THE UNIVERSITY OF SYDNEY

School of Information Technologies
Faculty of Engineering & IT

ASSIGNMENT/PROJECT COVERSHEET - INDIVIDUAL ASSESSMENT

Unit of Study: COMP5349

Assignment name: Assignment 1

Tutorial time: Thursday 4pm – 6pm

Tutor name: Songfen Lu

DECLARATION

I declare that I have read and understood the *University of Sydney Academic Dishonesty and Plagiarism in Coursework Policy*, and except where specifically acknowledged, the work contained in this assignment/project is my own work, and has not been copied from other sources or been previously submitted for award or assessment.

I understand that failure to comply with the the *Academic Dishonesty and Plagiarism in Coursework Policy*, can lead to severe penalties as outlined under Chapter 8 of the *University of Sydney By-Law 1999* (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgement from other sources, including published works, the internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

I realise that I may be asked to identify those portions of the work contributed by me and required to demonstrate my knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

Student ID: 470360839

Student name: Nagib Shah

Signed Nagib **Date** 06/04/2019

SIT Building, J12
The University of Sydney
NSW 2006 Australia

T +61 2 9351 3423
F +61 2 9351 3838
E sit.info@sydney.edu.au
sydney.edu.au/it

ABN 15 211 513 464
CRICOS 00026A

Workload	Implementation	Programming Language
Category & Trending Correlation	MapReduce	Python
Top 10 Controversial Videos	Spark	PySpark/Python

Workload 1 – Category & Trending Correlation

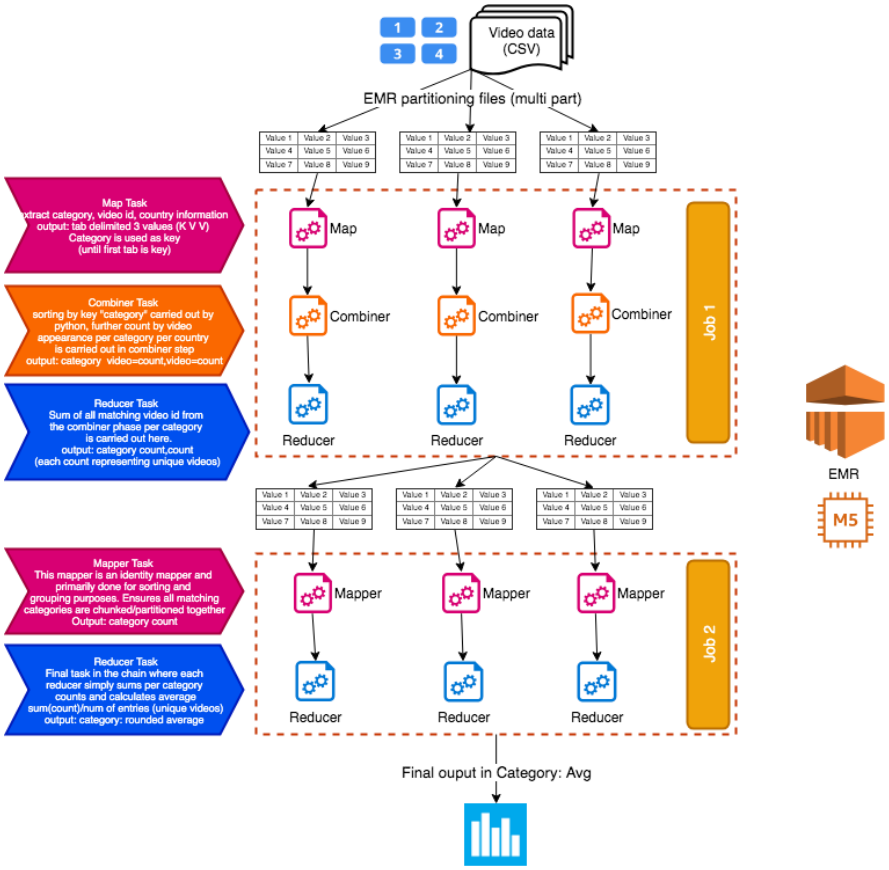

Figure 1: MapReduce Computation Graph for Calculating Average trend per category

Description:

Job1 firstly, takes individual parts/partitions from the MapReduce framework, for illustrative purposes only 3 Map Tasks are shown in in figure 1 where each map task extracts the category, video id, country from each line of the input partition and writes to the stdout in category \t video_id \t country format where the default MapReduce configurations are utilised to use Category as the Key in the (K \t V \t V) entry. Next a custom combiner step/task is executed to reduce the amount of data transfer between nodes (within a cluster) by carrying out aggregation on the Map partitions by default sorting by the Category as key and counting the video appearances per video in each category per country. In other words, the output of the Combiner task is a stdout entry in (key \t value=count, value=count) format (category \t videoid=count,videoid=count). The sorted output partitions are then fed through to a reducer task (final task in job 1) where the matching video id counts per category is simply summed up with output in (key value, value) format (category<tab>sum1,sum2) where each value is a sum of video appearances per country. The output of the Job 1 reduce task is still a multi part file where the same category appearing across multiple partitions (due to the initial file partitioning carried out before the first map task).

A chained MapReduce job 2 is then created which utilises the part outputs from job 1 as input to carry out further aggregation. The map task in Job 2 is a simple identity mapper where the output is simply (K V) format (category count) format. The primary role of this identity mapper is to enforce a sorting to be carried out automatically to ensure all matching category entries appear in the same partition (no overlap entries across parts). Lastly, a final reduce task is carried out to calculate the average correlation value (sum of all counts / num of entries). The output of this reduce step is a single entry per category and its corresponding correlation value rounded to 2 decimal points.

Workload 2: Top 10 Controversial Videos

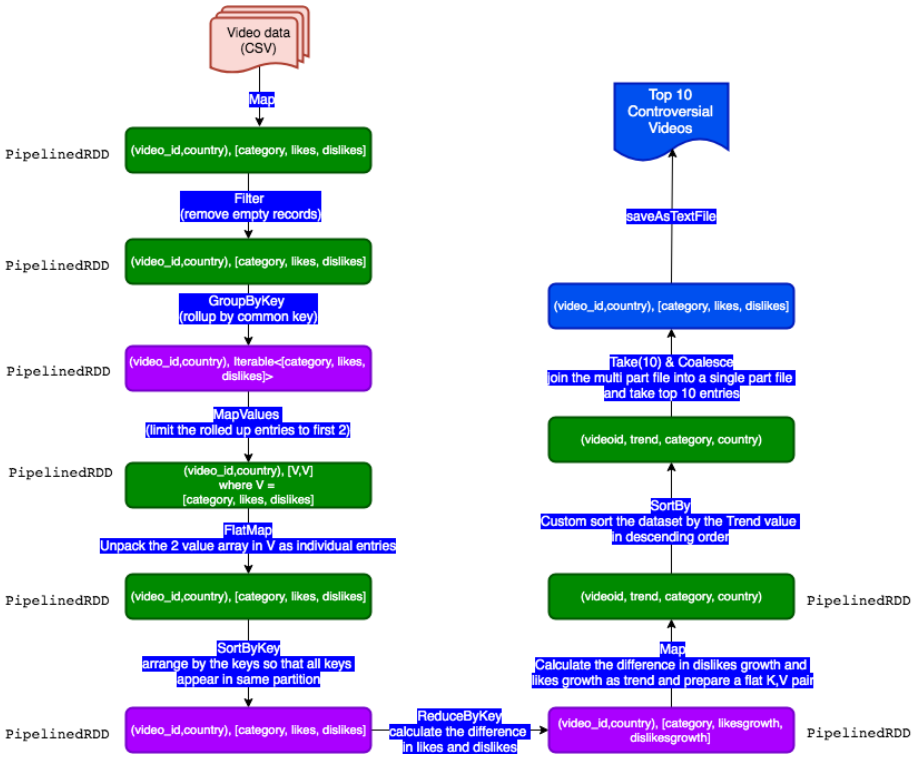

Figure 2: Spark Computation Graph for generating top 10 controversial videos (dislikes increase over likes increase)

Description:

The sequence of transformation and actions required on the initial RDD is illustrated in figure 1 to get the desired list of top 10 most controversial videos. The objective of the series of transformations are to identity the first 2 occurrences of trending videos (per video id) and subsequently calculating the difference in dislikes and likes, before calculating the controversial trend value (difference between growth of dislikes and growth of likes). Firstly, the CSV file is read into Spark as a RDD containing video id, and country as a composite key and the value pair containing an array of category, likes, and dislikes. A UDF (user defined function) is utilised to map the file in the K, V pairs as explained. A filter is applied firstly, on the pairRDD to ignore and filter out rows with valid values only (ignoring any null or empty entries). A groupByKey transformation function is then applied on the RDD to roll-up all the values belonging to each video id, country composite key resulting in an (K, Iterable<V>) RDD where each K is an id, country composite and each V is an array containing category, likes and dislikes loaded as integers. Further to the grouping a mapValues transformation is carried out on each value V (without changing the key). A UDF is utilised for the mapValues to only keep the first 2 occurrences within each iterable<V> resulting in a reduction of the RDD value (2 entries per video max) where the value contains arrays of max size 2. A flatMap transformation is then carried out to unpack the array to individual entries of K, V pairs where K is still the composite key and V contains a single array of category, likes, and dislikes.

With the reduced RDD containing a limited set of data from the original RDD pair, a sortByKey operation is carried out to ensure all entries are sorted together by their keys. This ensures all matching entries appear together in the same partition. With the base RDD now ready for some processing, a reduceByKey transformation is carried out on the (K, V) pair where an UDF is utilised again to calculate the difference of likes and dislikes by subtracting the values from their respective first entries. The output of the reduceByKey transformation is a new pairRDD where the K still remains the composite key but the value is replaced with a single entry per key containing category, and calculated dislikes and likes growth (dislike2 – dislike 2 & likes2 – likes1). For the final major transformation, a map transformation is carried out where the UDF carries out the required calculation by subtracting dislikes growth from the likes growth resulting in a new flat RDD containing id, category, trend, and country per record. In order to identify the top 10 most trending videos a custom sortBy method is utilised to sort by the trend value in descending order. Lastly, a simple take(10) and coalesce action is carried out on the reduced RDD to generate a list of 10 entries before writing the final output to file.