Lab 02: MapReduce Programming

Instructor: Doan Dinh Toan
Department of Computer Science
Faculty of Information Technology
University of Science - VNU-HCM
toandd.i81@gmail.com

Abstract

This lab assignment introduces students to the MapReduce programming model and provides practical experience with solving ten different problems using MapReduce. This assignment aims to help you develop your problem-solving skills and gain practical experience with the MapReduce programming model.

0 Preliminary

0.1 Reminder

The main objective of this course is to truly learn. You can discuss this with your classmate, but you need to take responsibility for your submission, which actually depends on your understanding.

For any kind of cheating and plagiarism, students will be graded 0 marks for the course.

0.2 Submission guideline

Each team submits its result to a folder named `teamABC`, with `ABC` being the team's name. The folder structure is as follows:

```
.
└── teamABC/
    ├── src/
    │   └── problem01
    │   └── problem02
    ├── docs/
    │   ├── report.pdf
    │   └── report.[md|tex|typ]
    │   └── images
```

- `src` is the folder for your source code. If the lab assignment is split into multiple sections, you have to save your script in a separate folder, corresponding to the given lab assignment.
- `docs` is the folder for your documents, including the work report and images associated with your report. If the lab assignment requires screenshots as proof, the images need to be stored in this folder if you inserted them in the report.
  - `report.[md|tex|typ]` is your raw report file. The report must be written in English. The report must include the following items:
    - Information about the course, the assignment, and notes to the instructors (if any).
    - Information about your team (Student ID, full name of each member).
    - Your team's result (How much work, in percent (%), have you finished in each section?)

    - The answer to each section's tasks.
    - Reflection of your team. (Does your journey to the deadline have any bugs? How have you overcome it? What have you learned after this process? If you cannot overcome the bugs, describe where the bottlenecks are in your work.)
    - References to your work.
  - `report.pdf` is the PDF file of your report. You need to check this file carefully before submit[1].

0.3 Rubrics

Students will get 1 point for each problem.

0.4 Instructions

Follow these instructions for each problem:

- Read the requirements of the problem and think about a solution.
- Try to solve the problem by yourself first.
- If you can solve the problem, write a MapReduce program to implement your solution. Include comments in your code to explain your thought process and any assumptions you made.
- Run your MapReduce program and capture the output. Include the output as part of your submission.
- If you are stuck or need help, reference external sources to get inspiration and guidance. However, you should always correctly cite your sources and not copy code or solutions without permission.
- Submit your solution code, instructions on how to run it, and the output.

Copying solutions without proper attribution is considered plagiarism and can result in serious consequences. Any copied solutions will receive a grade of 0 if there are no suitable references.

Some dataset in exercises could not be exist, so you can use other dataset to run your program. Some resources you can download dataset such as:

- `PUMA Benchmarks`
- `EHPD`
- `Alteryx Designer Knowledge Base`

You could search for other alternative datasets on the internet or generate a dataset to evaluate (please describe it in the report). Efficient algorithm design is crucial for processing large-scale graphs in the context of big data, you might found some interesting ideas in this work[1].

[1]The report must be written in one of the following formats: Markdown (.md), LaTeX (.tex), or Typst style (.typ). The report is embedded with a specific template in Typst, therefore, regardless of the tool used, it is **essential to ensure that the formatting and design layout remain consistent with the provided template**.

1 Problems

To address problems 1 to 9, please refer to the detailed requirements specified in the document [2]. You can find this document uploaded in the lab's drive folder.

For each problem, you are required to write your solution in a single file, using the appropriate name provided in the Table 1. If your solution requires multiple files, please ensure that the entry points of your program are included as specified.

No.	Problem	Solution filename
1	WordCount	`WordCount.java`
2	WordSizeWordCount	`WordSizeWordCount.java`
3	WeatherData	`WeatherData.java`
4	Patent	`Patent.java`
5	MaxTemp	`MaxTemp.java`
6	AverageSalary	`AverageSalary.java`
7	DeIdentifyData	`DeIdentifyData.java`
8	MusicTrack	`MusicTrack.java`
9	CallDataRecord	`CallDataRecord.java`
10	CountConnectedComponentProgram	`CountConnectedComponentProgram.java`

Table 1: Problem's name and their related filename

1.1 Wordcount Program

1.2 WordSizeWordCount Program

1.3 WeatherData Program

1.4 Patent Program

1.5 MaxTemp Program

1.6 AverageSalary Program

1.7 De Identify HealthCare Program

1.8 Music Track Program

1.9 Telecom Call Data Record Program

1.10 Count Connected Component Program

1.10.1 Problem statement

You are given an adjacency list representation of an undirected graph. Your task is to write a MapReduce program that counts the number of connected components in the graph.

1.10.2 Input

The input consists of a list of vertices and their adjacent vertices. Each vertex is represented by a unique number listed on the leftmost side of the input. The adjacent vertices of each vertex are listed after it, separated by a space.

1.10.3 Output

Number of conntected component in graph (example in Table 2) .

Input	Output
0 9 1 4 9 2 7 3 5 8 4 1 5 3 6 7 2 8 3 9 0 1	4

Table 2: Sample of input and output.

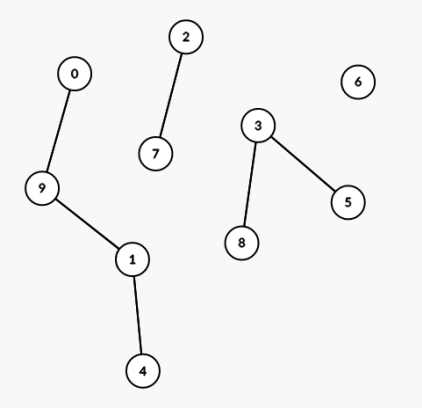

Figure 1: Visualization.

Bibliography

[1] J. Lin, and M. Schatz, "Design patterns for efficient graph algorithms in MapReduce," in *Proc. Eighth Workshop Mining Learn. Graphs*, Washington, D.C., Jul. 2010, pp. 78–85, doi: 10.1145/1830252.1830263. Accessed: Mar. 13, 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/1830252.1830263

[2] S. Balasubramanian, *Hadoop-Mapreduce Lab*, University of California, Berkeley, 2016.