University of Lleida

Master's Degree in Informatics Engineering

Massive Data Processing

Hadoop Activity

Francesc Contreras

Table of contents

1 Implementation

First of all, the main implementation uses the JSON files, because it is structured data. Also, the activity has the optional parts of sentiment, compressed sequence and custom writable.

1.1 Trending Topic

This part of the code has the aim of detecting all the hashtags (#word) and count the occurrences of it.

- **TrendingTopicMapper:** Extracts the hashtags using a regex from the text files and write them to the context. It uses the files that have the extension ".txt".
- **TrendingTopicMapperJSON:** Extracts the hashtags from the JSON files and writes them to the context.
- **TrendingTopicReducer:** Gets the data from the mapper *TrendingTopicMapper* or *TrendingTopicMapperJSON* and counts how many times each hashtags appears.

```
yarn jar ./HadoopActivity.jar trendingtopic/TrendingTopic /user/fcp5/out/cleanup
/user/fcp5/out/trendingtopic
```

1.2 Clean up

This part of the code has the aim of clean data and it have 4 Mappers (filters).

- **CorrectFieldsMapper:** Remove the tweets without text or hashtags.
- **LanguageFilterMapper:** Remove the tweets that are not in Spanish.
- **CustomFieldSelectorMapper:** Returns the JSON with the fields *text, hashtag, lang.*
- **LowerCaseMapper:** Convert all the tweets to lower case.

```
yarn jar ./HadoopActivity.jar cleanup/Cleanup /user/fcp5/tweets/*.json
/user/fcp5/out/cleanup
```

1.3 Top N

From the output generated by the clean up task, the Top N task will get the N hashtags with most occurrences.

- **TopNMapper:** Calculate the local top 2N and return all the records processed.
- **TopNReducer:** Obtain the records processed and calculates the global top N.

```
yarn jar ./HadoopActivity.jar topn/TopN 10 /user/fcp5/out/trendingtopic
/user/fcp5/out/topn
```

1.4 All Jobs

This part contains all the tasks described before. To link all the task I use JobControl to assign their dependencies (Cleanup,Trending Topic and Top N).

```
yarn jar ./HadoopActivity.jar alljobs/AllJobs /user/fcp5/tweets/*.json
/user/fcp5/out 10
```

1.5 Sentiments

This task is optional, it is related to detect the sentiments of a tweet. I use distributed cache for the positive and negative files and custom Writable for the tweet record.

- **SentimentMapper:** Uses the setup method to read files and checks the overall feelings of the tweet.
- **SentimentReducer:** Normalises the feeling of each hashtag to the length of the tweet and emits the feelings of each hashtag.

```
yarn jar ./HadoopActivity.jar sentiment/Sentiment /user/fcp5/out/cleanup
/user/fcp5/out/sentiment /user/fcp5/data/positive_words_es.txt
/user/fcp5/data/negative_words_es.txt
```

2 Performance

Once the implementation has been finished, the performance has been checked. Since some versions ago, mappers are already defined by default and the user cannot change them. However, we can change the file size and amount of files passed as input.

File	**Size**	**Time**
tweets.json	904MB	102122ms
tweets2.json	4.3MB	87225ms
tweets_es.json	86.2MB	87266ms
tweets2_es.json	364.5kB	87196ms
tweets/*.json	-	102105ms

As we can see, the system scales properly since rising the file size does not increase the time by the same proportion. That's due the high amount of I/O and that the files are not big enough to make the difference noticeable.