



GESTION INTELLIGENTE DE PARKINGS

LIFPROJET RAPPORT

DÉC 2022 // MUNOZ MATÉO
FERRER RAPHAEL
BOULET BENJAMIN

P2002495
P1908300
P2006010

ENCADRANT :
AKNINE SAMIR

Sommaire

1. Introduction	-----	3
2. Organisation	-----	4
3. Fonctionnalités	-----	5
4. Difficultés rencontrées	-----	12
5. Conclusions	-----	13

Introduction

Ce projet consiste au développement d'une simulation en C++, dans le cadre de l'UE LIFPROJET, lors de la troisième année de Licence Informatique à Lyon 1.

Pour ce projet, l'équipe de développement est constituée de 3 étudiants ;

- Ferrer Raphael
- Munoz Matéo
- Boulet Benjamin

L'objectif de ce projet est de proposer une méthode distribuée pour l'affectation des places de stationnement à des véhicules intelligents.

Cela passe par l'implémentation de l'ensemble des comportements nécessaires aux véhicules pour interagir avec les parkings et négocier le tarif de stationnement, qui est décidé dynamiquement en fonction de différents facteurs tels que la durée de stationnement ou bien le taux d'occupation du parking par exemple.

Notre but a été de concevoir une simulation au plus réaliste afin d'avoir une certaine cohérence dans les données, mais également de permettre à l'utilisateur de la simulation de pouvoir visualiser efficacement ces informations à l'aide de différents outils et fonctionnalités. Fonctionnalités permettant également une certaine intuitivité dans l'utilisation du programme, mais aussi permettant d'avoir un rendu plus attrayant.

Le projet a été développé sous linux et utilise les librairies, SDL2 qui permet l'affichage de la simulation et GNUPLOT qui permet la création et l'affichage de graphiques pour les données.

Organisation

Le projet à durer 3 mois au total. Sa date de début était le 15 septembre et sa date de fin, le 16 décembre.

Nous avons tout d'abord commencé par réfléchir à quelle classe nous aurions besoin lors de ce projet. Nous avons pu réaliser une première ébauche du diagramme de classes sur ces idées.

Lors du premier mois, nous avons codé les classes principales pour la voiture, l'utilisateur, le parking, les places et l'environnement, ainsi que des classes intermédiaires comme `vec2` (vecteur 2D). Nous avons également réfléchi à la manière d'organiser ces idées et de commencer à travailler sur l'affichage graphique en utilisant SDL2.

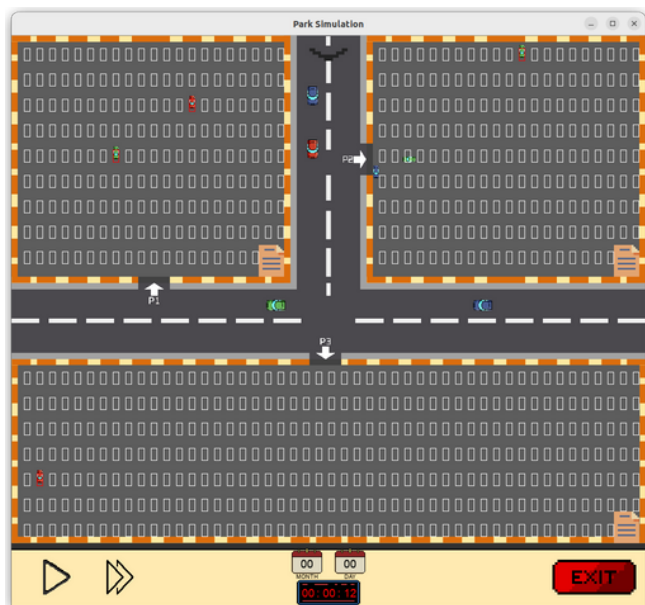
Après avoir codé une base solide pour le projet, nous nous sommes intéressés au système de conversation et de messagerie entre les voitures et les parkings, ainsi qu'à la stratégie de communication entre les deux IA. En parallèle, un membre de l'équipe a mis en place un système de recherche de chemin (pathfinding) pour améliorer la visualisation des voitures en mouvement et en stationnement. À la fin du deuxième mois, le système de conversation était terminé.

Le dernier mois a été consacré au peaufinage du programme. Nous avons amélioré l'affichage graphique pour mieux visualiser toutes les données, notamment en ajoutant des graphiques pour montrer le succès moyen entre tous les parkings, le profit de chaque parking et le nombre de places occupées dans le temps, ainsi que l'évolution du prix de départ de chaque parking. Nous avons également amélioré la stratégie de communication, par exemple en ajoutant une évolution des prix en fonction du nombre de visites effectuées par un utilisateur dans une voiture. Enfin, nous avons nettoyé en profondeur le programme pour le rendre plus lisible et cohérent, et nous avons résolu les problèmes de mémoire importants en mettant en place un système de sérialisation des données et en effectuant une recherche approfondie dans le programme.

Fonctionnalités

Plusieurs fonctionnalités ont été implémentées dans ce projet.

Tout d'abord, comme indiqué précédemment, un système de pathfinding a été ajouté afin d'avoir une clarté de l'information au niveau des voitures en mouvement et pour le stationnement.



Le système de pathfinding sert à ce que les voitures ne partent pas dans tous les sens, mais plutôt qu'elles essayent de reproduire un trafic routier afin d'avoir plus de clarté dans l'affichage.



On peut voir ici le chemin défini pour la voiture après l'appel de la fonction de pathfinding.

Ensuite, le système de conversation. Les conversations sont stockées dans le dossier *data/logs* et permettent de visualiser l'échange de messages entre la voiture qui a fait une requête de stationnement et les parkings.

Voici deux exemples de conversation : la voiture avec l'utilisateur 3335 va envoyer un message de demande de stationnement aux parkings, ici 1 et 2, et une "bataille" de négociations va s'engager.

Sur la conversation de gauche, la voiture va refuser alors que sur la droite, elle va accepter :

```
data > logs > E Conversation U1449P1_206.txt
1  messageNumber : 0
2  sender : User_1449
3  recipient : Unknown car park
4  date : 329
5  subject : CALL
6  price : -1
7  -----
8  messageNumber : 1
9  sender : Car_park_0
10 recipient : User_1449
11 date : 329
12 subject : OFFER
13 price : 6.72005
14 -----
15 messageNumber : 2
16 sender : User_1449
17 recipient : Car_park_0
18 date : 329
19 subject : COUNTER_OFFER
20 price : 3.26
21 -----
22 messageNumber : 3
23 sender : Car_park_0
24 recipient : User_1449
25 date : 329
26 subject : COUNTER_OFFER
27 price : 4.98
28 -----
```

Appel d'offre pour le stationnement par la voiture

Première offre du parking

Contre-offre de la voiture

Bataille de contre-offres

```
56 -----
57 messageNumber : 8
58 sender : User_1449
59 recipient : Car_park_0
60 date : 329
61 subject : ACCEPT
62 price : 5.99239
63 -----
64 messageNumber : 9
65 sender : Car_park_0
66 recipient : User_1449
67 date : 329
68 subject : ACCEPT
69 price : 5.99239
70 -----
71 messageNumber : 10
72 sender : User_1449
73 recipient : Car_park_0
74 date : 329
75 subject : RENOUNCE
76 price : -1
77 -----
78 messageNumber : 11
79 sender : Car_Park_0
80 recipient : User_1449
81 date : 329
82 subject : ABORT
83 price : -1
84 -----
```

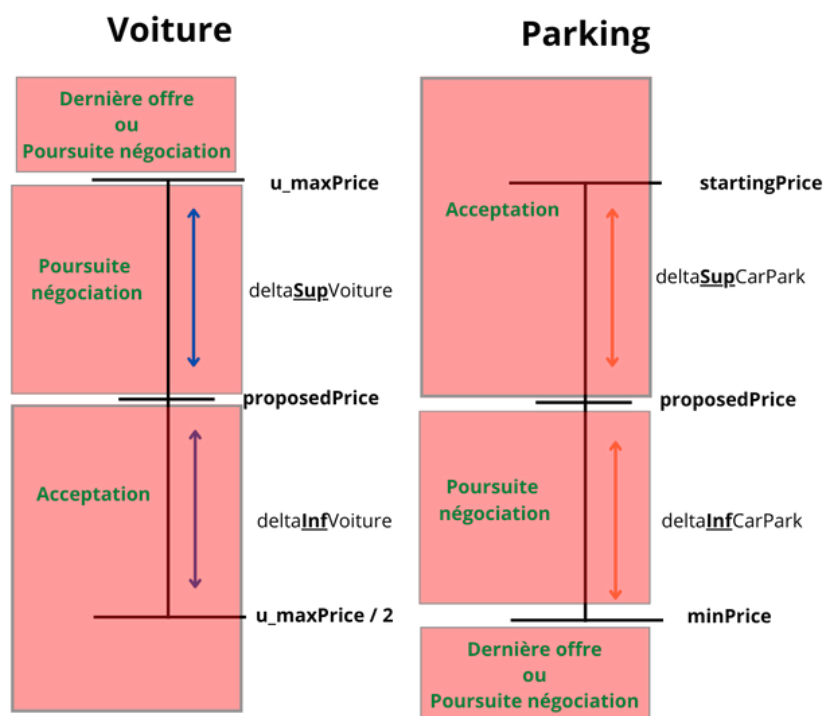
La voiture fini par accepter

La voiture refuse finalement car au même moment elle accepte un prix plus bas, dans la conversation de droite

```
data > logs > E Conversation U1449P1_206.txt
1  messageNumber : 0
2  sender : User_1449
3  recipient : Unknown car park
4  date : 329
5  subject : CALL
6  price : -1
7  -----
8  messageNumber : 1
9  sender : Car_park_1
10 recipient : User_1449
11 date : 329
12 subject : OFFER
13 price : 5.59064
14 -----
15 messageNumber : 2
16 sender : User_1449
17 recipient : Car_park_1
18 date : 329
19 subject : COUNTER_OFFER
20 price : 3.15
21 -----
22 messageNumber : 3
23 sender : Car_park_1
24 recipient : User_1449
25 date : 329
26 subject : COUNTER_OFFER
27 price : 4.36
28 -----
29 messageNumber : 4
30 sender : User_1449
31 recipient : Car_park_1
32 date : 329
33 subject : ACCEPT
34 price : 4.36
35 -----
36 messageNumber : 5
37 sender : Car_park_1
38 recipient : User_1449
39 date : 329
40 subject : ACCEPT
41 price : 4.36
42 -----
43 messageNumber : 6
44 sender : User_1449
45 recipient : Car_park_1
46 date : 329
47 subject : CONFIRM_ACCEPT
48 price : 4.36
49 -----
50 messageNumber : 7
51 sender : Car_Park_1
52 recipient : User_1449
53 date : 329
54 subject : OK_TO_PARK
55 price : 4.36
56 -----
```

Voici trois schéma montrant la stratégie de négociations entre les voitures et les parkings.

Actions de négociation en fonction de l'intervalle de prix



Déroulement de la 1^{ère} phase d'une négociation (fonction *managingConversation*)

Légende :

- Envoyé par la voiture
- Envoyé par le parking

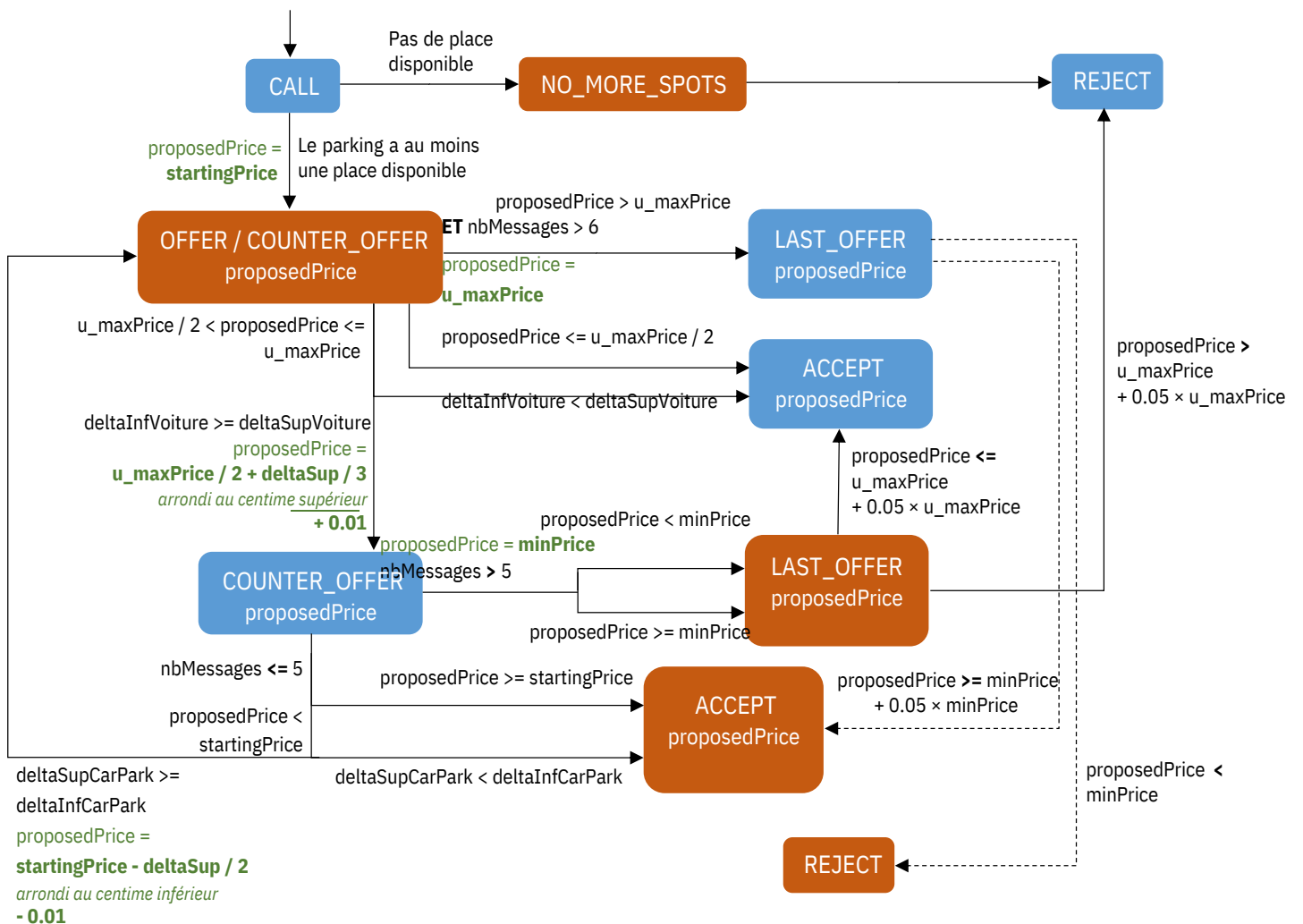
Annotation des flèches :

- En noir : condition
- En vert : affectation

Lexique :

- $[u_maxPrice / 2 ; u_maxPrice]$: intervalle de prix proposés par la voiture
- $[minPrice ; startingPrice]$: intervalle de prix proposés par le parking
- *proposedPrice* : prix courant
- $deltaInfVoiture = |u_maxPrice / 2 - proposedPrice| \delta_{supVoiture}$
- $= |proposedPrice - u_maxPrice|$
- $deltaSupCarPark = |startingPrice - proposedPrice|$
- $deltaInfCarPark = |proposedPrice - minPrice|$
- *nbMessage* : nombre de messages échangés

Début de la conversation



Déroulement de la 2e phase d'une négociation (fonction *confirmConversation*)

Légende :

■ Envoyé par la voiture

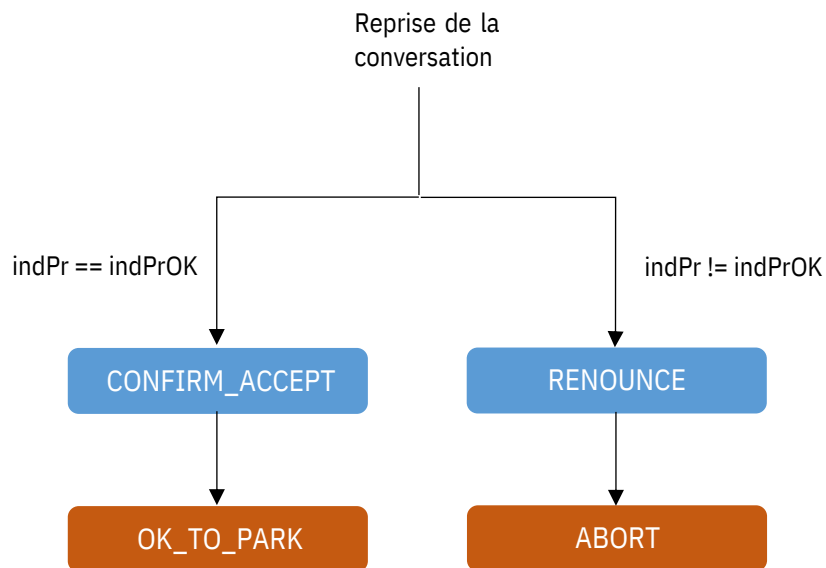
■ Envoyé par le parking

Lexique :

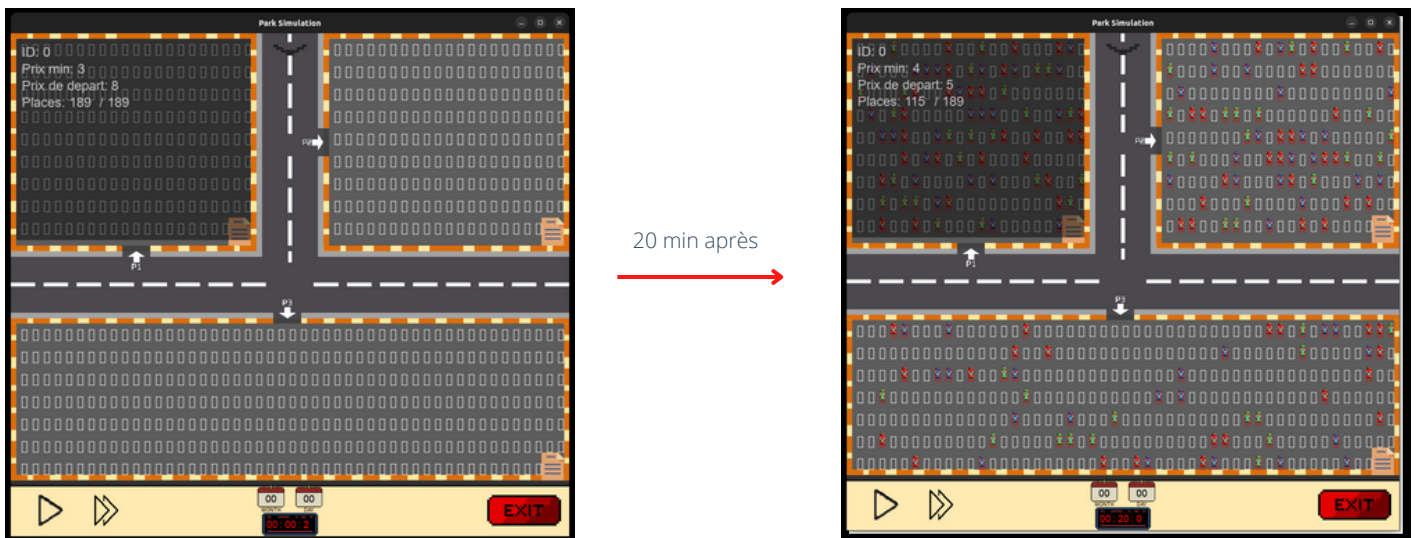
- *indPrOK* : indice passé en paramètre de *confirmConversation* et correspondant à l'ID de l'un des 3 parkings.
- *indPr* : ID du parking auquel s'adresse le message.

Annotation des flèches :

■ En noir : condition



Enfin, une certaine adaptabilité à l'environnement a été implémentée au parking. En effet, au fil du temps, les parkings vont faire évoluer leur prix en fonction de certains facteurs afin de satisfaire la demande.



On voit bien que le parking p1 a fait évoluer son prix de départ proposé pour passer de 8 à 5 , afin de faire rentrer le plus de voiture.

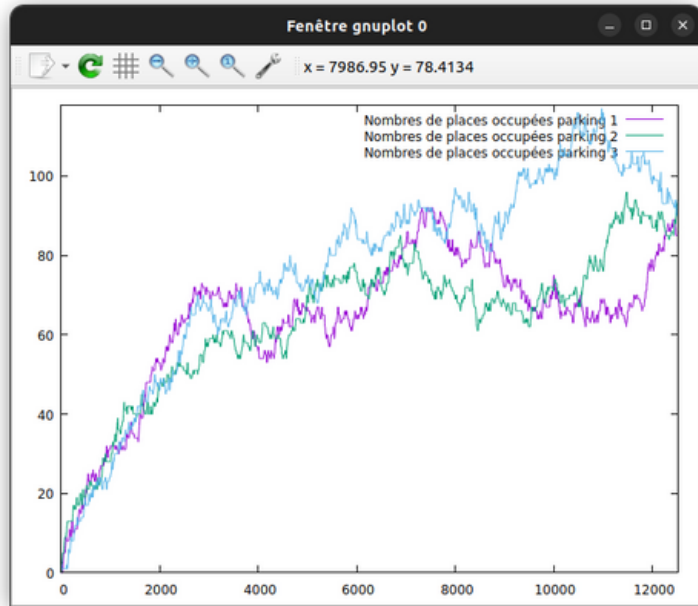
Ensuite, comme énoncer, pour une meilleure visibilité des données nous avons implémenter l'affichage de différentes données tel que,

L'utilisateur de la voiture afin de voir son id, son temps de stationnement ou bien son prix maximum qu'il va accepter

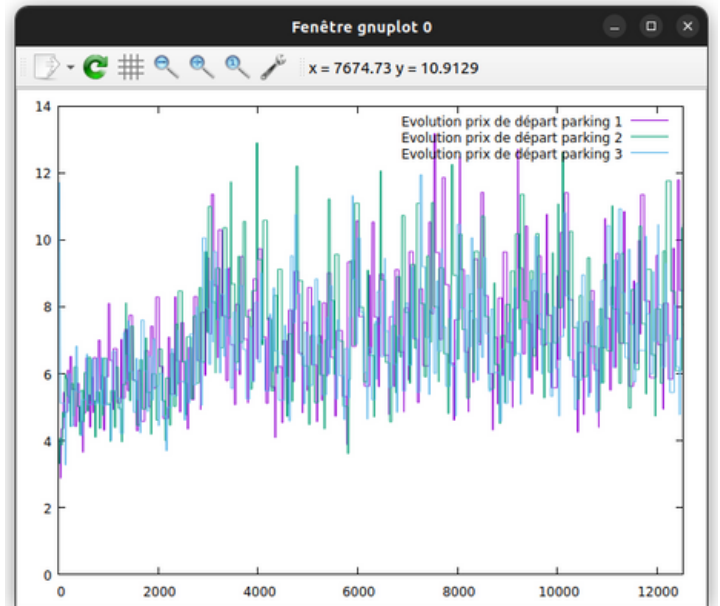
Le parking afin de visualiser ses informations comme son prix minimum par exemple



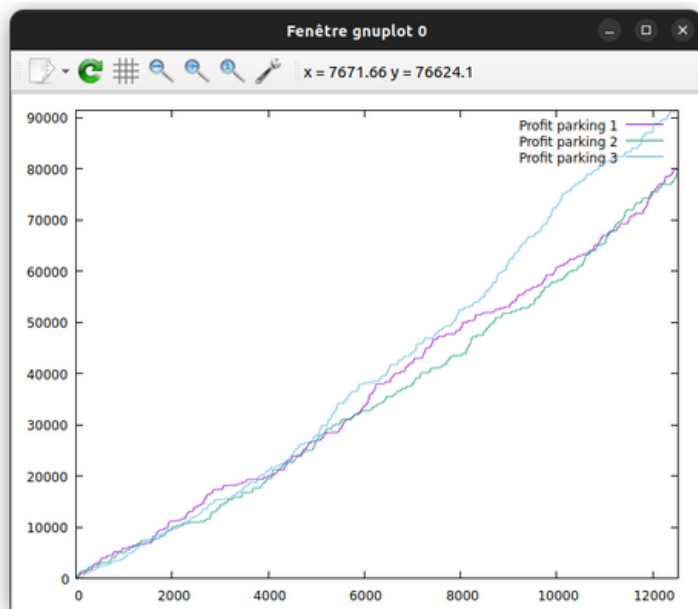
Mais également la création et l'affichage de graphiques pour une analyse plus complète des données :



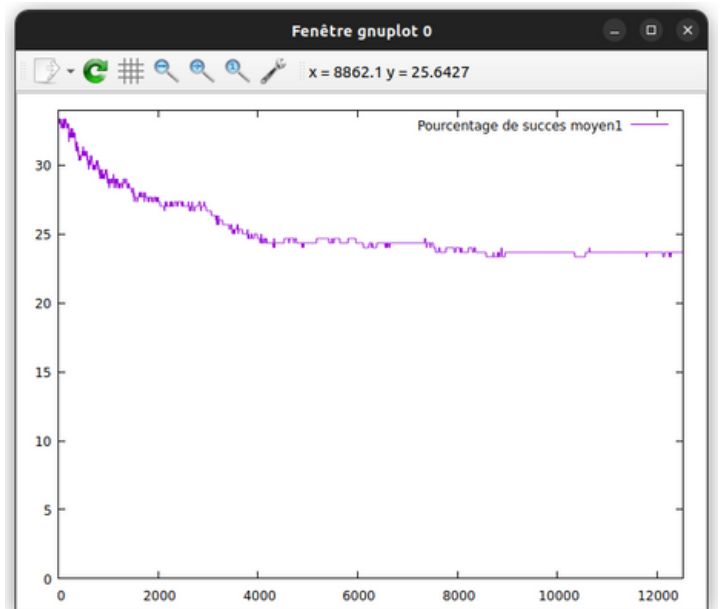
Nombre de places occupées



évolution du prix minimum



Profit



Succès moyen des parkings

Difficultés rencontrées

Lors de la réalisation du projet, nous avons dû consacrer un certain moment à l'apprentissage des threads et des mutex, essentiels aux conversations, mais également l'apprentissage de la librairie gnuplot pour la réalisation de graphique.

Nous avons du également faire face à certains problèmes de mémoire qui était problématique notamment afin de faire tourner la simulation pendant plusieurs heures afin de récolter un maximum de données et d'avoir des résultats plus cohérents. Mais également un problème de coordination au niveau de la méthode de codage qui a pu poser à certains moments quelques problèmes de compréhension ou d'erreurs lors de la mise en commun de certaines partie du projet.

Conclusions

En conclusion, ce projet nous a permis d'acquérir de nouvelles compétences et de mettre en pratique nos connaissances déjà acquises en informatique. Nous avons pu explorer différentes technologies et approches pour résoudre la problématique donnée. Bien que nous ayons rencontré quelques difficultés, et que certains points sont à revoir, tel que l'organisation et le fait de ne pas avoir mis en place une norme de codage, ce qui nous a coûté un certain temps. Nous avons surmonté ces difficultés grâce à notre travail d'équipe et à notre persévérance. Nous sommes fiers du résultat final et espérons que ce projet pourra correspondre aux attentes demandées.

Enfin, nous voulons exprimer notre gratitude envers les professeurs qui nous ont accompagnés tout au long de ce projet. Leur soutien, leur encadrement et leur disponibilité ont été précieux pour notre réussite. Grâce à leurs conseils avisés et leur expertise, nous avons pu surmonter les obstacles rencontrés et atteindre nos objectifs. Nous espérons que notre travail les satisfera et leur montrera que leur soutien a été bien utilisé.