

Présentation du projet Permis C

Réalisé par Paul P., Yani A., Mathis M.

Table des matières

Répartition des tâches	1
Planning de réalisation	1
Jalon 1 : Préparation du terrain	1
Jalon 2 : Les premiers traitements fonctionnels, de la commande jusqu'au graphique	1
Jalon 3 : Un programme complet !	2
Jalon 4 : Polissage final	2
Jalon 5 : Toujours plus vite ?!	2
Jalon 6 : BEAUCOUP TROP RAPIDE !! (Bonus, non réalisé pour le moment)	2
Limitations	2
Exemples de commandes et leur résultats	3
Lancement du traitement D1	3
Lancement du traitement D2	4
Lancement du traitement L	5
Lancement du traitement T	6
Lancement du traitement S	7
Lancement de tous les traitements avec la vitesse maximale	8

Répartition des tâches

- Paul P. : scripts Gnuplot pour tous les traitements, traitements D1, D2 et L sous Awk
- Yani A. : lecture du fichier CSV, implémentation AVL, Makefile, script Bash, bonus
- Mathis M. : traitements T et S en C

Planning de réalisation

Jalon 1 : Préparation du terrain

- **Programme C**
 - Programme en C basique [YA]
 - Makefile simple
 - Lecture du fichier CSV
 - Implémentation complète et optimisée des arbres AVL (hors suppression)
 - Lecture des arguments et lancement du traitement (avec une implémentation vide)
- **Scripts (Shell, Awk, Gnuplot)**
 - Traitements avec Awk [PP]
 - Traitement D1
 - Traitement D2
 - Traitement L

Jalon 2 : Les premiers traitements fonctionnels, de la commande jusqu'au graphique

- **Programme C**
 - Traitement D1 en C, avec AVL (bonus, à but d'exemple) [YA]
 - Traitement T en C [MM]

- **Scripts (Shell, Awk, Gnuplot)**
 - Script Bash pour compiler, lancer les traitements, et générer les graphiques [YA]
 - Scripts Gnuplot pour les premiers traitements [PP]
 - Traitement D1
 - Traitement D2
 - Traitement L
 - Traitement T

Jalon 3 : Un programme complet !

- **Programme C**
 - Traitement S en C [MM]
 - Optimisation de la lecture CSV et meilleure gestion des erreurs (bonus) [YA]
 - Traitement D1 en C, avec table de hachage, expérimental (bonus) [YA]
- **Scripts (Shell, Awk, Gnuplot)**
 - Script Gnuplot pour le traitement S [PP]
 - Compatibilité macOS du script Bash (bonus) [PP & YA]
 - Ajout de l'option `-Q/--quick` pour choisir les algorithmes utilisés (bonus) [YA]

Jalon 4 : Polissage final

- **Programme C**
 - Vérification entière du code, renforcement de la robustesse et de la gestion d'erreur [YA]
 - Algorithme expérimental pour le traitement T, plus rapide, avec des AVL (bonus) [YA]
- **Scripts (Shell, Awk, Gnuplot)**
 - Embellissement de l'expérience utilisateur du script Bash (bonus) [?]
 - Messages colorés, avec des emojis
 - Liens cliquables vers les graphiques et autres fichiers
 - Vérifications supplémentaires sur les dépendances requises (gnuplot, make)

Jalon 5 : Toujours plus vite ?!

- **Programme C**
 - Tous les traitements expérimentaux en C, avec des AVL et des tables de hachage (pour `-Q1`) [YA]

Jalon 6 : BEAUCOUP TROP RAPIDE !! (Bonus, non réalisé pour le moment)

- **Programme C**
 - Implémentation multi-cœurs des traitements, avec pthreads [YA]

Limitations

Les arbres AVL ont une hauteur de maximum 64 nœuds, soit 2^{64} éléments dans le cas optimiste, et 2^{61} dans le cas pessimiste (si $|\text{équilibre}| \geq 2$).

Les lignes du fichier CSV doivent faire moins de $2^{17} = 131072$ caractères.

Les identifiants de trajet et d'étape ne peuvent pas excéder le maximum de $2^{32} - 1 = 2147483647$.

Exemples de commandes et leur résultats

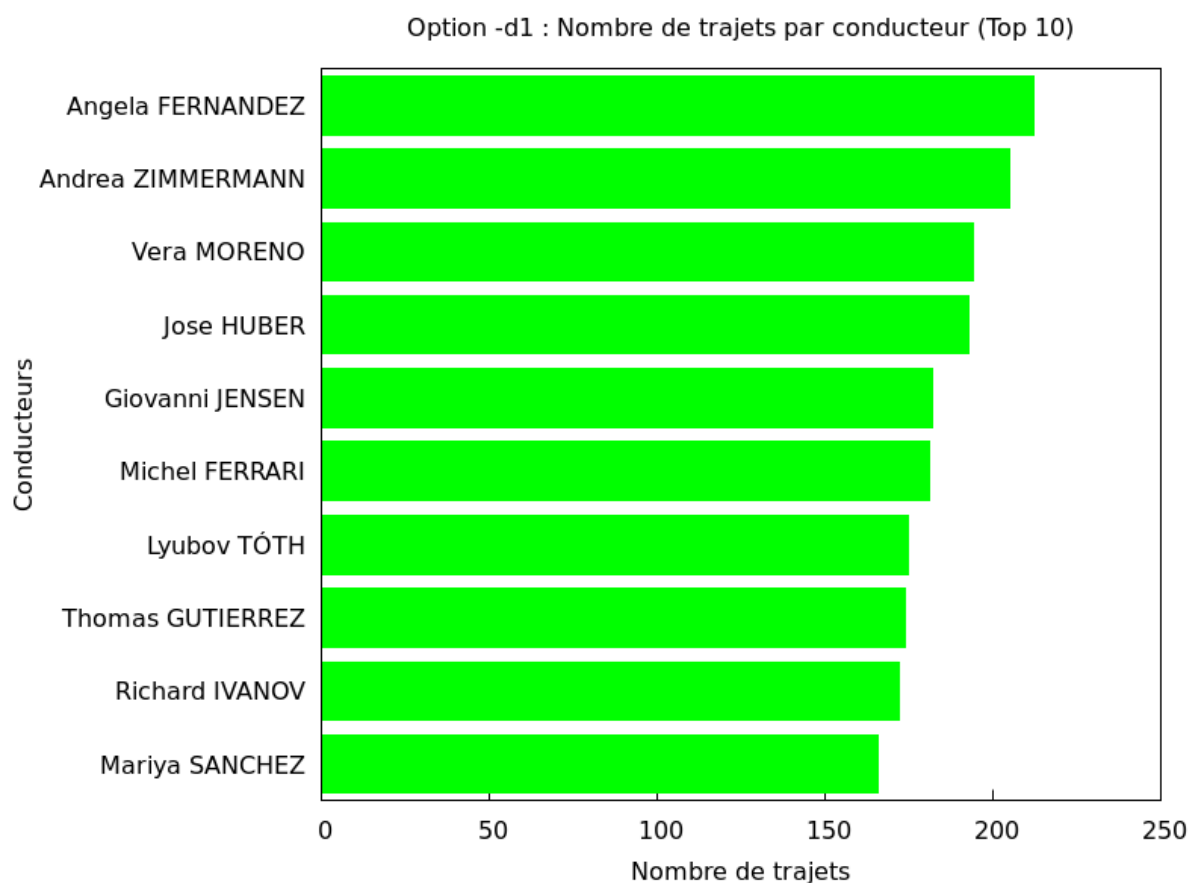
Lancement du traitement D1

```
./PermisC.sh data.csv -d1
```

Sortie :

```
🏠 | [x] Compilation de l'exécutable PermisC... Terminé !  
⚙️ | [x] Traitement D1 en cours... Terminé en 7077 ms !  
📊 | [x] Génération des graphiques... Terminé !  
🚗 | 📢 Programme terminé ! Les graphiques sont disponibles dans le dossier « images ».  
📁 | 📄 Ouvrir le graphique D1
```

En cliquant sur le lien « Ouvrir le graphique D1 », le graphique du dossier « images » s'ouvre :



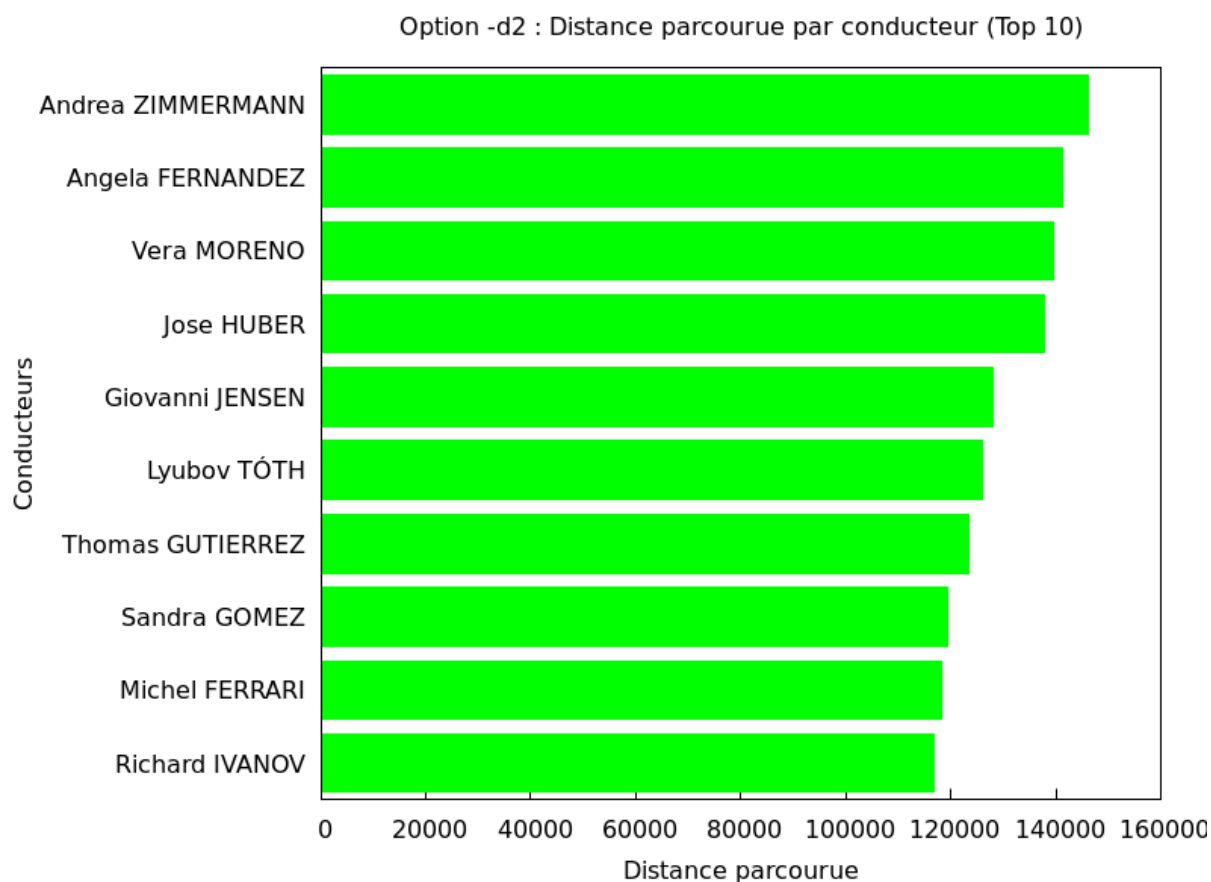
Lancement du traitement D2

```
./PermisC.sh data.csv -d2
```

Sortie :

```
⚙️ | ☒ Traitement D2 en cours... Terminé en 3127 ms !  
📊 | ☒ Génération des graphiques... Terminé !  
🚗 | Programme terminé ! Les graphiques sont disponibles dans le dossier « images ».  
📁 | Ouvrir le graphique D2
```

Le graphique généré est (encore) le suivant :



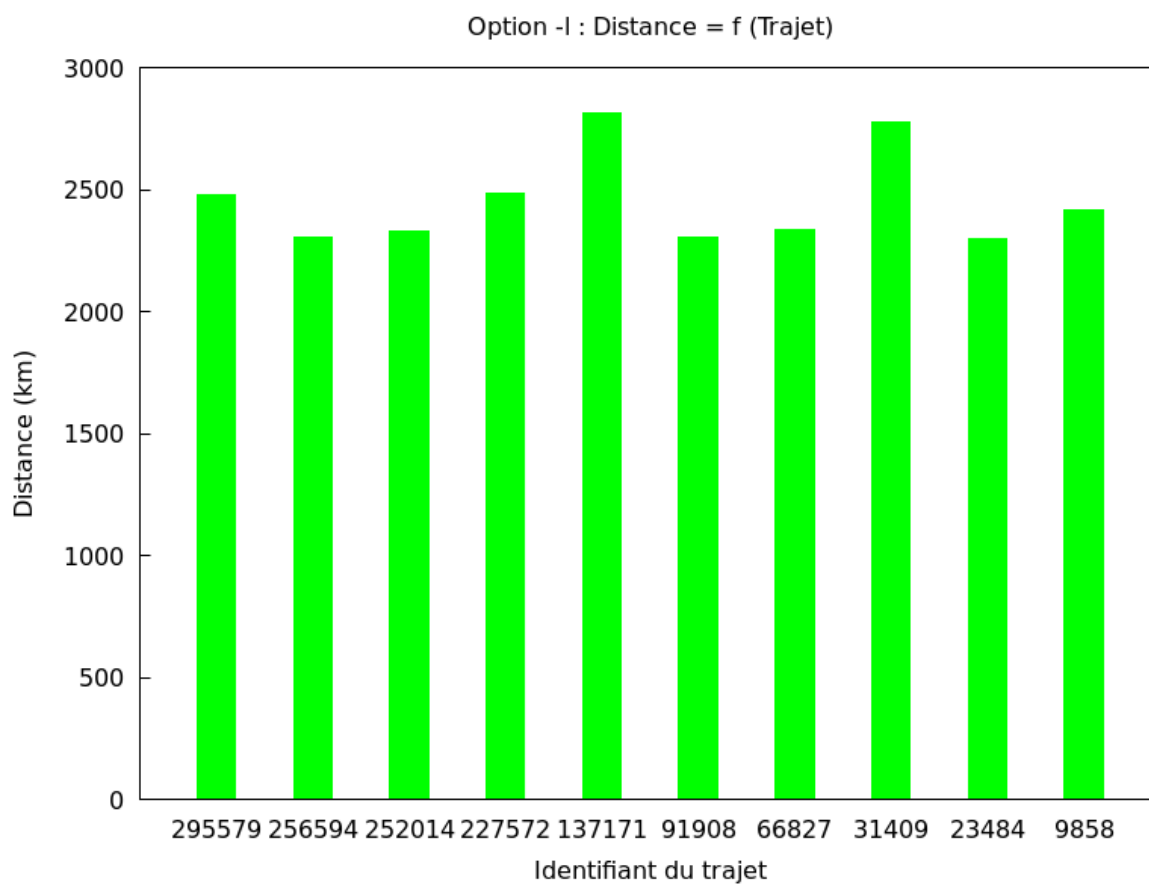
Lancement du traitement L

```
./PermisC.sh data.csv -l
```

Sortie :

```
⚙️ | ☒ Traitement L en cours... Terminé en 7783 ms !  
📊 | ☒ Génération des graphiques... Terminé !  
🖨️ | Programme terminé ! Les graphiques sont disponibles dans le dossier « images ».  
🔗 | Ouvrir le graphique L
```

Le graphique généré est toujours le suivant :



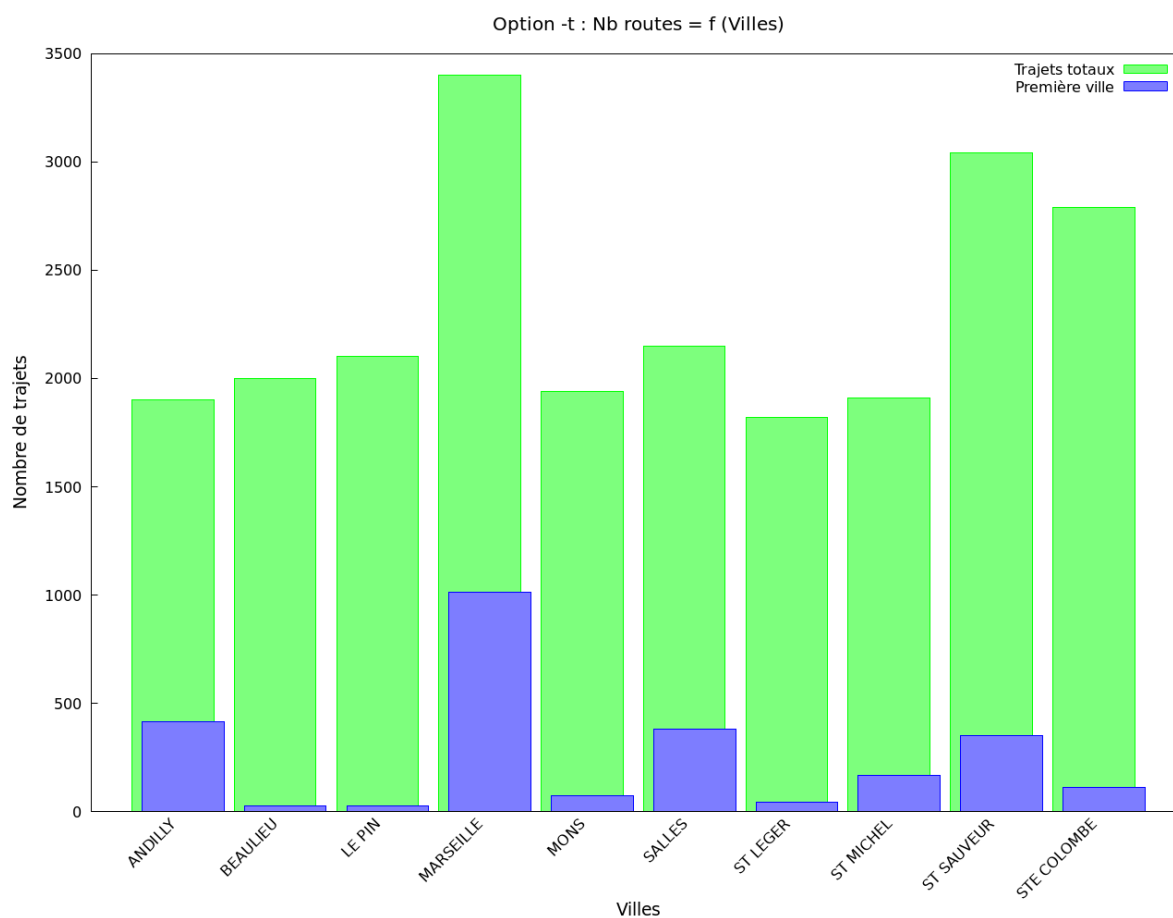
Lancement du traitement T

```
./PermisC.sh data.csv -t
```

Sortie :

```
⚙️ | ☒ Traitement T en cours... Terminé en 19101 ms !  
📊 | ☒ Génération des graphiques... Terminé !  
🚗 | Programme terminé ! Les graphiques sont disponibles dans le dossier « images ».  
➡️ | Ouvrir le graphique T
```

Le graphique généré est (encore et toujours) le suivant :



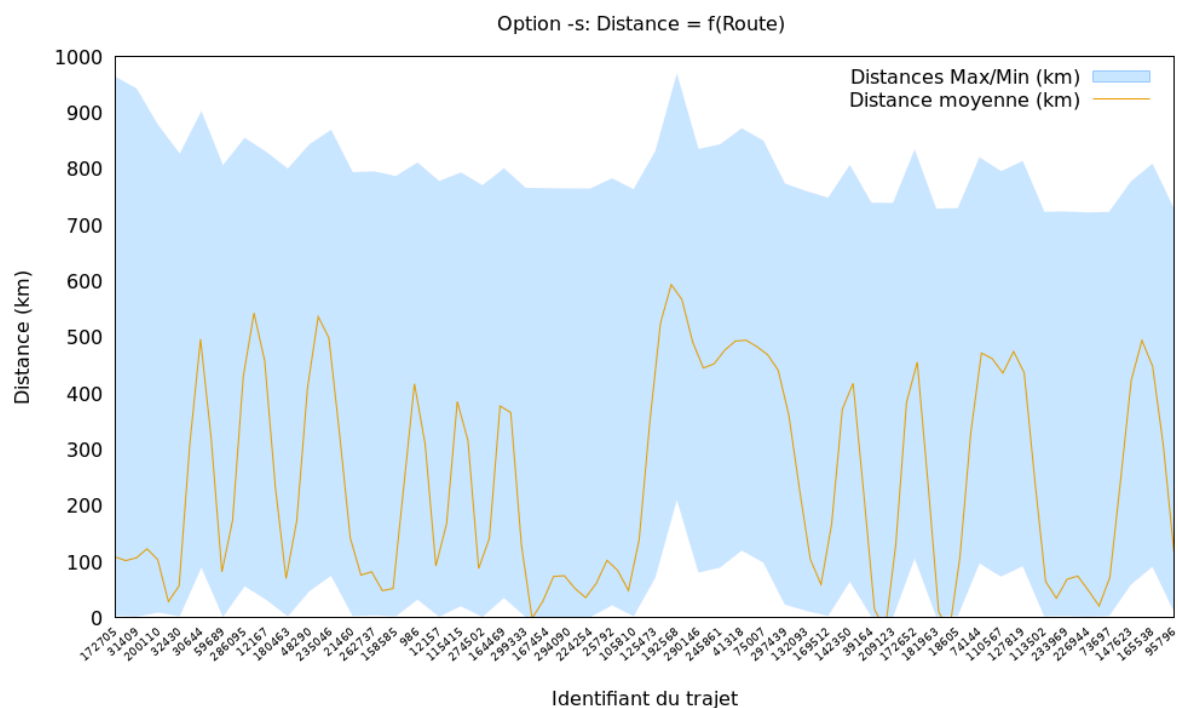
Lancement du traitement S

```
./PermisC.sh data.csv -s
```

Sortie :

```
⚙️ | ☒ Traitement S en cours... Terminé en 4037 ms !  
📊 | ☒ Génération des graphiques... Terminé !  
🚗 | Programme terminé ! Les graphiques sont disponibles dans le dossier « images ».  
📁 | Ouvrir le graphique S
```

Le graphique généré est celui-là (encore une fois) :



Lancement de tous les traitements avec la vitesse maximale

```
./PermisC.sh data.csv --all -Q2
```

Sortie :

```
🔧 | [x] Compilation de l'exécutable PermisC... Terminé !
⚙️ | [x] Traitement D1 en cours... Terminé en 568 ms !
⚙️ | [x] Traitement D2 en cours... Terminé en 455 ms !
⚙️ | [x] Traitement L en cours... Terminé en 382 ms !
⚙️ | [x] Traitement T en cours... Terminé en 1197 ms !
⚙️ | [x] Traitement S en cours... Terminé en 440 ms !
📊 | [x] Génération des graphiques... Terminé !
🚗 Programme terminé ! Les graphiques sont disponibles dans le dossier « images ».
📂 Ouvrir le graphique D1
📂 Ouvrir le graphique D2
📂 Ouvrir le graphique L
📂 Ouvrir le graphique T
📂 Ouvrir le graphique S
```

Les graphiques générés sont les mêmes que les précédents. (Ils sont toujours dans le dossier « images ».)

L'argument `--excès-de-vitesse` fait la même chose que `-Q2`, avec une petite différence...

Testez-le... 😊