provincia

Este algoritmo calcula el número total de energía consumida por cada provincia.

- **Map**: Para la clave se utiliza los dos primeros números del código postal. Después, si en esa línea se muestra un rango de electricidad se saca la media de ese rango y se multiplica por el número de casas para pasarlo como valor.

- **Reduce**: Se suman los elementos con misma clave y se devuelve junto con el número de la provincia.

- **Nota**: En este algoritmo se calcula la estimación asumiendo que la suma de todos los consumos eléctricos de todas las viviendas entre un rango es la media de ese rango.

4.2 Cálculo de Desplazamiento de la ISS

Para calcular el desplazamiento de la ISS en tiempo real se ha utilizado tanto Notify API como Apache Flume y Apache Spark.

4.2.1 Extracción de Datos

Para extraer de forma correcta los datos para después poder procesarlos, se ha utilizado un programa de Python que extrae cada dos segundos la latitud y longitud de la estación que después se guarda en un fichero.

4.2.2 Agente Flume

Una vez extrayendo la longitud y latitud de la estación cada dos segundos, es necesario crear un agente de Apache Flume que transporte los datos de un log a HDFS donde los datos se podrán procesar de una forma eficiente. Los componentes del agente se han configurado de la siguiente manera:

- **Source**: Ejecuta el comando "tail -F /ruta/del/fichero" constantemente para leer los datos nuevos que llegan del log.

- **Channel**: Almacena los datos temporalmente en memoria. Tiene de la capacidad de almacenar 1000 datos simultaneamente y escribir 100 datos por cada transacción

- **Sink**: Escribe los datos en el sistema de archivos HDFS.

4.2.3 Cálculo de la Distancia Recorrida

Una vez transportados los datos, estos se han de procesar. Para ello, se han eliminado las fechas en las que se han extraído estos datos. A veces, las peticiones a la API generan errores de conexión que se escriben después en el fichero de log. Para evitar que el programa que calcula la distancia de

error al pasar un mensaje de error de conexión, existe un preprocesado en el que, una vez se ha dividido la entrada mediante el carácter "-" se filtra para poder saber que lo que se está procesando no es un mensaje de error.

Para calcular la distancia de la tierra se ha creado un procesamiento en tiempo real mediante la librería de Spark Streaming en Python. Se ha establecido que la duración de la ventana será de 30 segundos, es decir, cada 30 segundos se recopilarán todos los datos. Después, se coge el primer dato y último dato que se ha recopilado y se para por la fórmula para calcular la distancia de Haversine:

$$2r \arcsin \sqrt{\sin^2\left(\frac{\Delta\phi}{2}\right) \cos\left(\phi_1\right) \cos\left(\phi_2\right) \sin^2\left(\frac{\Delta\lambda}{2}\right)}$$

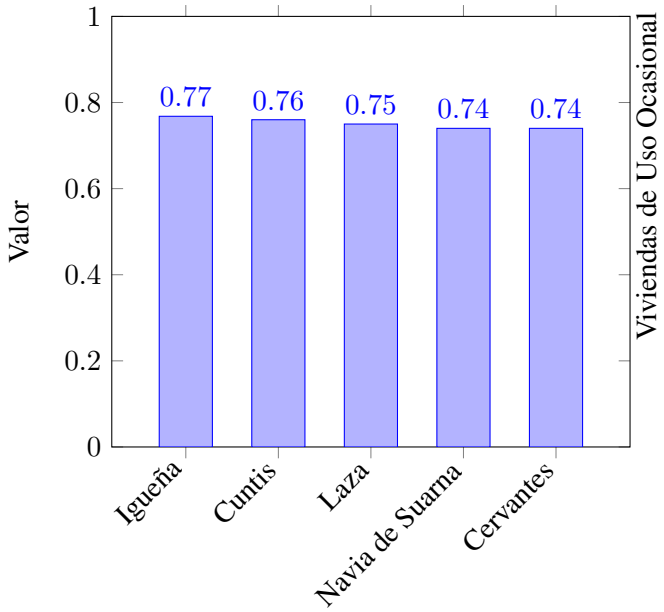
Donde ϕ_1 y ϕ_2 son la latitud en el primer y segundo punto respectivamente, λ_1 y λ_2 son la longitud en el primer y segundo punto y r es el radio de la Tierra.

Después de esto, el resultado se imprime por pantalla.

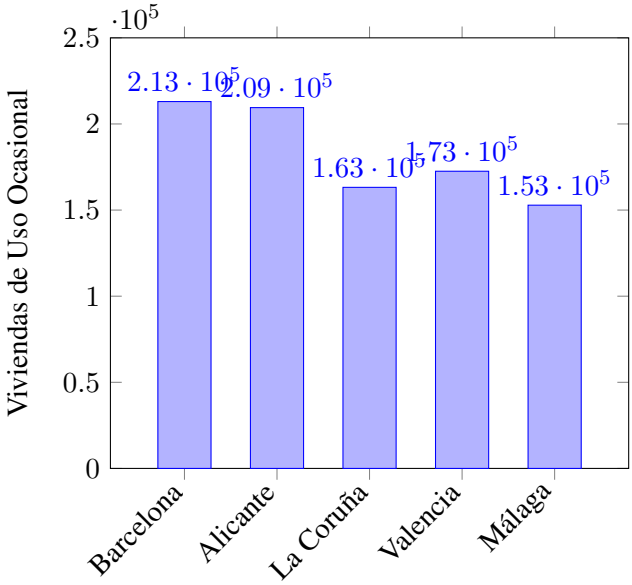
5 Resultados

5.1 MapReduce

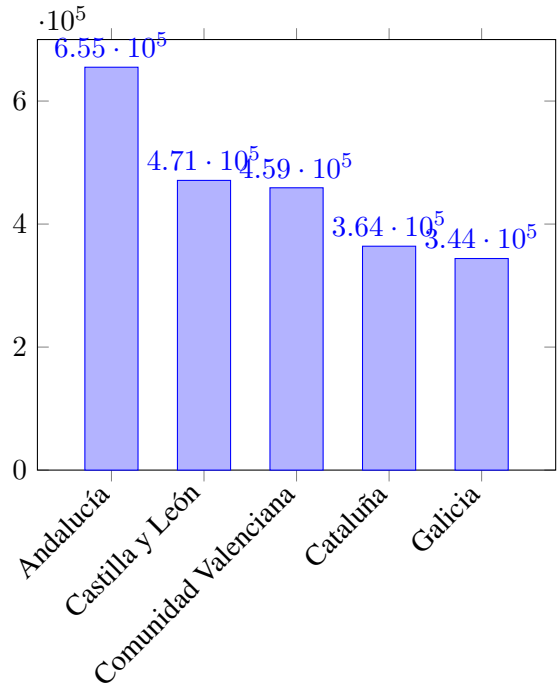
5.1.1 Municipios con Mayor Porcentaje de Viviendas Vacías



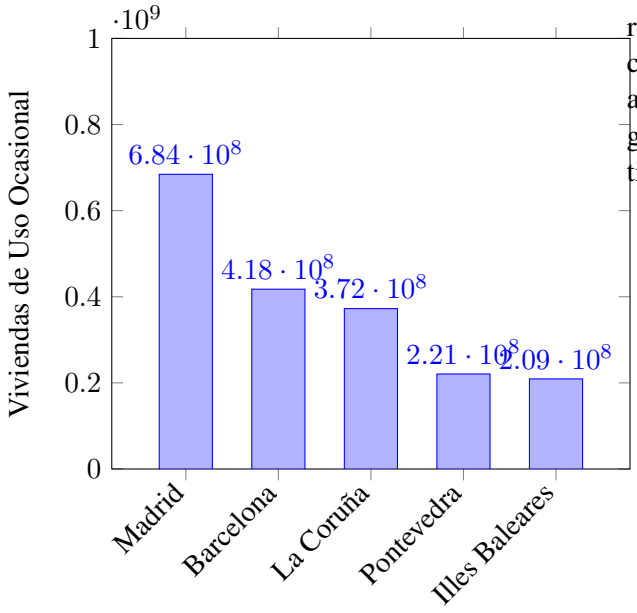
5.1.2 Municipios con Mayor Número de Viviendas Ocasionales



5.1.3 Número de Viviendas Vacías en Cada Provincia

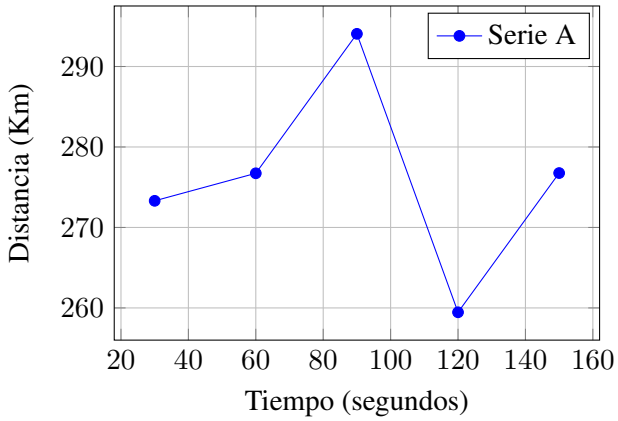


5.1.4 Estimación de la Energia Total Consumida or Provincia



Este gráfico muestra las cinco provincias que más energía consumen en $\frac{Kw}{h}$.

5.2 Cálculo de la Distancia Recorrida



6 Conclusiones

6.1 MapReduce

Tal y como se puede ver en los siguientes gráficos, la escasez de la vivienda depende en gran parte de la provincia. Mientras que en Madrid y Barcelona se encuentra en este gran problema con la adquisición de la vivienda. En cambio, en otros lugares como Andalucía o Castilla y León que son las comunidades con mayor número de viviendas vacías en toda España este suceso no ocurre. Por tanto, si se quiere solventar el problema que tiene España, se debe incentivar que la gente se mude a localidades que el porcentaje de viviendas vacías es muy alto.

6.2 Distancia Recorrida por la ISS

Ta y como se puede ver en el gráfico la distancia recorrida por la ISS es bastante estable. Además, cabe recalcar la gran velocidad en la que orbita alrededordde la Tierra. Esta velocidad mirando el gráfico es de $9.2\frac{km}{s}$ pudiendo hacer una vuelta a la tierra en tan solo 90 minutos.