**Database Project (Fall 2023)**

**Homework #2 (Due date: Oct 20)**

**Student ID**: 2020315798

**Name**: Choi Jin Woo

**Compress 1) your codes and 2) the document as follows:**

* ‘DBP\_HW2\_STUDENTID.zip’
  + Code
    - brute\_force.py
    - mapper.py, reducer.py, combiner.py
  + Document: DBP\_HW2\_STUDENTID.pdf

**NOTE:** You need to install matplotlib library.

1. **[60pts]** Consider you’re searching for restaurants to book in Suwon. You must choose optimal restaurants in **‘Suwon’** while considering conflicting features. Use pareto-optimal set (skyline query) to filter the results from the database to keep only those objects that are not worse than others.
   1. **[30pts]** Write the function named ‘pareto\_optimal’ in the code file ‘brute\_force.py’ to obtain the set of Pareto optimal restaurants in **‘Suwon’** from the ‘restaurant.txt’ dataset and plot the result.

**Instructions:**

* Refer to the definitions below and the baseline code which uses a brute-force algorithm to find the Pareto-optimal set. When plotting, please use the provided code. All data features should be considered. In general, higher quality and service are preferred, and lower prices are preferred.
* You can check the implementation of pareto\_optimal function with the compare\_result function in the main.py file.

**[Definition 1: Dominant relationship]**

For tuple , (Dominant relationship) satisfies when two conditions are true.

For each , is less than or equal to .

There is at least one that satisfies .

**[Definition 2: Dominant tuple]**

Given a set of tuples , a set of dominant tuples is defined as:

a set of tuples except for has no element that satisfies }

Answer: Enter your code and result here.

|  |
| --- |
| **pareto\_optimal function code** |
| **Plot** |

* 1. **[30pts]** Write the **‘mapper.py’** and **‘reducer.py’** code to produce the same results using MapReduce as the pareto\_optimal code you wrote in Problem 1 – (1).

**Instructions:**

* You can check the result output with the compare result function in the main.py file.

|  |
| --- |
| **mapper.py** |
| **reducer.py** |

1. **[40pts]** Write additionalcode **‘combiner.py’**. Compare the time difference when using the combiner versus not using it (from 1-(2)) and provide an explanation for the observed variance.

**Instruction:**

* Run the command *$ bash base.sh* and *$ bash base\_combiner.sh* to run MapReduce and compare the time consumed. **Capture and report the images from each case, and briefly explain the result.**
* **If you are using M1 mac and wxw-matt:docker-hadoop repository, you must modify the version of Hadoop streaming in shell files.**
* To ensure correct functionality of the shell files, the output file in HDFS must be in the **/hw/output** path. If you are going to use your own path, you must modify the shell files.

|  |
| --- |
| **combiner.py** |
| **Report**  [Without using combiner.py]    [With using combiner.py]    Using combiner.py reduced almost 8 seconds compare to one that is not using it. The purpose of the combiner is to perform local aggregation, and in this scenario, the logic for the local aggregation is the same as the global aggregation logic in the reducer. As a result, employing the combiner can help in minimizing the data transfer and improving the overall performance. |