BEY Benjamin

TAMINE Yassine

YAZID Ayman

C-Wire Project Report

Teacher: GRIGNON Romuald

Introduction :

The C-Wire project consists of developing a program that analyzes the electricity distribution in a complex network. This network, organized like a rooted tree, starts from power plants and delivers energy through several levels of intermediate stations to the final consumers.

The goal is to create two complementary tools: a Shell script to filter and prepare the data, and a C program that performs calculations on capacity and energy consumption using optimized data structures like AVL trees. These tools must efficiently handle a very large CSV file containing millions of lines.

In this report, we will first explain how we organized ourselves, then describe our development process over time, and finally discuss the problems we encountered and how we solved them.

Our giltlab:

We have a Code_C directory containing the C program and all associated files, such as the Makefile and the executable. An input directory holds all the source files. The tests directory contains all possible commands, without optional options, sorted by increasing capacity, with the source file c-wire_v00.dat. At the same time, the tmp directory is used to store the intermediate files that helped produce the final results. Finally, also contains the README and this report.

Organisation :

In Initially, we planned to use the GitHub platform to upload the different progress updates of the code. However, we faced difficulties when trying to perform a 'git clone', even using a token. As a result, with the agreement of our teaching assistant, Mr. Grignon, we decided to switch to GitLab.

For communication, the group chose to use the social network Discord for audio calls, screen sharing, and Snapchat for sending photos and videos via phone.

Regarding commit management, we decided to assign this task to Benjamin. After facing merge issues in sub-branches in previous years, we opted to centralize the code versions on Benjamin's PC, so he handled the commits in the main branch on GitLab.

We then decided to assign tasks based on each person's skills and availability according to the days. Ayman and Yassine were more comfortable with the C language than with Shell, so we gave priority to Benjamin for the Shell part, while Ayman and Yassine handled more external tasks related to the code, as the C part was less important in the project.

Once the roles were understood, we progressed gradually, either when we met at school or working remotely, mainly through Discord calls, with each person completing the tasks assigned to them. The code began to take shape until the final result was achieved.

1) Development:

For this project, we first created an AVL capable of reading data from a text file and inserting it into the tree. This step helped us better understand how files and C programs interact.

Next, we worked on the shell part. We started by filtering the lines of the CSV file with all the data based on the parameters defined by the user. The filtering varied according to the specified criteria. We analyzed and understood the different methods for identifying relevant lines so that only the ones we needed were kept in the file.

After filtering, we passed the data to our executable for processing, line by line. The filtered file contains only three columns. Our program checks that each item is a valid data , meaning an integer. Each value is then processed. We also implemented a search function to check if a node with the same ID already exists in the tree. If it does, we add the consumption values together.

Finally, we display the tree with the relevant data, such as the ID of each power plant, its capacity, and total consumption. This data is then exported to a result.csv file via redirection, which is renamed based on its specific characteristics (for example, lv_all.csv), making it easier to identify.

Then, we handled error messages, especially to signal cases where the user enters too many parameters or incorrect commands. Additionally, by adding timers at different points in the program and using large test files, we were able to identify and implement more efficient commands, which allowed us to optimize execution time and improve the overall performance of the program.

At the beginning of each compilation, the tests, last_result, and tmp directories are created if they don't already exist. Moreover, the tmp directory deletes all files generated from the last compilation. The files in the last_result directory are moved to the tests directory to save the previous results, and the last_result directory is cleared, ready to hold the new result file generated by the program. In case of an error, the last_result directory remains empty.

2) Problems Encountered and Functional Limits:

One of our main problems was figuring out if we were able to detect memory leaks. We finally solved this issue by enabling the "fsanitize=address" option during compilation. Another challenge was identifying errors in the filtered file sent to the C program, as well as returning a positive number while displaying an error message in the output. We eventually solved these issues by using clear and simple functions.

Overall, our program works correctly. However, if the CSV file provided as a parameter contains incorrect information on some lines (for example, 'hvb' lines), but we only process the 'hva' lines, the program does not stop and continues running because the filtered file sent to the C program only contains valid 'hva' lines.

Conclusion:

The C-Wire project allowed us to tackle the challenge of developing an efficient solution while ensuring optimal memory management and processing time.

By using a Shell script for data processing and a C program with optimized data structures, such as AVL trees, we were able to process large files smoothly and efficiently. Despite the technical challenges we faced, we managed to adapt. This required clear communication within the team, made easier by collaborative tools like Discord, and a smart distribution of tasks based on everyone's skills.

In the end, our solution provides a robust method for analyzing large amounts of data from CSV files structured according to the specifications. This project not only strengthened our technical skills in development and data management but also improved our ability to work as a team.