**การบ้าน MapReduce**

**ข้อ1. MapReduce ด้วย Hadoop**

ผู้จัดทำเขียน Blog อธิบายเพิ่มเติมไว้สามารถดูได้ที่ : [https://medium.com/equinox-blog/ทำ-mapreduce-word-count-บน-ubuntu-server-b2f149c43f77](https://medium.com/equinox-blog/%E0%B8%97%E0%B8%B3-mapreduce-word-count-%E0%B8%9A%E0%B8%99-ubuntu-server-b2f149c43f77)

การทำ MapReduce บน HDFS นั้นสามารถทำได้ตามขั้นตอนดังนี้

**1.Input Data**

ข้อมูล text file ที่จะนำเข้าใน MapReduce



**2.Imprement Mapper**

สร้าง file mapper2.py ไว้สำหรับ map ข้อมูลจาก data source ที่อ่านเข้ามา

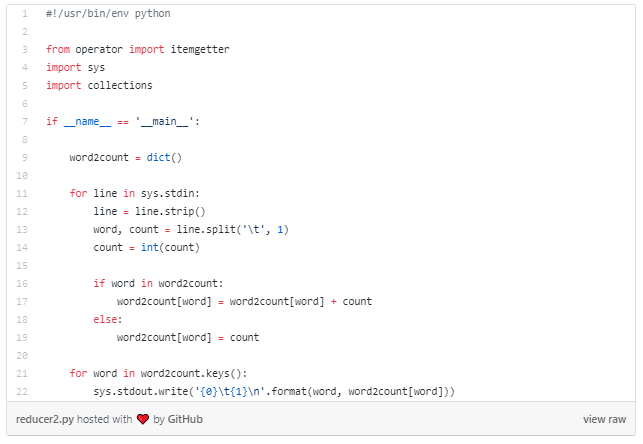


หรือใช้ #wget [https://raw.githubusercontent.com/Kzis/map\_reduce/master/mapper2.py](https://raw.githubusercontent.com/Kzis/map_reduce/master/mapper.py)

ได้เลยเนื่องจากผู้จัดทำได้นำไปวางไว้บน Host server เรียบร้อยแล้ว

**3.Imprement Reducer**

สร้าง file reducer2.py ไว้สำหรับ reduce data ที่ได้มาด้วย key หาก key ที่อ่านมามีค่าซ้ำก็ให้ทำการนับเพิ่ม จากนั้นจึง write ออกมา



หรือใช้ #wget [https://raw.githubusercontent.com/Kzis/map\_reduce/master/reducer2.py](https://raw.githubusercontent.com/Kzis/map_reduce/master/mapper.py)

ได้เลยเนื่องจากผู้จัดทำได้นำไปวางไว้บน Host server เรียบร้อยแล้ว

**4.Put file to Hadoop**

นำ file นี้ขึ้นไปวางไว้บน Hadoop ด้วยคำสั่ง #hadoop fs -put text.txt /user/cloudera/text.txt

จากนั้นดู file ที่ put ด้วยคำสั่ง #hadoop fs -ls /user/cloudera

**5.Run MapReduce**

ทำการ Run MapReduce Job ด้วยคำสั่งดังต่อไปนี้

# hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar

–input /user/cloudera/text2.txt –output /user/cloudera/wx2

–mapper mapper2.py –reducer reducer2.py

–file mapper2.py –file reducer2.py

โดยคำสั่งมี Option ดังนี้

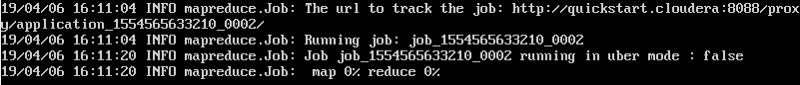
-input : text file ที่จำนำมาเป็น data source

-output : path ที่จะ write file output ลงไป

-mapper : file ที่จะทำหน้าที่เป็น mapper

-reducer : file ที่จะทำหน้าที่เป็น reducer

จากนั้นเมื่อทำการ Submit job ก็จะได้ดังภาพ



เมื่อ job success แล้วเราก็สามารถไปดูผลลัพธ์ได้ที่ path /user/cloudera/wx2

ด้วยคำสั่ง #hadoop fs -cat /user/cloudera/wx2/part-00000



**ข้อ2. MapReduce ด้วย Spark บน Google Colaboratory**

ผู้จัดทำเขียน Blog อธิบายเพิ่มเติมไว้สามารถดูได้ที่ : [https://medium.com/equinox-blog/](https://medium.com/equinox-blog/%E0%B8%97%E0%B8%B3-mapreduce-wordcount-%E0%B8%94%E0%B9%89%E0%B8%A7%E0%B8%A2-pyspark-b9bf2550529e)ทำ-mapreduce-wordcount-ด้วย-pyspark-b9bf2550529e

การทำ MapReduce ด้วย Spark บน Google Colaboratory นั้นสามารถทำได้ตามขั้นตอนดังนี้

**1.Install Java, Spark, and Findspark**

เนื่องจาก Google Colabortory เป็น online editor ที่เป็น ubuntu server จึงต้องทำการ install environment ที่ต้องการจะใช้งานด้วยคำสั่งดังต่อไปนี้

1.1 ทำการ Install Java8

!apt-get install openjdk-8-jdk-headless -qq > /dev/null

1.2 ทำการ Download Apache Spark 2.4.3

!wget -q http://www-eu.apache.org/dist/spark/spark-2.4.3/spark-2.4.3-bin-hadoop2.7.tgz

1.3 ทำการ Install Apache Spark

!tar xf spark-2.4.3-bin-hadoop2.7.tgz

1.4 ทำการ Install Findspark เพื่อเขียน Python สำหรับ Spark

!pip install -q findspark

**2.Set Environment Variables**

เมื่อทำการ Install ตามขั้นตอนที่ 1 แล้วให้ทำการ Set java home path สำหรับใช้งาน Apache Spark และ Set spark home path สำหรับใช้งาน PySpark ด้วยคำสั่งดังต่อไปนี้

import os

os.environ[“JAVA\_HOME”] = “/usr/lib/jvm/java-8-openjdk-amd64”

os.environ[“SPARK\_HOME”] = “/content/spark-2.4.3-bin-hadoop2.7”

**3.Start a SparkSession**

**เ**มื่อสร้าง Environments สำหรับ PySpark เรียบร้อยแล้วให้ทำการสร้าง SparkSession กำหนด Cluster Manager เป็น Local และ สร้าง SparkContext สำหรับ Call API ของ Spark ด้วยคำสั่งดังต่อไปนี้

import findspark

findspark.init('/content/spark-2.4.3-bin-hadoop2.7')

from pyspark.sql import SparkSession

from pyspark import SparkContext

spark = SparkSession.builder.master("local").getOrCreate()

sc = SparkContext.getOrCreate()

**4.Create function read data source**

ทำการสร้าง file test.csv โดยภายในมีข้อมูลดังนี้

****

จากนั้นทำการสร้าง function สำหรับการอ่านข้อมูล text file ที่เป็น data source ดังนี้

****

**5.Create function convert data to 2gram list**

ทำการสร้าง function สำหรับ convert list data source ที่ทำการอ่านขึ้นมาเป็น list 2 gram word ดังนี้



จากนั้นทำการนำ convert list data source เป็น list 2 gram ด้วยคำสั่ง ดังต่อไปนี้

map\_list = mapper2gram(line)  
 extract\_list = extract2gram(map\_list)

จะได้ว่า extrac\_list เก็บข้อมูลของ data source แบบ 2gram

**6.Create RDDs from list**

ทำการ convert list 2 gram เป็น RDDs โดยการ parallelize ผ่าน Spark context ด้วยคำสั่ง ดังต่อไปนี้

rdd = sc.parallelize(extract\_list)

**7.Map key**

ทำการ Tranformation RDDs ตัวใหม่ด้วยคำสั่ง map โดยมี function การ map คือ ค่าที่ส่งเข้าจะถูกแปลงเป็น tuple ของค่าที่ส่งเข้าไปเป็นคีย์ และมีค่าเป็น1 ( คู่อันดับ <K,V> ) ด้วยคำสั่ง ดังต่อไปนี้

words = rdd.map(lambda word: ((word, 1)))

**8.Reduce by key**

ทำการ Actions RDDs ด้วยคำสั่ง reduceByKey โดยมี function reduce คือ การส่ง parameter เข้าไปทีละ 2 ตัว และ ถ้า key เหมือนกันจะนำ value มารวมกัน

counts = words.reduceByKey(lambda a, b: a + b)

print(counts.collect())

ก็จะได้ผลลัพธ์ออกมาดังนี้

[('1 1', 2), ('1 2', 2), ('2 2', 2), ('2 3', 1), ('3 3', 2), ('3 4', 1), ('4 5', 1), ('5 6', 2), ('a b', 1), ('b c', 2), ('c 1', 1), ('2 7', 1), ('7 8', 1), ('8 9', 1), ('9 5', 1), ('6 b', 1)]

**Download Source Code ทั้งหมด ได้ที่ :** <https://github.com/Kzis/map_reduce>