RAPPORT DE PROJET : C-WILDWATER

Filière pré-ing2 – Année 2025-2026

- Yannis LEROY
- Ilyes MOUDAKKIR
- Mehdi ABDERRAHMANE

Lien du dépôt GitHub : https://github.com/Yannis1307/Projet-info-MI2_K

1. Introduction

Ce projet, réalisé dans le cadre du module d'informatique, vise à traiter un volume important de données (fichier .dat de plus de 8 millions de lignes) modélisant le réseau de distribution d'eau en France. L'objectif était de développer une application performante capable de générer des statistiques (histogrammes) et de simuler des pertes en eau sur un réseau complexe.

La solution technique combine l'efficacité d'un script Shell pour l'automatisation et le filtrage des données, et le langage C pour les algorithmes tels que les AVL.

2. Organisation et Répartition des tâches

Le projet a été réalisé par une équipe de 3 personnes sur une durée totale de 3 semaines. Nous avons adopté une organisation claire avec des points de mise en commun réguliers pour le débogage global et la mise en commun du code.

2.1 Tâches Communes

Tout le groupe a participé aux phases suivantes du projet :

- Analyse du sujet et définition des structures de données.
- Mise en place de l'environnement de développement (Dépôt Git, Makefile).
- Débogage global et tests d'intégration (connexion entre le Shell et le C).
- Rédaction du rapport pdf et ReadMe.

2.2 Ressenti initial

Lors du premier td d’informatique on a pu découvrir le sujet, qui au premier abord nous paraissait peu clair, on avait du mal à savoir par où commencer afin de se répartir les tâches. On a passé l’heure de td à tenter de comprendre le sujet en posant des questions à nos camarades et à notre professeur, suite à cela nous avons réussi à nous répartir les tâches.

2.3 Responsabilités Spécifiques

- **Yannis (Partie Shell) :**
    - Développement du script myScript.sh.
    - Gestion des arguments et sécurisation des entrées utilisateur.

    - Optimisation des performances via le pré-filtrage awk (réduisant le temps de traitement de plusieurs minutes à quelques secondes).
    - Intégration de Gnuplot pour la génération des graphiques.
- **Ilyes & Mehdi (Partie C) :**
    - Développement des modules C (Histo et Leaks).
    - Implémentation de l'AVL pour le tri et le stockage des usines (Histogramme).
    - Modélisation du réseau sous forme de Graphe pour le suivi des connexions (Leaks).
    - Implémentation de l'algorithme récursif de calcul de fuites avec gestion des sources multiples.
    - Gestion de la mémoire dynamique.

3. Planning de réalisation

Le projet s'est déroulé sur 3 semaines, structurées comme suit :

- **Semaine 1 : Architecture et Mise en place**
    - Création du dépôt Git et analyse du fichier data.dat.
    - Écriture du parser CSV en C.
    - Mise en place du Makefile et du squelette du script Shell.
- **Semaine 2 : Développement de l'histogramme**
    - Implémentation des structures AVL.
    - Développement de la fonctionnalité histo (calculs Max, Src, Real).
    - Premiers tests de génération de fichiers .dat et intégration gnuplot.
- **Semaine 3 : Développement de la partie Leaks et Optimisation**
    - Développement du module leaks : construction du graphe en mémoire.
    - Mise au point de l'algorithme récursif de parcours en profondeur.
    - **Phase d'optimisation majeure :** Modification du Shell pour filtrer les données en amont afin d'accélérer le traitement des fuites.
    - Tests finaux, génération des preuves (dossier tests) et rédaction du rapport.

4. Problèmes rencontrés

Nous avons rencontré plusieurs problèmes pendant la réalisation de notre projet,tout d'abord lors du changement de la consigne, les arguments de notre programme shell n'étaient pas dans le bon ordre ce qui a du être changé par la suite pour respecter la consigne.Puis lors de la génération de nos premiers histogrammes nous ne parvenions pas à récupérer uniquement les lignes des usines et l'histogramme affichait des stockages nous avons donc modifier notre code afin d'être sûrs d'avoir uniquement des usines dans notre histogramme.

5. Limitations Fonctionnelles

À la date de rendu, l'état du projet est le suivant :

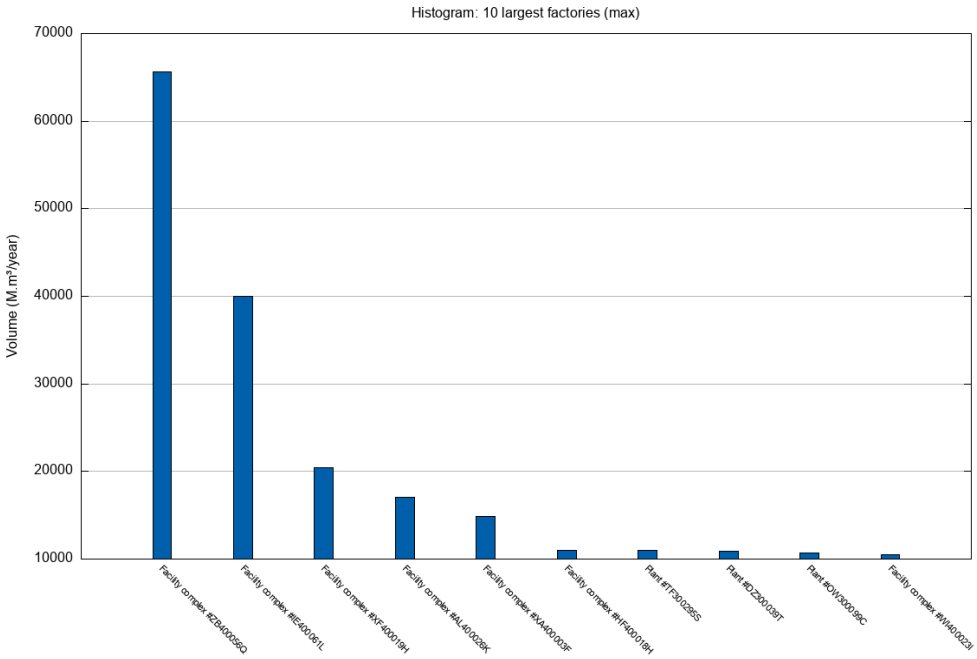
- **Fonctionnalités implémentées et valides :**
    - Compilation automatique et robuste (Makefile).

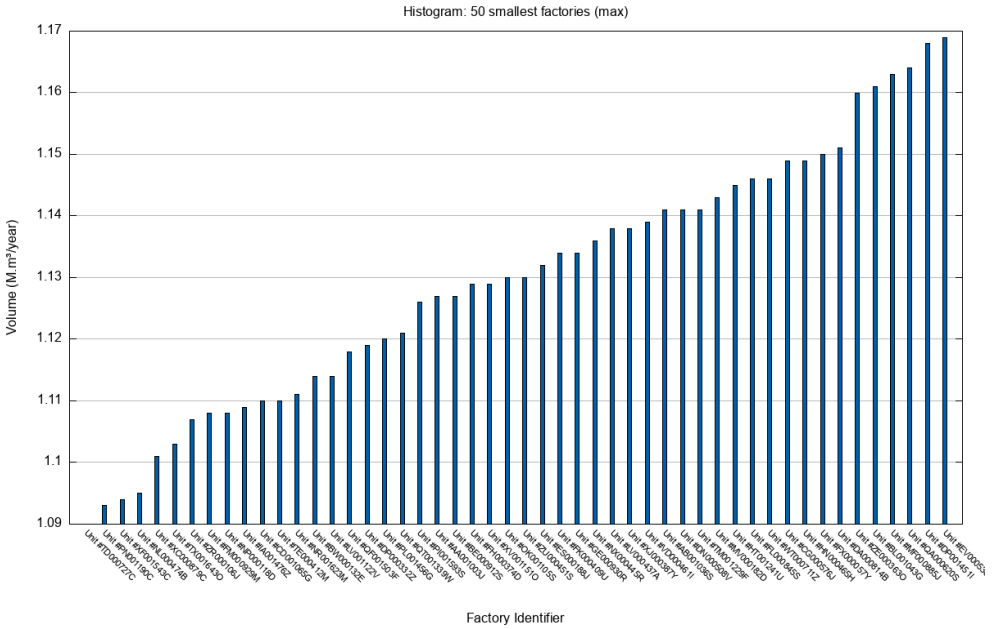
    - Commande histo : Génère correctement les fichiers .dat et les images .png (min50 et max10) pour les trois modes.
    - Commande leaks : Calcule précisément le volume de perte en prenant en compte les sources multiples et l'historique des requêtes (ajout dans leaks.dat sans écrasement des anciennes requêtes).
    - Gestion des erreurs : Le programme ne plante pas (pas de segfault) et rejette les arguments invalides (identifiant erroné ou inexistant).
- **Limitations :**
    - Le bonus n'a pas été implémenté par manque de temps.
    - L'affichage des graphiques dépend de la présence de gnuplot sur la machine.

6. Présentation des résultats

Le dossier tests présent sur le dépôt contient les preuves de fonctionnement :

1. **Histogrammes :** Les images histo_max_max10.png et histo_max_min50.png montrent la répartition des capacités des usines.





2. **Calcul de fuites :** Le fichier leaks.dat contient l'historique des tests effectués.
    - *Exemple :* Pour l'usine Plant #VC300055Q, le programme a calculé un volume de perte de **0.777568 M.m³**.