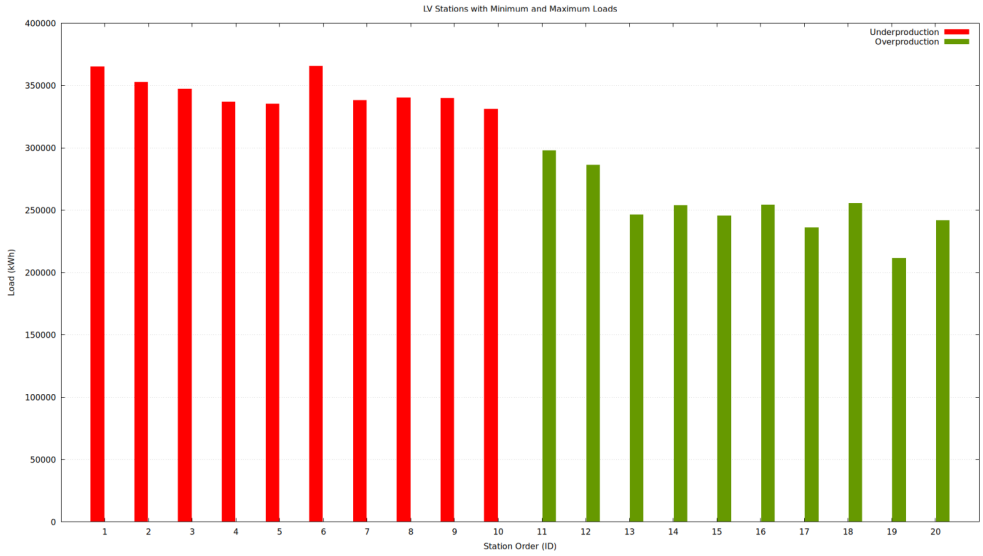
Informatique III

Encadré par M. Mohamed HADDACHE

MI-5 groupe A • Pré-ing 2

Rapport de projet

C-WIRE



Réalisé par :

Romain MICHAUT-JOYEUX

Nathan CHOUPIN

Guirec VETIER

Année universitaire 2024/2025

Sommaire

I. Introduction

Dans le cadre de notre année de Préing-2, nous avons réalisé un projet en informatique. Il s'agit de « **c-wire** », un programme visant à synthétiser les données d'un système de distribution électrique pour observer si des stations sont en situation de sous-production ou de sur-production. Cette distribution correspond à l'échelle d'un quart de la France. Notre objectif est de filtrer efficacement les données utiles et d'effectuer des calculs à partir de fichiers .csv selon l'analyse souhaitée, et notamment pour des fichiers lourds (≈ 260 Mo). À l'aide d'un script en shell, nous avons sélectionné les données nécessaires à cette analyse, puis nous avons écrit des algorithmes de calculs en C pour les traiter. Nous avons réussi à créer un programme performant et fonctionnel, qui traite les informations en un temps correct. On détaillera le processus que nous avons suivi pour atteindre cet objectif.

Dans un premier temps, nous parlerons de notre planning puis de la réalisation des tâches. Dans un deuxième temps, nous examinerons quelques difficultés rencontrées.

I. Réalisation des tâches

Étapes	Dates	Tâches principales
Phase 1 : Initialisation	16 - 22 novembre	- Analyse du cahier des charges. - Répartition des tâches. - Création du dépôt GitHub.
Phase 2 : Développement	23 novembre – 8 décembre	- Développement du script Shell. - Début de l'implémentation du programme en C (AVL, structures).
Phase 3 : Tests unitaires	9 - 13 décembre	- Test du script Shell avec des données partielles. - Validation des fonctions principales du C.
Phase 4 : Intégration	14 - 17 décembre	- Intégration script Shell et programme C. - Génération des fichiers de sortie.
Phase 5 : Finalisation	18 - 22 décembre	- Génération des graphiques (bonus). - Documentation (README, PDF). - Préparation des livrables.

Répartition des tâches

Romain : Gestion des données et script Shell

- Développement du script Shell pour le filtrage des données.
- Gestion des fichiers d'entrés, de sorties et des options utilisateur.
- Gestion des consommateur dans le C.
- Headers dans le C.
- Makefile.
- BONUS : Implémentation de Gnuplot dans le Shell et script Gnuplot

Nathan : Développement du programme en C

- Implémentation de l'arbre AVL et son équilibrage.
- Gestion du trie des stations dans le programme C.
- Optimisation et libération mémoire.
- Gestion des contenus des fichiers de sorties.
- BONUS : Implémentation de Gnuplot dans le programme C et script Gnuplot

Guirec : Visualisation et gestion des résultats

- Recherche sur Gnuplot

II. Problèmes rencontrées

Bien que nous ayons implémenté tous les éléments du cahier des charges, nous avons rencontré des problèmes au cours de la réalisation du projet. Premièrement, nous avons passé beaucoup de temps à travailler sur le passage des données filtrées par le script shell vers les scripts C. Notre première idée était d'envoyer ces données au programme via un flux de données (pipe) mais c'est un processus extrêmement long lorsqu'on transmet de grands volumes d'informations. C'est pourquoi nous avons opté pour l'utilisation de fichiers intermédiaires : le script en shell filtre les informations utiles et les place dans des fichiers temporaires qui seront lus par le programme en C pour effectuer des calculs. Cette technique est environ dix à vingt fois plus rapide que l'utilisation d'une pipe sur nos machines.

Pendant notre écriture des scripts en C, on a rencontré des problèmes concernant la gestion de la mémoire pour les tableaux servant à stocker les stations. Pour nous aider à comprendre nos erreurs de type « segmentation fault », nous avons utilisé une intelligence artificielle de façon minime. Nos tableaux prenaient une valeur en trop lors de leur remplissage, ce qui les rendait inutilisables dans l'ensemble du programme. Pour remédier à cette erreur d'inattention, on a supprimé cette valeur en surplus. Aussi, les valeurs de début, milieu et fin du tableau était erronée ce qui nous empêchait d'utiliser notre fonction de tri fusion.

Enfin, nous avons eu du mal à utiliser Gnuplot, car c'est un outil qui a une syntaxe assez complexe et qui demande des connaissances précises en création de graphique. On a réalisé notre programme à l'aide de la documentation en ligne, mais il était difficile de gérer l'affichage des couleurs sur

le graphique : le programme Gnuplot ne dispose pas d'un outil permettant de changer la couleur des barres du graphique à n'importe quel moment.
On a finalement trouvé un moyen de le coder nous-mêmes en créant un autre fichier intermédiaire à l'aide du programme en shell qui permet d'indiquer la couleur de chaque barre en fonction de la charge d'une station. La mise en place de cette fonction nous a fait perdre du temps à la fin du projet.

III. Tests

Voici les test.

Test 1 : HV-B company

```
nathan@DESKTOP-TJVC2N2:~/Informatique Projet$ ./c-wire.sh input/GROS.csv hvb comp

Starting data processing for station type hvb and consumer type comp...
...1. input/GROS.csv extracted in 1.93s.
...2. Writing the output data completed successfully in 0.00s.

Program completed successfully in 1.98s.
```

Test 2 : HV-A company centrale 2

```
nathan@DESKTOP-TJVC2N2:~/Informatique Projet$ ./c-wire.sh input/GROS.csv hva comp 2

Starting data processing for station type hva and consumer type comp...
...1. input/GROS.csv extracted in 1.79s.
...2. Writing the output data completed successfully in 0.00s.

Program completed successfully in 1.84s.
```

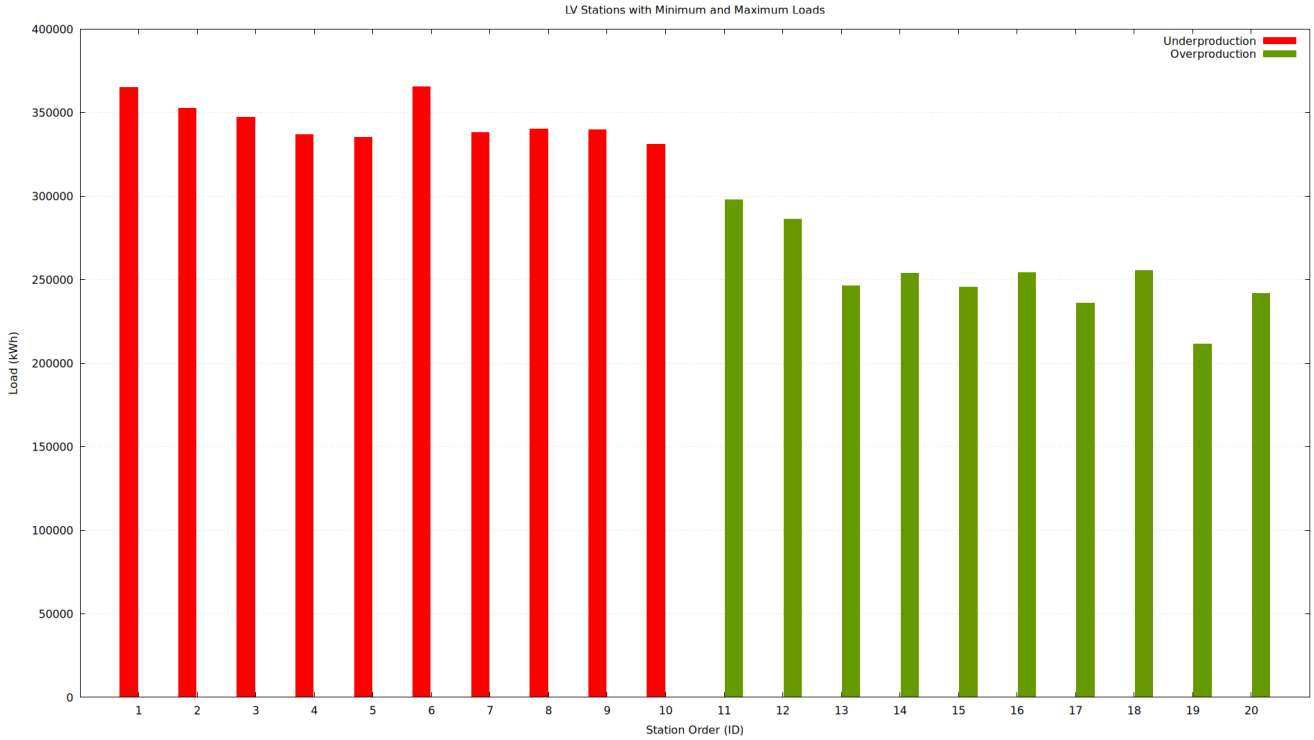
Test 3 : LV all (avec graph)

```
nathan@DESKTOP-TJVC2N2:~/Informatique Projet$ ./c-wire.sh input/GROS.csv lv all

Starting data processing for station type lv and consumer type all...
...1. input/GROS.csv extracted in 5.89s.
...2. Writing the output data completed successfully in 1.28s.
...3. Preparing temporary data file and making graphs...
...4. Graphs have been created in ./graphs in 0.73s.

Program completed successfully in 7.60s.
```

Graph LV all :



En rouge, les stations LV sur-chargées et en vert les stations sous-chargées.

Conclusion

En conclusion, nous avons réalisé un programme fonctionnel et robuste qui satisfait notre objectif : à savoir trier rigoureusement et rapidement de gros volumes de données. Nous sommes parvenus à implémenter chaque caractéristique et contrainte du cahier des charges, ce qui en fait un projet complet. Ce travail nous a permis de nous familiariser avec le shell et d'apprendre à traiter des fichiers de données à l'aide des commandes unix. Il était d'ailleurs intéressant de voir comment des programmes en différents langages peuvent communiquer entre eux et d'apprendre à réaliser des graphiques à l'aide de Gnuplot. Enfin, nous avons perfectionné notre maîtrise de l'outil git pour travailler en collaboration, ce qui est essentiel pour la suite de nos études.