Rapport Informatique - Projet C-Wire

Informatique 3 - Romuald Grignon

Melanie Avel

Kahina Hammad

Ashana Maheswaran

PréIng 2 MI-2

2024 - 2025

Description de l’équipe et du sujet

Notre équipe est constituée de trois personnes : Kahina Hammad, Mélanie Avel et Ashana Maheswaran.

L’objectif de notre projet est de réaliser un programme permettant la synthèse de données d’un système de distribution d’électricité. Notre programme doit filtrer et traiter des données à l’aide d’un script Shell et un programme C.

Nous devions automatiser la création de fichiers contenant des données filtrées sur le type de station et de consommateurs que l’utilisateur a choisi.

Organisation de l’équipe

Le projet a commencé par la création d’un repository GitHub par Kahina pour faciliter la gestion et le partage des fichiers.

Un échange initial a été organisé en TD pour comprendre les fonctionnalités du projet et pour définir les rôles de chacune. GitHub a été utilisé pour suivre les tâches et les modifications apportées, et WhatsApp a servi de moyen pour une communication rapide.

Tout d’abord, Mélanie a commencé à coder les premières vérifications pour le script Shell tandis que Kahina et Ashana codaient les fonctions nécessaires pour les AVL.

Après avoir codé les fonctions permettant de trier les données dans le shell et les fonctions nécessaires pour les AVL. Nous avons travaillé sur la récupération des données du shell par le C. Ensuite, nous avons continué le Shell pour la création des fichiers. Lorsque nous avions fini d'ajouter les derniers détails, Ashana a codé pour la création des graphiques pour les lv all, Mélanie a fait le makefile et Kahina s’occupait du rapport et du readme.

Pendant les séances de TD, l’équipe avançait ensemble pour identifier et corriger les bugs en temps réel. Hors TD, chacune travaillait individuellement et partageait son travail sur GitHub pour un suivi régulier.

Problèmes rencontrés et limites

Tout d'abord, nous avions eu l'idée de passer par des fichiers pour la récupération des données. Mais notre professeur nous a indiqué que nous pouvions utiliser le scanf pour faciliter la récupération, c'est donc ce que nous avons fait.

Au début, lorsqu'on exécutait les programmes, les données obtenues n'étaient pas correctes. Puis, nous nous sommes rendus compte qu'il y a un nombre très long dans le fichier de données et nous avions utilisé des "int" dans le programme C. Nous avons ainsi changé en "long".

Mais cette donnée qui était grande nous a aussi causé un autre problème. En effet, nous devions classer le fichier lv all en fonction de leur quantité d'énergie consommée. La station avec la grande valeur nous donnait un résultat en notation scientifique. Nous n'arrivions donc pas à trier cette station.

Lorsque nous avions fini nos programmes, nous avons voulu le tester avec le vrai fichier de données. Mais, nous nous sommes rendus compte que nos programmes prennent du temps à s'exécuter : par exemple, pour les HVB comp, cela prenait 28 secondes et pour les LV (indiv/comp/all), cela prenait énormément de temps, donc nous stoppions toujours. Les professeurs sont venus regarder nos programmes et ont fait la remarque qu'il y a trop de commandes pour le script Shell. Donc, nous avons essayé de simplifier le plus possible le Shell en ajoutant des pipes, nous avions également essayer d'optimiser le plus possible le programme C. Puis, le problème persistant encore, nous avons regardé sur ChatGPT comment améliorer l'optimisation des programmes et l'IA nous a proposé une fonction facilitant la lecture des données du Shell par le C. Cette fonction a permis d'accélérer le temps d'exécution. Nous sommes maintenant en dessous de 10 secondes pour les lv all également pour les autres stations (voir capture d'écran pour le dossier tests).

Une fois nos programmes finis, nous avons décidé d'essayer de faire le bonus. Pour cela, nous avons fait des recherches sur internet sur gnuplot pour en connaître les bases. Quelques sites internet nous ont aidés et guidés pour la création de graphiques avec gnuplot. En effet, nous avons pu comprendre des bases comme mettre une légende aux axes. Puis, nous avions également quelques complications car le graphique n'est pas simple : en effet, l'axe des ordonnées devait aller du négatif au positif, donc l'axe des abscisses devait être plus haut. Nous avions d'autres problèmes comme les numéros des centrales en abscisses qui se chevauchaient. Ainsi, par

l’intermédiaire de l’IA, nous avos pu comprendre et utiliser quelques commandes résolvant nos problèmes tels que :

- set arrow from screen 0, first 0 to screen 1, first 0 nohead lw 1 lc rgb 'black'
- set xtics rotate by -45

Donc, nous avons réussi à créer des graphiques. Mais il nous reste tout de même un souci au niveau des couleurs. En effet, nous voulions que les valeurs négatives soient rouges et les positives en vert. Mais nous n’arrivons pas à faire une condition du type “si la valeur de la colonne du fichier csv est inférieur à 0, mettre en rouge”. Ainsi, nous avons juste écrit “faire un graphique avec la quatrième et première colonne et mettre en vert”, puis plus loin, nous avons mis la même chose avec le rouge car nous arrivions pas à mettre les conditions. Comme la dernière commande et pour le rouge, donc toutes les barres du graphique sont rouges (voir capture d’écran).
Voici notre ligne de code :

echo "plot 'tmp/intermediaire.csv' using **4:xtic(1) with boxes lc rgb 'green'** title 'Surproduction', 'tmp/intermediaire.csv' using **4:xtic(1) with boxes lc rgb 'red'** title 'Sous-production'" >> gnuplot_script

Donc, nous avons compris où se trouve l’erreur mais nous n’avons pas réussi à la corriger.

Résultats obtenus

Ainsi, nous avons des programmes fonctionnels qui répondent aux objectifs du projet. Selon le type de stations et la catégorie de clients que l'utilisateur souhaite examiner, notre programme crée un fichier csv avec les informations pertinentes et triées (capacité croissante) de manière structurée. Les fichiers intermédiaires et graphiques sont stockés dans leur dossier dédié. Pour le cas des lv all, nous avons un fichier avec les 20 postes LV triés et également un graphique les représentant. Enfin, nous avons optimisé au mieux le temps d'exécution des programmes.