Contador de letras con MapReduce

Abertas: martes, 16 de abril de 2024, 00:00 AM
Pendente: xoves, 13 de xuño de 2024, 00:05 AM

Práctica Mappers y Reducers Python en local.

Objetivos

- Conocer y trabajar con mappers.
- Conocer y trabajar con reducers.

Fecha de entrega: viernes 12 de junio de 2024 hasta las 23:00 (hora del servidor)

Elabora y envía un documento PDF con tu nombre completo y DNI en la portada. Usa una nueva hoja por cada apartado, copia el enunciado y contesta.

En esta práctica realizaremos la implementación en Python de dos programas. El primero de ellos será un mapper que tiene como misión procesar parte de los datos en bruto. El segundo será un reducer que tiene como misión unir todos los datos previamente generados por los reducers.

Las primeras comunicaciones cifradas de las que se tiene constancia datan del siglo V a.c. En la época de los romanos, el cifrado "Julio Cesar" consistía en sustituir cada letra por la que tenía 3 lugares más adelante en el alfabeto. Así por ejemplo la letra "A" se sustituía por la "D". Hasta entrada la Edad Media variaciones de este sistema de cifrado se consideraban seguros pero en el siglo IX, el matemático árabe al-Kindi, considerado uno de los padres de la criptografía, desarrolló un sistema que usaba la frecuencia de aparición de las letras para determinar el idioma y la clave de descifrado.

En esta práctica vamos a determinar el número de apariciones de cada letra en un texto similar escrito en diferentes idiomas.

El objetivo de esta práctica es familiarizarse con la filosofía MapReduce que es la usada en el campo de procesamiento distribuido en BigData. La idea es hacer la práctica siguiendo el enunciado y después plantearse donde habría que tocar, en el mapper o en el reducer, para realizar alguna modificación.

Enunciado

1.- Crea un archivo mapper.py que recorra el "Whatsapp_español.txt" y para cada letra guarde en "salida_mapper.txt" una pareja de clave-valor donde la clave es la letra y el valor sea 1 indicando que es una aparición. Nuestro mapper no debería distinguir entre mayúsculas y minúsculas ni letras acentuadas. Por ejemplo, "U", "ü", "ú",... son la misma letra. Los números, espacios en blanco, signos de puntuación, comillas y similares no deben ser tenidos en cuenta, solamente queremos las letras del alfabeto.

Como respuesta a esta pregunta muestra tu código comentado.

2.- Aplica el archivo Python "ordenar.py" al archivo "salida_mapper.txt" para ordenar los resultados en un nuevo archivo "entrada_reducer.txt".

Muestra la salida del comando "head -n 20 entrada_reducer.txt" para ver las 20 primeras líneas del resultado.

3.- Crea un archivo llamado reducer.py que procese el "entrada_reducer.txt" y nos devuelva el número de apariciones de cada letra de nuestro texto en un archivo llamado "salida_reducer.txt".

Como respuesta muestra tu código comentado.

4.- Prueba tu mapper y reducer (aplicando ordenar.py como fase intermedia) con el resto de los archivos de prueba y elabora una tabla con las apariciones de cada letra en los distintos idiomas.

Devuelve el resultado de cada trabajo mapreduce donde se vea la cantidad de veces que aparece cada letra del alfabeto y una la tabla con los resultados finales de los 4 idiomas.

	ordenar.py	16 de abril de 2024, 11:07 AM
	Práctica mapreduce 01.pdf	16 de abril de 2024, 11:07 AM
	Whatsapp_español.txt	16 de abril de 2024, 11:07 AM
	Whatsapp_filipino.txt	16 de abril de 2024, 11:07 AM
	Whatsapp_inglés.txt	16 de abril de 2024, 11:07 AM

Whatsapp_portugués.txt 16 de abril de 2024, 11:07 AM

E ntrega

Estado da entrega	Aínda non se realizaron entregas
Estado das cualificacións	Sen cualificar
Tempo restante	A tarefa está atrasada por: 12 días 18 horas
Última modificación	-
Comentarios a entrega	▸ Comentarios (0)

◄ WordCount en Python con MRJob

Ir a...

Archivos de pruebas (actualización) ►