DATA8005 – Assignment 2 Part 2

Due: 16th December @ 23:59 (Note: the module descriptor says Week 12, but I will allow 1 extra week for part 2 considering there is no final exam for the module and there are 13 weeks for a module)

Worth: 30%

Overview

Wikimedia is a global movement whose mission is to bring free educational content to the world. It consists of the following projects:

- Wikipedia
- Wiktionary
- Wikiquote
- Wikibooks
- Wikisource
- Wikinews
- Wikiversity
- Wikispecies
- MediaWiki
- Wikidata
- Wikimedia Commons
- Wikivoyage
- Wikimedia (meta-wiki)
- Wikimedia incubator
- Wikimedia labs
- Wikimedia foundation

For more information on any of these projects please visit: https://www.wikimedia.org/

Wikimedia provides dumps with page view statistics. These dumps are accumulated daily, on an hourly basis, and contain about 100MB of compressed entries. The following presents three entries from the dump accumulated 17th of March 2016 at 10am:

```
. . .
en Main_Page 242332 4737756101
en.d Index:German 1 6316
en.voy South_Korea 6 744464
. . .
```

- The first column of each entry represents the country and the project name. Whereas en stands for English speaking version of the page, a lack of extension stands for Wikipedia (other projects have an associated extension: { (.d, Wiktionary), (.q, Wikiquote), (.b, Wikibooks), (.s, Wikisource), (.n, Wikinews), (.v, Wikiversity), (.voy, Wikivoyage), … }
- The second column of each entry represents the title of the page retrieved.
- The third column of each entry represents the total number of requests during the hour the dump accumulates statistics for.

- The fourth column of each entry represents the total size of the content returned.

You have been provided with the complete dump for 17th of March 2016 at 10am. The dump has been previously *cleaned* (by filtering for only English speaking entries) and distributed among the following 25 text files:

- new_pagecounts-20160317-100000_0.txt
- new_pagecounts-20160317-100000_1.txt
  …
- new_pagecounts-20160317-100000_24.txt

Each file has been sorted by project name (i.e. each file first contains all the entries for Wikipedia, then all entries for Wikibooks, and so on).

Your task

Following the MapReduce development environment presented in the lectures, you are provided with the following 4 Python files:

- my_mapper.py
- my_mapper_simulation.py
- my_sort_simulation.py
- my_reducer.py

The goal of the assignment is to complete the files my_mapper.py and my_reducer.py so as to develop a Hadoop MapReduce job finding the 5 most popular queries (the ones with most requests) per Wikimedia project.

Python Files Description

1. **my_mapper.py**
   This program executes a single mapper. It contains the following functions:
   - **i)** **process_line(line)**
     Given a line of our text file, this function returns a list with all its words.
   - **ii)** **create_initial_list(length)**
     Given a target length, this function creates a new list of length 'length', where each element of the list is set to the tuple (0, 0, "").
   - **iii)** **check_if_popular(current_list, new_request, new_transferred, new_page)**
     Given the current list of popular entries of the project, and the new entry of the project just read, this function determines whether the new entry is one of the 5 most popular ones. If this is the case, it replaces the least popular query processed so far by the new entry just read.
   - **iv)** **print_key_value(project, current_list, output_stream)**
     Given the current project we have processed all entries for, the list with the 5 most popular entries and the output stream, this function prints the (key, value) pairs of the popular entries. A possible (key, value) pair could be:
     project (request, transferred, page_title)
   - **v)** **my_map(input_stream, output_stream)**

Given the input and output stream channels, performs the map process of the file being passed by input stream, outputting the generated (key, value) pairs by the output stream.

vi) my_main( )

This function provides two operating modes:
Mode 1 → Testing: For testing the mapper on a single file of the dataset.
Mode 2 → Actual MapReduce: For operating on the Cloudera Environment. This mode should be used once you have ensured that the mapper works properly (by testing it over the entire dataset with successful results).
The function is provided completed, so you do not have to edit it. Just select the mode you want to work with (mode 1 or 2). In both modes, the function just triggers my_map using the input and output stream channels being selected.

Finally, the main entry point of the Python program triggers the execution of the function my_main().

2. my_mapper_simulation.py

This program executes the simulation of the mappers over the entire dataset. It is provided completed, so you do not have to edit it.

3. my_sort_simulation.py

This program executes the simulation of the sort phase over the results provided by the mappers. It is provided completed, so you do not have to edit it.

4. my_reducer.py

This program executes a single reducer. It contains the following functions:

i) get_key_value(line)

This function is very similar to the function process_line of my_mapper.py. The only difference is that, now, the text line being read has the format (key, value) generated as an output by the mapper (instead of the original Wikimedia dump entry read in process_line of my_mapper.py).

ii) create_initial_list(length)

This function is the same as the one in my_mapper.py, so just reuse it.

iii) check_if_popular(current_list, new_requests, new_transferred, new_page)

This function is the same as the one of my_mapper.py, so just reuse it.

iv) print_key_value(project, current_list, output_stream)

This function is the same as the one of my_mapper.py, so just reuse it.

v) my_reduce(input_stream, output_stream)

This function is very similar to the function my_map of my_mapper.py. The only difference is that the text line being read now has to be processed with the function get_key_value (instead of with the function process_line).

vii) my_main( )

This function provides two operating modes:
Mode 1 → Testing: For testing the reducer on the file generated by sort_simulation.py.
Mode 2 → Actual MapReduce: For operating on the Cloudera CDH

Environment. This mode should be used once you have ensured that the reducer
works properly (by testing it over the file sort_simulation.txt with successful results).
The function is provided completed, so you do not have to edit it. Just select the
mode you want to work with (mode 1 or 2). In both modes, the function just triggers my_reduce using the input and output stream channels being selected.

Finally, the main entry point of the Python program triggers the execution of the function my_main().

Marking Scheme

- Total marks: 30 marks, distributed as follows:
  - Function process_line: 4 marks.
  - Function create_initial_list: 3 marks.
  - Function check_if_popular: 4 marks.
  - Function print_key_value: 3 marks.
  - Function my_map: 11 marks.
  - Function get_key_value: 4 marks.
  - Function my_reduce: 1 marks (pretty much the same as my_map).

Submission

You should submit everything except the contents of the my_dataset folder – include all folders and scripts, just not the .txt files.

Upload as a zip file to the e-assignment in Blackboard.