MBIT School
Madrid Business Intelligence Technology

El índice inverso

Arquitecturas Big Data

Diego J. Bodas Sagi

Índice

- ¿Qué es un índice inverso?
- Caso práctico, análisis literario
- Breve explicación de los pasos a seguir
- Análisis de resultados
- Resumen

Índices inversos

MBIT School
Madrid Business Intelligence Technology

- Examinaremos el problema de los índices inversos muy frecuentes en MapReduce
- Esencialmente, un índice inverso es un ***glosario***
  - Es decir, tenemos una lista de palabras junto con las referencias a donde aparecen

Palabra	Documento
Sancho	El Quijote
Caballero	El Quijote, El Cid…
Análisis	Data Mining, Manual de…

  - También podría aparecer, la lista del byte offset (posición) donde podemos encontrar la palabra a lo largo del documento

Ejercicio

MBIT School
Madrid Business Intelligence Technology

- Nos descargaremos (projecto Gutenberg)
  - La Celestina (Fernando de Rojas)
  - Historia de la vida del Buscón… (Quevedo)
  - Vida de Lazarillo… (Anónimo)
- Para mejor identificación, renombraremos los ficheros descargados a “celestina.txt”, “buscon.txt” y “lazarillo.txt”
- Nos creamos un nuevo proyecto llamado api.exercises
- En la ruta asociada al proyecto, creamos el directorio inputFiles donde movemos los tres libros anteriores

Pasos

- El Mapper
  - ¿Qué recibimos y qué emitimos?
- Obtener el contexto y nombre del fichero
  - Como hemos visto, se crea una tarea mapper para cada división (“split”) de los datos de entrada
  - El contexto nos proporciona el método *getInputSplit()* con el objetivo de capturar esta división
  - Posteriormente, el nombre del fichero puede obtenerse a través de:
    - *String fileName = fileSplit.getPath().getName();*

Reducer

MBIT School
Madrid Business Intelligence Technology

- La cuestión está en que tenemos que devolver varios resultados para una misma clave
  - <caballero, (celestina.txt, buscon.txt, lazarillo.txt)>
- En realidad, lo que sucede es que estamos devolviendo varios String como salida
- La opción más sencilla de conseguir esto, es empleando un objeto StringBuilder de Java
- Añadiremos al StringBuilder las distintas localizaciones encontradas para cada palabra (clave) ***separadas por coma***
  - Hay que controlar que NO se introduzca una coma para la primera localización

El Driver

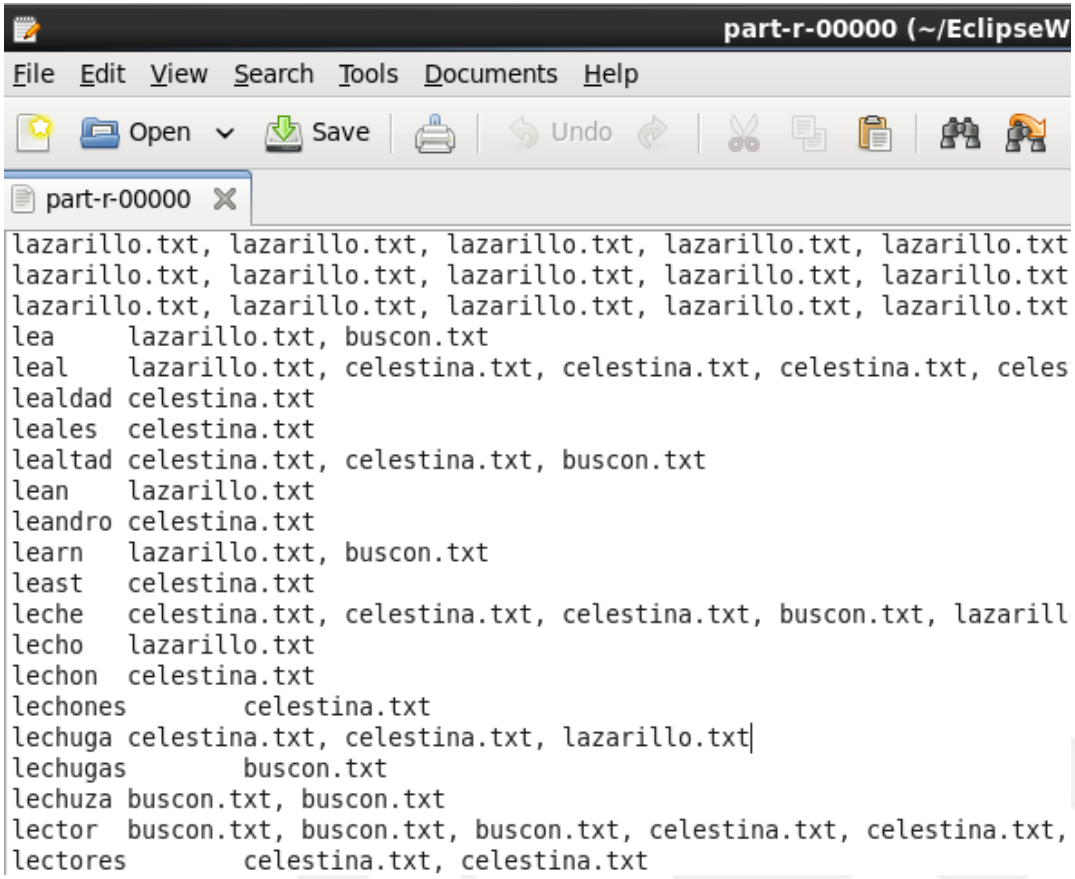
MBIT School
Madrid Business Intelligence Technology

- Exige leves cambios respecto al WordCount
- Repasar qué devuelve el Mapper
- Repasar qué devuelve el Reducer

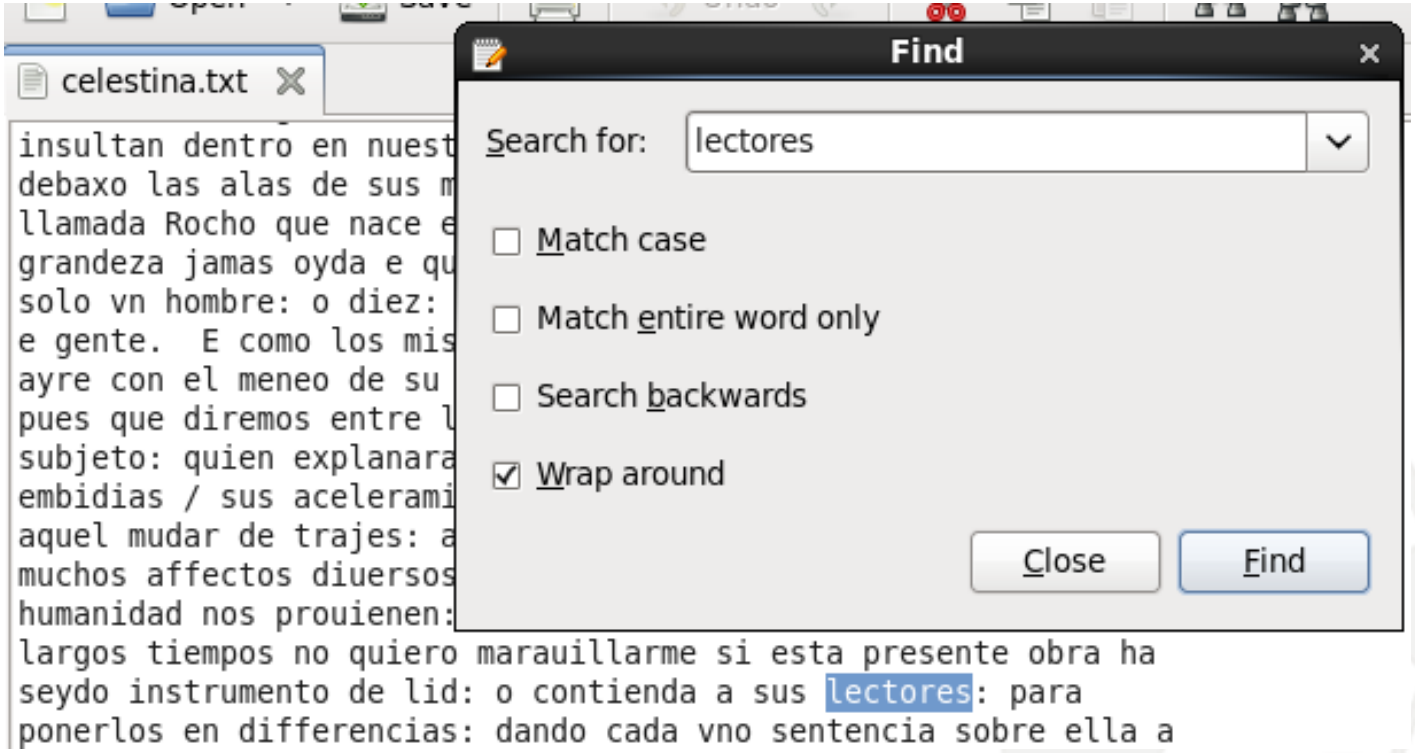
- Crear una nueva configuración para este programa
  - Recordar que ahora se deben leer todos los ficheros contenidos en el directorio
- Asegurarse de organizar las librerías importadas en todos los archivos

Resultado

- Si una palabra aparece varias veces, la localización también aparece varias veces
- ¡Podríamos hacer una análisis de estructura sintáctica basándonos en el resultado!
- Comprobemos que es correcto
  - ¿De verdad que “lectores” sólo aparece en La Celestina y dos veces?



Comprobando los resultados

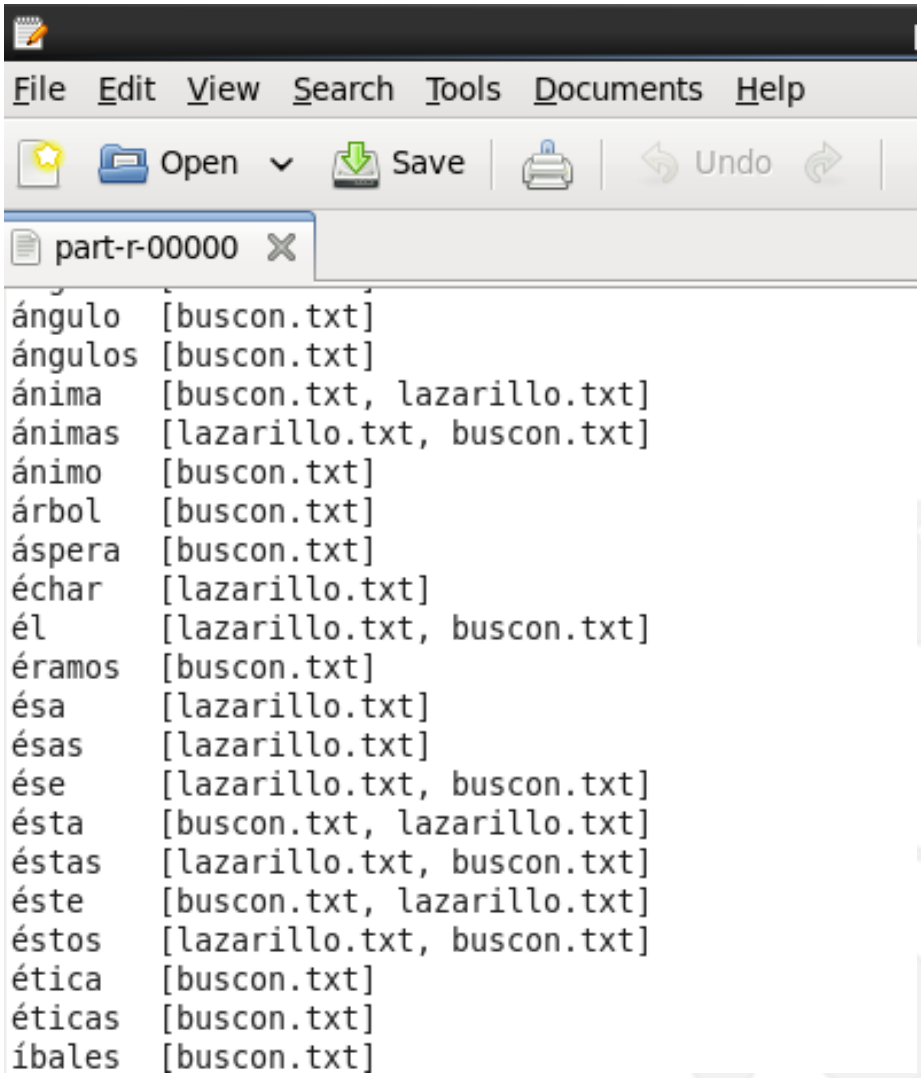


Mejoras

MBIT School
Madrid Business Intelligence Technology

- Debemos intentar mejorar la salida
- No tiene sentido que la localización salga repetida tantas veces como se repita la palabra en el texto
- Sugerencia:
  - Emplear en el Reducer la clase *LinkedHashSet<String>*
  - Al ser un “conjunto”, no permitirá elementos repetidos

Resultado mejorado




```
ángulo  [buscon.txt]
ángulos [buscon.txt]
ánima   [buscon.txt, lazarillo.txt]
ánimas  [lazarillo.txt, buscon.txt]
ánimo   [buscon.txt]
árbol   [buscon.txt]
áspera  [buscon.txt]
échar   [lazarillo.txt]
él      [lazarillo.txt, buscon.txt]
éramos  [buscon.txt]
ésa     [lazarillo.txt]
ésas    [lazarillo.txt]
ése     [lazarillo.txt, buscon.txt]
ésta    [buscon.txt, lazarillo.txt]
éstas   [lazarillo.txt, buscon.txt]
éste    [buscon.txt, lazarillo.txt]
éstos   [lazarillo.txt, buscon.txt]
ética   [buscon.txt]
éticas  [buscon.txt]
íbales  [buscon.txt]
```

archivos VS # mappers

MBIT School
Madrid Business Intelligence Technology

- Empaquetaremos la solución con Maven
- Llevaremos a HDFS el directorio junto con los tres libros
- Ejecutar la solución con “hadoop jar” especificando como parámetros los directorios de entrada y salida
- ¿Cuántas tareas mapper se crean?
  - 3: porque procesamos 3 ficheros de poco tamaño
- Reducer: 1 (opción por defecto)

Innovando

MBIT School
Madrid Business Intelligence Technology

- Hasta ahora, sabemos qué palabras aparecen en qué libros, pero
- ¿Sabemos cuántas veces aparecen?
- Si permitimos que imprima el título con cada aparición, sí, pero
  - Tenemos que contar el número de veces que aparece el título
    - Opción inviable
- Otras opciones:
  - Encadenar un segundo Job MapReduce después del primero
  - Diseñar un tipo personalizado que permita emitir pares <texto, entero> (todavía no lo hemos visto)

Probando otra opción

MBIT School
Madrid Business Intelligence Technology

- Podemos probar una opción cogiendo la idea de la ordenación secundaria (secundary sort)
- La ordenación secundaria consiste en aprovechar el trabajo de Hadoop a la hora de agrupar y ordenar por clave
- En este caso, se podría pensar en modificar el mapper, de forma que en vez de emitir <palabra, libro>, emita <palabra:libro, 1>

¿Por qué?

MBIT School
Madrid Business Intelligence Technology

- Porque el shuffle va a pasar los valores emitidos al Reducer ordenado en función de la clave
- Así, podremos tener lo siguiente:
  - palabra1:libro1, <lista de 1s>
  - palabra1:libro2, <lista de 1s>
  - palabra2:libro1, <lista de 1s>
- Con este formato, se puede comparar de forma sencilla en qué libros aparecen qué palabras y cuántas veces lo hace

Resumen

MBIT School
Madrid Business Intelligence Technology

- Los índices inversos son muy similares a un glosario
- Permite detectar en qué documentos o archivos están presentes las claves analizadas
- A pesar de ser una aplicación mapReduce sencilla, tiene muchas aplicaciones
- Para finalizar, hemos visto una pequeña variación que permite saber cuántas veces aparece cada palabra en cada libro