République algérienne démocratique et populaire.

Ministère de L'enseignement Supérieure de la recherche scientifique.

Université 8 Mai 45 –Guelma-

Faculté des Mathématiques, d'informatique et des Sciences de la Matière

Département d'Informatique

Mémoire de Fin d'études Master

Filière : Informatique

Option : Sciences et Technologies de l'Information et de la Communication

Thème :

Optimisation du problème d'appariement dans un système de covoiturage

---

Présenté par : Zemouli Chayma

Membres du jury :

N	Nom et Prénom	Qualité
1	Dr. Karima Benhamza	Président
2	Dr. Hiba Abdelmoumène	Superviseur
3	Dr. Mohammed Chaoui	Examinateur

Juin 2023

Remerciement

Je souhaite exprimer ma gratitude envers mon encadrante, Dr. Abdelmoumène Hiba, pour avoir accepté de me guider dans la conception et l'élaboration de ce travail. Je tiens également à la remercier pour son dévouement et son soutien constant.

Mes remerciements vont également à toutes les personnes qui, de près ou de loin, ont contribué à l'élaboration de mon mémoire. Votre soutien, qu'il soit technique, logistique ou moral, a été précieux et a contribué à la réussite de mon travail.

Dédicace

Je souhaite exprimer ma profonde gratitude et mes sincères mots en dédiant ce modeste travail à mes chers parents. Leur dévouement et leur sacrifice pour notre réussite ont été remarquables, et ils ont éclairé notre chemin avec leurs conseils judicieux .

Nous aspirons à pouvoir un jour leur rendre ne serait-ce qu'une partie de ce qu'ils ont fait pour nous. Nous souhaitons sincèrement que Dieu leur accorde bonheur et longévité .

Je souhaite également dédier ce travail à mes frères " Bilel et Karim ", à ma jumelle " Hind " ainsi qu'à mes amis.
Je tiens à exprimer ma reconnaissance envers tous mes professeurs qui m'ont enseigné et envers toutes les personnes qui me sont chères.

Résumé

Un système de covoiturage dynamique permet aux utilisateurs de trouver des trajets en temps réel en mettant en relation les conducteurs et les passagers qui partagent un itinéraire similaire. Il optimise l'utilisation des véhicules en réduisant le nombre de voitures sur la route et en favorisant le partage des frais de transport.

Dans notre travail, nous nous sommes intéressées au problème d'appariement dynamique dans un système de covoiturage.

Nous avons proposé de solutionner le problème par l'apprentissage par renforcement. La solution proposée prend en considération des contraintes spatio-temporelles, des contraintes de capacité, des contraintes de temps d'attente et de détour. Le modèle que nous avons proposé vise à minimiser le temps d'attente des passager ainsi que le temps nécessaire pour faire le détour.

Nous avons vérifié notre modèle en utilisant des données réelles provenant de l'ensemble de données publiques des taxis de New York City ainsi qu'en utilisant un simulateur que nous avons développé afin d'évaluer les performances de l'approche.

Les résultats obtenus sont prometteurs et ont renforcé la validité de nos choix et démontré avec succès l'efficacité et la robustesse de notre approche en temps réel.

Mots clés

Covoiturage dynamique , Appariement dynamique , Apprentissage par renforcement, Contraintes spatio-temporelles, Détour, Temps d'attente.

Abstract

A dynamic ridesharing system enables users to find journeys in real time by connecting drivers and passengers who share a similar itinerary. It optimizes vehicle use by reducing the number of cars on the road and promoting the sharing of transportation costs.

In our work, we focus on the dynamic matching problem in a ridesharing system.

We proposed to solve the problem using reinforcement learning. The proposed solution takes into account spatio-temporal constraints, capacity constraints, waiting time constraints and detour constraints. Our proposed model aims to minimize passenger waiting time and detour time.

We verified our model using real data from New York City's public cab dataset, as well as using a simulator we developed to evaluate the performance of the approach.

The results obtained are promising, reinforcing the validity of our choices and successfully demonstrating the effectiveness and robustness of our real-time approach.

Keywords

Dynamic ridesharing, Dynamic matching, Reinforcement learning, Spatio-temporal constraints, Detour, Waiting time.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

Le covoiturage a gagné en popularité dans notre société moderne. Il représente un mode de transport collaboratif où des passagers partagent un véhicule pour effectuer un trajet commun. Le covoiturage offre de nombreux avantages à la fois sur le plan économique et environnemental. Il permet de réduire les coûts de transport, de diminuer les émissions de gaz à effet de serre et de soulager la congestion routière.

Le covoiturage peut être vu comme étant un problème d'optimisation. Il s'agit de trouver les meilleures combinaisons possibles pour maximiser l'efficacité des trajets, en minimisant le nombre de véhicules utilisés.

La résolution de ce problème exige l'utilisation des algorithmes sophistiqués qui prennent en compte des variables telles que les emplacements, les horaires, les préférences individuelles et les contraintes spécifiques.

Notre travail se concentre sur la gestion dynamique d'un ensemble d'offres de covoiturage fournies par des conducteurs et d'un ensemble de demandes émises par des passagers. Notre objectif princial est de développer un système capable d'optimiser les appariements dynamiques entre les conducteurs et les passagers, tout en minimisant le temps d'attente des passagers et le temps de détour des conducteurs et en respectant un ensemble de contraintes spécifiques.

Ainsi, les décisions opérationnelles liées au covoiturage sont de nature séquentielle et sont fortement influencées par des facteurs spatio-temporels. Cette caractéristique rend l'apprentissage par renforcement particulièrement pertinent, et nous envisageons d'explorer cette approche dans notre travail.

Pour atteindre notre objectif, nous mettons l'accent sur plusieurs facteurs clés. Tout d'abord, nous accordons une grande importance à la minimisation du temps d'attente des passagers. Pour ce faire, nous développons des mécanismes efficaces pour détecter

rapidement les offres de covoiturage disponibles et les demandes correspondantes, afin de les associer dans les meilleurs délais.

De plus, nous nous efforçons de réduire au maximum le temps de détour des conducteurs. Cela signifie que nous recherchons des correspondances qui ajoutent un minimum de temps supplémentaire au trajet initial du conducteur, tout en satisfaisant les demandes des passagers.

Pour parvenir à ces résultats, nous allons prendre en compte divers contraintes tels que les points de départ et d'arrivée des passagers, leurs itinéraires potentiels avec des détours éventuels, ainsi que les contraintes temporelles.

Notre approche vise donc à trouver un équilibre entre la réduction du temps d'attente des passagers et la minimisation du temps de détour des conducteurs, en prenant en compte toutes les contraintes et les caractéristiques spécifiques de chaque demande de covoiturage.

Ce mémoire est structuré de la manière suivante :

Chapitre 1 : Ce chapitre offre un aperçu du covoiturage, en mettant en évidence ses composants et contraintes. De plus, nous exposons les concepts fondamentaux de l'apprentissage par renforcement et ses différents composants. Nous procédons ensuite à une synthèse des travaux connexes.

Chapitre 2 : Ce chapitre est dédié à notre contribution. Nous commençons par expliquer les motivations et les justifications qui ont guidé nos choix. Nous introduisons ensuite la modélisation que nous avons proposée et nous détaillons les différentes phases de réalisation de notre travail.

Chapitre 3 : Ce chapitre se concentre sur les détails de l'implémentation de notre système. Nous présentons en détail les choix et les étapes de développement que nous avons suivis. Ensuite, nous discutons les résultats obtenus à travers nos expérimentations et les évaluations que nous avons réalisées pour évaluer l'efficacité de notre système

Chapitre 1 : Etat de l'art

1.1 Introduction

Le covoiturage est une solution de transport qui vise à optimiser l'utilisation des véhicules en permettant à plusieurs personnes de partager un même trajet. Au lieu de conduire seul dans sa voiture, les conducteurs offrent des places à d'autres passagers qui se rendent dans la même direction. Cette solution permet de réduire le nombre de voitures sur les routes, ce qui contribue à atténuer la congestion routière et à diminuer les émissions de gaz à effet de serre, c'est un mode de transport qui présente des caractéristiques particulières en termes de dynamisme, de contraintes et de considérations théoriques.

Dans ce chapitre, nous allons présenter, dans un premier temps, les aspects théoriques du covoiturage dynamique, ses composants ainsi que ses caractéristiques. Dans un second temps, nous allons étudier les travaux qui ont abordé la problématique d'appariement dynamique par l'apprentissage par renforcement (Sutton et Barto, 2018).

1.2 Covoiturage dynamique

Le covoiturage dynamique est une forme de covoiturage qui utilise des plateformes en ligne et des applications mobiles pour faciliter la mise en relation des conducteurs et des passagers en temps réel.

1.2.1 Définition

Le concept de covoiturage dynamique fait référence à un système automatisé qui facilite le partage de trajets ponctuels entre conducteurs et passagers, en tenant compte de l'heure de départ souhaitée, ce système est également appelé covoiturage en temps réel (Agatz et al., 2012).

Le covoiturage dynamique est un système de covoiturage en temps réel qui permet à

chacun de trouver un conducteur ou des passagers pour partager les frais de transport. Contrairement au covoiturage statique, qui consiste à organiser à l'avance un trajet via une plateforme internet, le covoiturage dynamique fournit une offre en quasi temps réel.

Un système de covoiturage se distingue par la présence simultanée de multiples demandes de trajets, chacune ayant ses propres points de départ et d'arrivée, ainsi qu'un nombre défini de véhicules disponibles. L'objectif primordial est d'optimiser les itinéraires afin de répondre de manière optimale à ces demandes. Cette optimisation implique la prise en compte de plusieurs paramètres, tels que le temps de trajet, la distance à parcourir, la disponibilité des véhicules et les préférences des passagers. En réussissant à concilier ces différents éléments, on peut atteindre une efficacité maximale du système de covoiturage, offrant ainsi une meilleure expérience aux utilisateurs et contribuant à réduire la congestion routière et les émissions de CO2. La clé du succès d'un système de covoiturage réside dans sa capacité à orchestrer de manière intelligente et efficiente les trajets et les ressources disponibles pour répondre aux besoins des utilisateurs, tout en minimisant les coûts et en maximisant les avantages pour toutes les parties prenantes (Martins et al., 2021,Ting et al., 2021).

1.2.2 Composants d'un système de covoiturage dynamique

Dans le cadre du covoiturage dynamique, la demande et l'offre de trajets dépendent des besoins des passagers et de la disponibilité des conducteurs. L'objectif du système est de faire correspondre ces deux facteurs de la manière la plus efficace possible, tout en offrant une expérience de haute qualité aux passagers et aux conducteurs.

Un système de covoiturage est constitué d'un certain nombre de composants nécessaires à sa mise en œuvre. Nous allons présenter, dans ce qui suit, ces différents composants.

- ***Passager :*** utilise généralement une plateforme pour exprimer une demande de partage d'un trajet. (Furuhata et al., 2013).
- ***Demande :*** Ce sont les requêtes postées par les passagers souhaitant partager un trajet. Ces demandes comprennent le point de départ, le point d'arrivée, le temps de départ et le temps d'arrivée.
- ***Conducteur :*** C'est un élément essentiel des systèmes de covoiturage dynamique.

Ils fournissent des services de transport aux passagers et sont chargés d'assurer une expérience de transport satisfaisante et confortable.

- ***Offre :*** elle est postée par un conducteur exprimant sa volonté de partager son trajet dans sa propre voiture avec des passagers. Cette offre doit être accompagnée d'une liste de paramètres : point de départ, point d'arrivée , nombre de sièges disponibles et les temps de départ et d'arrivée.
- ***Contraintes :*** L'appariement d'une demande avec une offre dans un système de covoiturage est soumis à un ensemble de contraintes qui doivent être respectées. Ces contraintes peuvent être liées à plusieurs aspects, tels que l'emplacement (par exemple, un trajet identique, un trajet du passager inclus dans le trajet du conducteur, ou des trajets partiellement identiques), le temps (les heures de départ et d'arrivée doivent correspondre), le respect du nombre de sièges disponibles, le temps d'attente des passagers et des conducteurs, le temps de détour, la distance de détour, etc (Agatz et al., 2011).

  La satisfaction de ces contraintes est essentielle pour garantir un appariement approprié entre les demandes et les offres, assurant ainsi une expérience de covoiturage efficace et satisfaisante pour tous les utilisateurs. En respectant ces contraintes, le système peut optimiser les trajets, maximiser l'utilisation des véhicules et offrir des solutions de covoiturage pratiques et adaptées aux besoins spécifiques de chaque passager et conducteur.

La figure 1.1 illustre les composants d'un système de covoiturage dynamique et la relation entre eux.

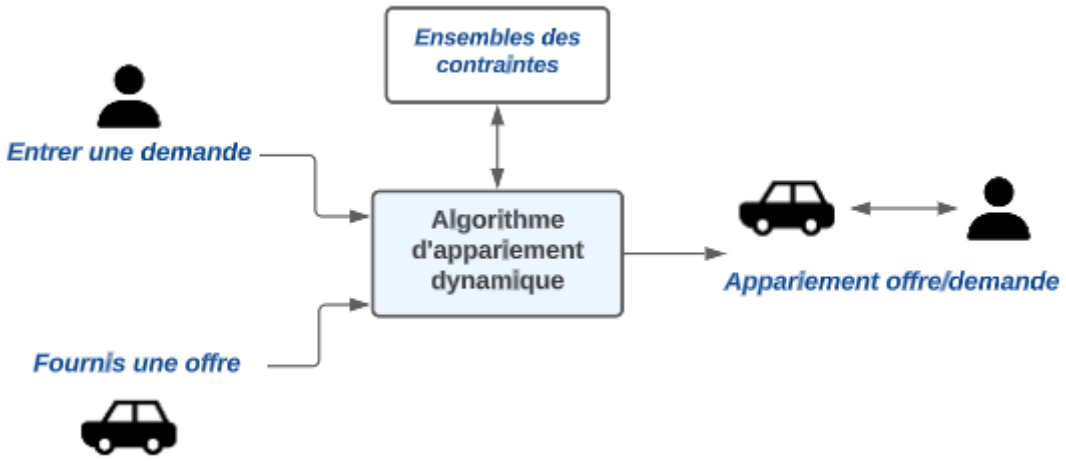

FIGURE 1.1 – Composant d'un système de covoiturage.

1.2.3 Variantes du covoiturage dynamique

Les conducteurs qui proposent des trajets peuvent choisir de prendre un seul passager ou d'accepter plusieurs passagers à bord de leur véhicule. De même, les passagers qui font une demande de trajet peuvent être ouverts à voyager avec un seul conducteur ou peuvent être prêts à partager leur trajet avec plusieurs conducteurs. Cela signifie qu'une demande peut être affectée à plusieurs véhicules afin d'atteindre sa destination finale, chaque conducteur intervenant uniquement sur une partie du trajet du passager. Ce type de covoiturage est connu sous le nom de "sauts multiples", où le trajet d'un passager peut être composé de plusieurs segments effectués avec différents conducteurs. Cette approche permet une plus grande flexibilité et une optimisation accrue des itinéraires, offrant ainsi une solution de covoiturage plus efficace et adaptée aux besoins spécifiques des passagers et des conducteurs.

De ces différentes configurations, découlent quatre variantes du covoiturage, que nous allons présenter ci-dessous (Plate, 2019).

1.2.3.1 Un conducteur et un passager (1C - 1P)

Il correspond à un appariement dans lequel chaque conducteur du système ne peut prendre qu'un seul passager, et chaque passager souhaite se rendre avec un seul conducteur à sa destination (Plate, 2019 ; Herbawi, 2013) .

1.2.3.2 Un conducteur et plusieurs passagers (1C - NP)

Le covoiturage dynamique avec un seul conducteur et plusieurs passagers est un type de transport dans lequel un seul conducteur propose un trajet à plusieurs passagers qui vont dans la même direction (Plate, 2019). Les décisions concernant le trajet sont prises en temps réel, et les passagers partagent généralement le coût d'itinéraire. Selon (Jindal et al., 2018), un véhicule individuel peut être partager entre un groupe de passagers en fonction de plusieurs critères tels que la capacité et le délai de détour (Z. T. Qin et al., 2022).

1.2.3.3 Plusieurs conducteur et un passager (NC - 1P)

Dans cette variante, un passager peut se rendre à sa destination en se déplaçant avec différents conducteurs. Chaque conducteur le rapprochera un peu plus de sa destination finale. Il est parfois appelé covoiturage à sauts multiples, car le passager effectue plusieurs "étapes" ou "transferts" ou encore des "sauts" pendant son trajet (Singh et al., 2021).

1.2.3.4 Plusieurs conducteurs et plusieurs passagers (NC - NP)

Cette variante de covoiturage implique des scénarios où un conducteur peut être associé à plusieurs passagers et où un passager peut être associé à plusieurs conducteurs. Cela entraîne une combinaison des défis liés à l'appariement d'un seul conducteur avec plusieurs passagers et à l'appariement de plusieurs conducteurs avec un seul passager. Dans cette configuration, la gestion des itinéraires et des contraintes devient plus complexe, car il faut tenir compte des préférences et des contraintes de chaque partie impliquée. Cependant, en permettant ces combinaisons, on peut optimiser l'utilisation des véhicules et offrir davantage d'options aux passagers, améliorant ainsi l'efficacité globale du système de covoiturage (Herbawi, 2013).

1.2.4 Contraintes sur le covoiturage dynamique

L'objectif d'un système de covoiturage dynamique est de répondre à un ensemble de demandes tout en respectant un ensemble de contraintes. Les caractéristiques du problème, telles que le temps, le lieu, le coût, l'itinéraire et d'autres facteurs, peuvent être utilisées pour générer ces contraintes.

Nous allons présenter, dans ce qui suit, quelques contraintes liées au covoiturage dynamique. Il faut noter que cette lise n'est pas exhaustive.

- **Contrainte de capacité**

  Une contrainte de capacité dans les systèmes de covoiturage limite le nombre de passagers utilisant le même véhicule en même temps au nombre de sièges vacants dans ce véhicule (Santos et Xavier, 2015).

- **Contrainte de temps**

  Dans les systèmes de covoiturage, les passagers soumettent généralement leurs demandes avec une fenêtre d'horaire comprenant l'heure de départ prévue et l'heure d'arrivée souhaitée. De leur côté les conducteurs aussi peuvent spécifier leur fenêtre temporelle (Agatz et al., 2011).

- **Contrainte de détour**

  Lorsqu'un conducteur propose un covoiturage, il établit un itinéraire initial avec des points de départ et d'arrivée spécifiques. Cependant, il peut arriver que, dans certaines situations, que pour récupérer ou déposer un passager, le conducteur aura besoin de faire un détour. Cette contrainte a, généralement, pour effet d'augmenter le temps et de rallonger la distance. La figure 1.2 présente deux scénarios de covoiturage avec et sans détour.

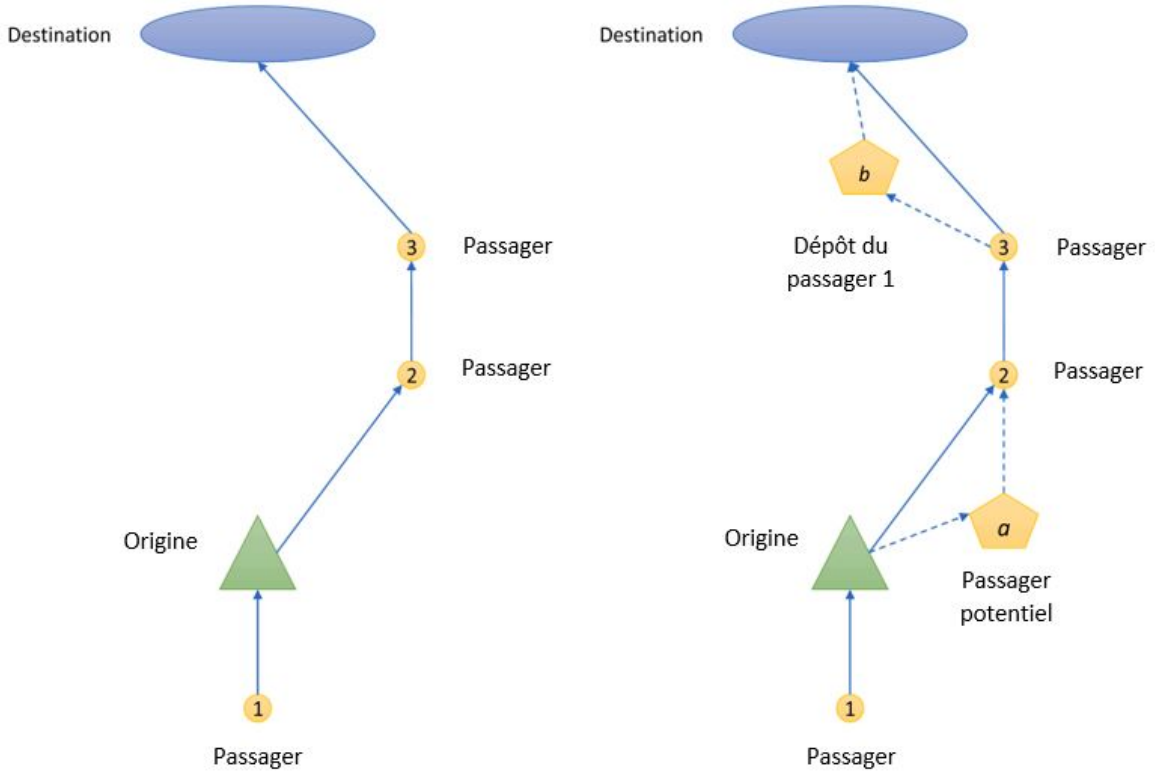

FIGURE 1.2 – Représentation de deux scénarios de covoiturage : sans détours et avec détours (Martins et al., 2021).

- **Contrainte de temps d'attente**

  Les conducteurs et les passagers dans le système de covoiturage peuvent définir un délai d'attente maximal, ce qui représente leur flexibilité en termes de temps. Les passagers peuvent attendre jusqu'à ce délai avant le départ, tandis que les conducteurs peuvent effectuer des détours pour prendre certains passagers dans ce laps de temps (Mourad et al., 2019).

- **Contrainte de coût**

  Un passager peut indiquer le coût maximum qu'il est prêt à payer pour un trajet partagé. Par conséquent, il doit être apparié avec des trajets partagés qui ne dépassent pas ce montant maximum spécifié (Mourad et al., 2019).

1.3 Appariement dynamique

1.3.1 Critères à optimiser

La majorité des objectifs que les problèmes de covoiturage visent à atteindre peuvent

être divisés en trois catégories : les objectifs liés à l'espace, liés au temps et liés à la qualité du système.

De plus, un système de covoiturage peut optimiser un seul objectif (***mono objectif***) comme il peut viser à atteindre plusieurs objectifs (***multi-objectifs***) (Mourad et al., 2019).

1.3.1.1 Objectifs liés à l'espace

- **Minimiser la distance parcourue**

  Il s'agit de miniser la distance totale parcourue par tous les participants pour se rendre à leur destination. D'un point de vue sociétal, cet objectif est significatif puisqu'il réduit le traffic et la pollution.

- **Minimiser la distance de détour**

  Afin de répondre à certaines requêtes, les conducteurs se trouvent face un détour à faire. Il est important que la distance parcourue lors du détour soit optimisée afin de pouvoir optimiser la distance totale parcourue (Herbawi, 2013).

1.3.1.2 Objectifs liés au temps

- **Minimiser le temps d'attente**

  Ce paramètre, qui correspond au temps d'attente d'un passager avant qu'un conducteur vienne le chercher, est le fondement de nombreux systèmes de covoiturage. L'optimisation de ce critère a pour but d'augmenter la qualité du service en satisfaisant les passagers desservis et de contribuer à minimiser le temps total des trajets.

- **Minimiser le temps de réponse**

  Le temps qu'il faut pour appairer les conducteurs et les passagers est ce qui est considérer comme temps de réponse. L'objectif est d'obtenir une réponse acceptable dans les délais les plus courts possibles afin de minimiser le temps d'attente pour le service. Cela revêt une importance particulière pour les services en ligne ou en temps réel, où les demandes peuvent être déposer peu de temps avant l'heure de départ souhaitée (Plate, 2019).

- **Minimiser le temps du trajet**

  Il reflète le temps passé dans la voiture pour se rendre du point de départ au point

d'arrivée. Ce critère joue un rôle essentiel en termes de commodité pour les participants. (Agatz et al., 2012)

- **Minimser le temps de détour**

  Le temps de détour représente la durée supplémentaire que le conducteur et les passagers qui sont à bord de sa voiture doivent prendre en compte lors de la récupération et de la dépose d'un passager (Z. T. Qin et al., 2022).

1.3.1.3 Objectifs liés à la qualité du système

- **Maximiser le nombre de participants**

  Cet objectif permet de maximiser la satisfaction des conducteurs et des passagers dans le système de covoiturage (Agatz et al., 2012).

- **Maximiser le nombre de requête desservies**

  Ce critère vise à obtenir le meilleur rendement possible en termes de satisfaction des demandes de trajet. Cela signifie que l'objectif est de répondre au plus grand nombre de requêtes de covoiturage, en veillant à ce que chaque passager soit pris en charge et déposé à sa destination de manière efficace. (Plate, 2019).

- **Minimser le taux d'annulation de requête**

  Il s'agit de l'annulation après l'appariement, généralement causée par l'attente de la récupération (Z. T. Qin et al., 2021).

Les contraintes et critères d'optimisation mentionnés précédemment, ainsi que la problématique théorique du covoiturage dynamique, ouvrent la voie à différentes techniques d'optimisation pour résoudre ce problème. Parmi ces techniques, l'apprentissage par renforcement (AR) a récemment connu un grand succès dans la communauté du covoiturage dynamique. Cette popularité peut s'expliquer par la capacité de l'AR à bien représenter les aspects décisionnels, séquentiels et temporels de ce domaine.

Nous nous intéressons dans notre travail à l'application de l'AR dans ce cadre. Cependant avant d'entamer notre contribution, nous allons présenter les concepts clés de l'AR ainsi qu'une synthèse des travaux connexes.

1.3.2 Apprentissage par renforcement

L'apprentissage par renforcement (AR) est une approche d'apprentissage automatique

qui vise à former un agent à prendre des actions optimales en interagissant avec son environnement et en maximisant une récompense cumulative. Cette méthode d'optimisation est particulièrement adaptée pour résoudre des problèmes de prise de décision séquentielle, où les actions prises par l'agent ont un impact sur les états futurs de l'environnement et les récompenses qu'il reçoit(Z. T. Qin et al., 2022 ; Sutton et Barto, 2018).

Initialement, l'agent sélectionne une action de manière aléatoire dans l'espace des actions, étant donné un état, car il n'a pas connaissance de quelle action prendre dans un état donné. Cela indique que l'agent explore son environnement en effectuant des actions aléatoires .À mesure que le processus se réalise, l'agent gagne en confiance quant à ses actions prédites et commence à exploiter ses connaissances en choisissant une action ayant la plus grande valeur estimée et offrant la récompense la plus élevée (Jindal et al., 2018 ; Wiering et Van Otterlo, 2012).

Le framework du processus de décision de Markov (PDM) permet de formaliser les éléments du problème d'apprentissage par renforcement (AR). Les PDMs sont composés d'un ensemble d'états représentant les différentes situations dans lesquelles l'agent peut se trouver, d'un ensemble d'actions représentant les choix disponibles à l'agent, de transitions qui décrivent comment l'état de l'environnement évolue en fonction des actions de l'agent, et d'une fonction de récompense qui évalue la qualité des actions entreprises par l'agent (Wiering et Van Otterlo, 2012) .

1.3.2.1 Composants de l'apprentissage par renforcement

L'AR est basé sur le cadre du processus décisionnel de Markov (MDP : Markov Decision Process).

- ***Agent :*** unité qui prend la décision.
- ***Etat :*** l'état actuel de l'agent correspond à l'ensemble des variables qui décrivent l'environnement. Dans le cas du covoiturage, l'état peut être l'emplacement du véhicule, par exemple il s'agit des coordonnées géographiques du conducteur et de l'heure du jour (Wang et al., 2018).
- ***Action :*** C'est la décision prise par l'agent, et elle varie en fonction de la manière dont l'agent est modélisé, qu'il soit au niveau du véhicule ou du système dans son ensemble (Z. T. Qin et al., 2022).

- ***Récompense :*** L'agent est immédiatement récompensé par l'environnement pour avoir réalisé l'activité . Cela dépend de la fonction objectif qui peut être le profit, par exemple (Wang et al., 2018).
- ***Transition :*** La transition d'état, représente la probabilité de passer de l'état actuel à l'état suivant, et est déterminée en fonction de l'état actuel et de l'action choisie (Knox et Stone, 2012).
- ***Episode*** représente une séquence d'étapes nécessaires pour atteindre l'état cible à partir d'un état initial (Dangeti, 2017).
- ***Politique :*** La politique détermine quelle action choisir et exécuter pour chaque état de l'environnement (Knox et Stone, 2012).
- ***Fonction de valeur :*** Elle correspond à une estimation des récompenses futures pour chaque état (Knox et Stone, 2012).

Les méthodes d'apprentissage par renforcement peuvent être regroupées en deux grandes catégories : les méthodes basées sur un modèle *" model-based"* et les méthodes sans modèle *" model-free"* (Jindal et al., 2018). Les méthodes sans modèle, ou "model-free", ne nécessitent pas la disponibilité de modèles de transition et de récompense connus à l'avance, c'est-à-dire un modèle complet du processus de décision de Markov (MDP). Dans les méthodes RL basées sur un modèle, le modèle de transition est d'abord appris à partir des interactions avec l'environnement, puis il est utilisé pour dériver une politique optimale (Jindal et al., 2018).

1.4 Synthèse des travaux

(Wang et al., 2018) utilisent la modélisation du problème de répartition des trajets sous la forme d'un processus décisionnel de Markov (MDP) et offrent des solutions d'apprentissage qui reposent sur des réseaux Q profonds, accompagnés d'une recherche d'action, afin d'améliorer l'optimisation de la politique d'appariement des conducteurs sur les plateformes de covoiturage.

Un passager soumet une demande de transport avec des informations sur l'origine et la destination, que le système convertit en coordonnées GPS. Lorsqu'une demande est soumise, le système de répartition la saisit et s'efforce de l'attribuer à un conducteur

disponible tout en respectant une politique de répartition spécifique (Z. Qin et al., 2020).

(Ke et al., 2020) ont établi un cadre qui combine l'apprentissage par renforcement profond et l'optimisation combinatoire multi-agents, dans lequel le moment où chaque demande de passager entre pour la correspondance est déterminé dynamiquement à l'aide de techniques d'apprentissage par renforcement multi-agents, tandis que l'optimisation combinatoire permet d'obtenir une correspondance parfaite bidirectionnelle entre les passagers en attente et les conducteurs inactifs.

(Xu et al., 2018) ont modélisé la répartition des commandes comme un problème de décision séquentielle à grande échelle en utilisant le processus de décision de Marcov, où la décision d'attribuer une commande à un conducteur est déterminée par un algorithme centralisé de manière coordonnée.

L'étude de (Kullman et al., 2022) aborde la problématique d'un opérateur qui gère une flotte de véhicules électriques utilisés pour un service de covoiturage dans l'objectif de maximiser ses bénéfices en assignant les véhicules aux demandes au fur et à mesure de leur apparition, tout en planifiant leur recharge et leur repositionnement en prévision des demandes futures.

(Tang et al., 2021) considèrent les problèmes liés à la l'appariement et au repositionnement, deux opérations clés des plateformes de covoiturage. Ils suggèrent l'utilisation d'une fonction de valeur centralisée comme fondement pour l'apprentissage et l'optimisation afin de saisir les interactions entre ces deux tâches. Dans cette optique, ils proposent une approche novatrice basée sur un ensemble de valeurs, permettant ainsi un apprentissage en ligne rapide et un entraînement hors ligne à grande échelle.

(Li et al., 2019) traitent la problématique de l'appariement des demandes en utilisant l'apprentissage par renforcement multi-agents (MARL), qui peut saisir la dynamique stochastique de l'offre et de la demande dans les scénarios de covoiturage à grande échelle.

(Jin et al., 2019) ont introduit CoRide, une solution hiérarchique basée sur l'apprentissage par renforcement multi-agents, qui vise à fusionner la répartition des demandes et la gestion des véhicules dans les plateformes de covoiturage à différentes échelles.

(Zhou et al., 2019) présentent une approche décentralisée pour traiter les demandes

, en utilisant l'apprentissage par renforcement multi-agent, dans le but de résoudre les problèmes liés à la gestion des commandes à grande échelle.

développement de (Al-Abbasi et al., 2019), appelé DeepPool, est un modèle distribué qui utilise des techniques de réseau neuronal profond Q (DQN) pour apprendre des politiques d'envoi optimales en interagissant avec l'environnement. Ce modèle *"model-free"* intègre également efficacement les statistiques de la demande de déplacements et des modèles d'apprentissage profond pour améliorer la gestion de l'envoi des véhicules et optimiser les services de covoiturage.

Le travail de (Jindal et al., 2018) consiste en deux aspects principaux. Tout d'abord, ils créent un modèle de réseau neuronal profond appelé ST-NN (Spatio-Temporal Neural Network) qui permet de prédire la durée des trajets en taxi en se basant sur les données brutes provenant des trajets GPS. Ensuite, ils développent un environnement de simulation de covoiturage utilisant l'ensemble de données des trajets en taxi à New York, où ils utilisent les prédictions fournies par le ST-NN comme entrée pour l'entraînement par renforcement.

Nous allons présenter, dans ce qui suit, un tableau récapitulatif des travaux mentionnés précédemment. Nous allons classer ces travaux en fonction des caractéristiques du problème, les critères à optimiser et les composants du MDP.

	Articles	Agent	Etat	Action	Récompense	Transition	contraintes
Mono obejctifs	(Wang et al., 2018)	Conducteur	- paire de coordonné de la localisation actuel. - le temps	accepter ou refuser la demande .	le montant total de voyage .	Le prochain état correspond à l'emplacement, à l'heure du dépôt.	**Variante :** 1C - 1P **Contrainte :** de temps **Objectifs :** -Maximiser le monant total.
	(Xu et al., 2018)	Conducteur	- temp - localisation	- service d'une demande - Idle (il n'est apparié à aucune passager dans une période de temps définie).	Le prix d'une commande	dépendre d'une heure d'arrivée et d'un prix prévu	**Variante :** 1C - 1P **contraintes :** temps. **Objectif :** Minimiser la distance de prise en charge.
	Ke et al., 2020	demande des passagers	- Localisation actuelle. - le temps d'attente . -la distance .	a = {1 , 0} .	le temps d'appariement	À chaque intervalle de temps, l'état de chaque agent est actualisé en fonction des interactions entre les actions des agents et l'environnement.	**Variante :** 1C - 1P **Contraintes :** temps . **Objectifs :** -Minimiser le temps d'appariement .

Mono obejctifs	(Kullman et al., 2022)	un opérateur central	Tuple de : le temps current - les coordonnées d'origine et destination de demande - coordonnées de position actuel , état du véchile , l'origine et destination du véchile et	Servir, Repositionnement ,NOOP (rejeter la demande).	les revenus générés par le service d'une nouvelle demande	Le changement d'état dépend de l'action choisie et de l'événement qui déclenche l'épisode suivant.	**Variante :** NC - NP **Contrainte :** temps , détour **Objectif :** - maximiser le profit .
	(Tang et al., 2021)	Conducteur	Localisation current et le temps	une affectation de voyage, soit un repositionnement	montant du voyage	état actuel et action (option)	**Variante :** 1C - 1P **Contrainte :** temps **Objectif :** - Maximiser le profit total de tous les conducteurs.
	(Li et al., 2019)	Conducteur	La localisation , horaire et drapeau de voyage(on-trip flag)	l'origine et la destination de demande	Prix de demande	Au pas de temps , chaque agent effectue une action, ce qui forme un ensemble de paires d'ordres de conduite conjointe.	**Variante :** 1C - 1P **Contrainte :** temps et détour **Objectif :** - maximiser le revenu total.

Multi obejctifs	(Jindal et al., 2018)	Conducteur	- Localisation actuel (latitude ,longitude) - Temps	deux type d'action : - prise en charge un passager. -prise en charge plusieurs passager .	La distance de voyage .	Lorsqu'un véhicule accomplit une action qui lui a été affectée, l'état du véhicule est mis à jour .	**Variante :** 1C - NP **Contrainte :** temps , détour **Objectif :** - Maximiser l'efficacité des transports . - Minimiser la congestion du trafic
	(Zhou et al., 2019)	Conducteur	l'indice de grille, le nombre de véhicules inactifs,le nombre de commandes valides et la répartition des destinations des commandes.	prise en charge le passager ou non	Le prix d'une commande	correspond 'a la terminsaion d'une demande.	**Variante :** 1C - 1P **Contrainte :** temps **Objectif :** - Maximiser le revenu cumulé des conducteurs. -taux de réponse aux demandes.
	(Z. Qin et al., 2020)	Conducteur	- Localisation : un système de grille hexagonale . - Le temps : en minutes	affectation de trajet ou inoccupé	Le prix d'une demande	Etat suivant (Destination) , récompense et destination	**Variante :** 1C - 1P **Contrainte :** de temps **Objectifs :** - Maximiser le revenu total. - Minimiser la distance de prise en charge

Multi obejctifs	Al-Abbasi et al., 2019	Conducteur	- Localisation actuelle. - Nombre de siège disponible . - Le temps de prise en charge du passager . - La destination de chasue passager.	a = {1 , 0} . a=1 , si le conducteur décide de servir un nouvel passager ,sinon a=0.	Temps de détour . temps de trajet . le nombre de passagers desservis .	Etat actuel . action .	**Variante :** 1C - NP **Contraintes :** temps , nombre de siége , detour . **Objectifs :** -Minimiser l'inadéquation entre l'offre et la demande. - Minimiser le temps d'attente des passager. - Minimiser le temps de détour . - Maximiser le nombre de participant.
	Jin et al., 2019	cellule de la région	le nombre de véhicules, le nombre de commandes, l'entropie et la distribution des caractéristiques des commandes (par exemple, le prix, la durée).	défini comme le vecteur de poids pour les caractéristiques de hiérarchisation.	le revenu actualisé	À chaque instant, l'ensemble du système multi-agents génère une action conjointe pour chaque manager et chaque travailleur, ce qui entraîne une transition dans l'environnement.	**Variante :** 1C - 1P **Contraintes :** temps . **Objectifs :** Maximise le revenu cumulé du conducteur et le taux de la réponse .

TABLE 1.1 – Classification des travaux connexes.

1.5 Conclusion

Nous avons présenté dans ce chapitre les différents types de covoiturages, les concepts liés au covoiturage dynamique ainsi que ses différents composants. Nous avons spécifié par la suite, un ensemble de contraintes, un ensemble de critères et les caractéristiques théoriques de ce système. En dernier, nous avons passé en revu quelques travaux de ce domaine qui ont exploré l'apprentissage par renforcement et nous avons donné une classification de ces travaux. Dans le chapitre suivant, nous allons exposé notre contribution pour résoudre le problème d'appariement dynamique dans un système de covoiturage.

Chapitre 2 : Contribution

2.1 Introduction

Dans ce chapitre, nous abordons notre contribution majeure, qui se concentre sur la modélisation et la résolution du problème d'appariement dynamique en utilisant l'apprentissage par renforcement. Nous mettons en évidence l'importance des aspects dynamiques, spatio-temporels et décisionnels inhérents à ce problème, et nous explorons comment l'AR peut être une approche efficace pour résoudre cette problématique complexe.

Le problème que nous considérons prend en compte plusieurs contraintes, notamment les contraintes temporelles telles que les fenêtres temporelles, les temps d'attente des passagers et les temps de détour des conducteurs. De plus, les contraintes spatiales et les contraintes de capacité sont également prises en compte. Ces différentes contraintes ajoutent une dimension supplémentaire de complexité au problème d'appariement dynamique.

Nous adoptons une approche d'optimisation multi-critères pour résoudre ce problème. Notre objectif est de minimiser à la fois le temps d'attente des passagers, afin de garantir une expérience de voyage agréable, et le temps de détour des conducteurs, afin d'optimiser l'efficacité du système. En prenant en compte ces critères multiples, nous cherchons à trouver un équilibre entre la satisfaction des passagers et la rationalité des trajets des conducteurs.

2.2 Problématique et objectifs de notre approche

Problématique

Le covoiturage, un concept existant depuis longtemps, a gagné en popularité ces dernières années grâce à l'émergence de plateformes technologiques telles qu'Uber, DiDi et Lyft.

Ces applications permettent aux utilisateurs de trouver facilement des trajets partagés avec d'autres personnes, ce qui favorise la mise en commun des déplacements. En plus de réduire le nombre de voitures sur les routes, le covoiturage contribue également à la diminution des embouteillages et de la pollution atmosphérique.

En pratique, le covoiturage consiste à regrouper des personnes se rendant dans une même direction afin qu'elles partagent un véhicule commun. La mise en relation des conducteurs et des passagers est appelée appariement dynamique.

Le problème d'appariement dynamique dans le covoiturage implique de trouver le meilleur conducteur disponible pour chaque passager en temps réel, tout en respectant des contraintes spécifiques. Ces contraintes peuvent être des contraintes temporelles, des contraintes de capacité et des contraintes spatiales. Pour résoudre ce problème complexe, il est souvent nécessaire d'utiliser des algorithmes d'optimisation, qui cherchent à maximiser ou minimiser une fonction objective tout en respectant l'ensemble des contraintes correspondantes.

Nous nous intéressons dans ce travail à résoudre le problème d'appariement dynamique sous contrainte par l'Apprentissage par Renforcement (AR). Cette méthode d'apprentissage est basée sur l'optimisation des décisions séquentielles d'un agent interagissant avec un environnement donné. En effet, le covoiturage dynamique nécessite des prises de décisions opérationnelles séquentielles qui dépendent à la fois de l'espace et du temps, ce qui en fait une application idéale pour l'apprentissage par renforcement.

Objectifs

L'objectif principal de notre travail consiste à trouver une solution au problème de gestion dynamique d'un ensemble d'offres de covoiturage proposées par des conducteurs et d'un ensemble de demandes postées par des passagers. Nous optons à mettre en place un système qui doit optimiser les solutions d'appariement dynamique d'un conducteur avec plusieurs passager (1C→NP), ayant comme objectifs : **la minimisation du temps d'attente des passagers et la minimisation du temps de détour des conducteurs**.

Cette optimisation doit respecter les fenêtres temporelles des conducteurs et des passagers, ainsi que leurs points de départ et d'arrivée, le nombre de siège disponible dans le véhicule,

le temps d'attente maximal des passagers et le temps de détour maximal des conducteurs.

2.3 Description du problème

Nous considérons le problème de mise en correspondance, en temps réel, de couples (offre, demande) de partage de trajet. Ce problème est caractérisé par un ensemble d'offres, un ensemble de demandes et un ensemble de contraintes que nous détaillons dans ce qui suit.

2.3.1 Offres

Nous notons $C_n(t) = \{C_1, C_2, ..., C_k, ..., C_n\}$, un ensemble d'offres de partage de trajet de $\boldsymbol{n}$ véhicules déposées par les conducteurs au temps t. Une offre d'un conducteur k, C_k est définie par : $C_k(t) = (Co_k, Cd_k, Dp_k, A_k, S_k, MD_k)$, où :

- Co_k : est le point de départ du conducteur k .
- Cd_k : est le point d'arrivée du conducteur k.
- Dp_k : spécifie le temps de départ au plus tard du conducteur k.
- A_k : indique le temps d'arrivée au plus tard du conducteur k.
- S_k : indique le nombe de sièges disponibles dans le véhicule du conduteur k.
- MD_k : le temps de détour maximal du conducteur k

2.3.2 Demandes

$P_{nd}(t) = \{P_1, P_2, \ldots, P_i, \ldots, P_{nd}\}$ est l'ensemble de $\boldsymbol{nd}$ demandes déposées par des passagers au temps t avec $nd \leq n$. Une demande d'un passager i est définie par :
$P_i(t) = (Po_i, Pd_i, A_i, Dp_i, MW_i)$:

- Po_i : est le point de départ du passager i .
- Pd_i : est le point d'arrivée du passager i.
- A_i : spécifie le temps d'arrivée au plus tard du passager i.
- Dp_i : indique le temps de départ au plus tard du passager i.
- MW_i : le temps d'attente maximal du passager

2.3.3 Contraintes

Les contraintes que nous considérons pour notre problème sont :

2.3.3.1 Contrainte spatio-temporelle

Notre objectif consiste à mettre en place un système de covoiturage dynamique qui respecte les itinéraires et les horaires des conducteurs et des passagers (point de départ et point d'arrivée ; temps de départ et temps d'arrivée).

2.3.3.2 Nombre de sièges disponibles

Chaque conducteur peut avoir plus d'un passager à servir en fonction du nombre de sièges disponibles. Le nombre de passagers appariés doit être inférieur ou égale au nombre de sièges vides dans le véhicule.

2.3.3.3 Détour

Dans le covoiturage dynamique, la contrainte de détour est un élément essentiel à prendre en compte car elle peut affecter la durée du trajet ainsi que la planification des conducteurs et des passagers. Elle se réfère à la distance supplémentaire que le conducteur doit parcourir par rapport à son itinéraire habituel pour aller chercher ou déposer un ou plusieurs passagers.

Exemple illustratif

Nous présentons à travers cet exemple un scénario de covoiturage dynamique relatif aux caractéristiques du problème que nous considérons (Figure 2.1). Nous supposons qu'un conducteur i propose un trajet du point ($Co_i = 1$) au point d'arrivée ($Cd_i = 5$). Le passager P_0 dépose sa demande au point 1 et son trajet est inclus dans le trajet du conducteur. Les passagers P_1 et P_2 déposent leurs demandes au point 2 et le passager P_3 postule au point 6. Certains trajets des passagers sont inclus dans le trajet initial du conducteur et d'autres nécessitent un détour. Les différentes contraintes relatives à l'offre et aux demandes sont illustrées sur la figure.

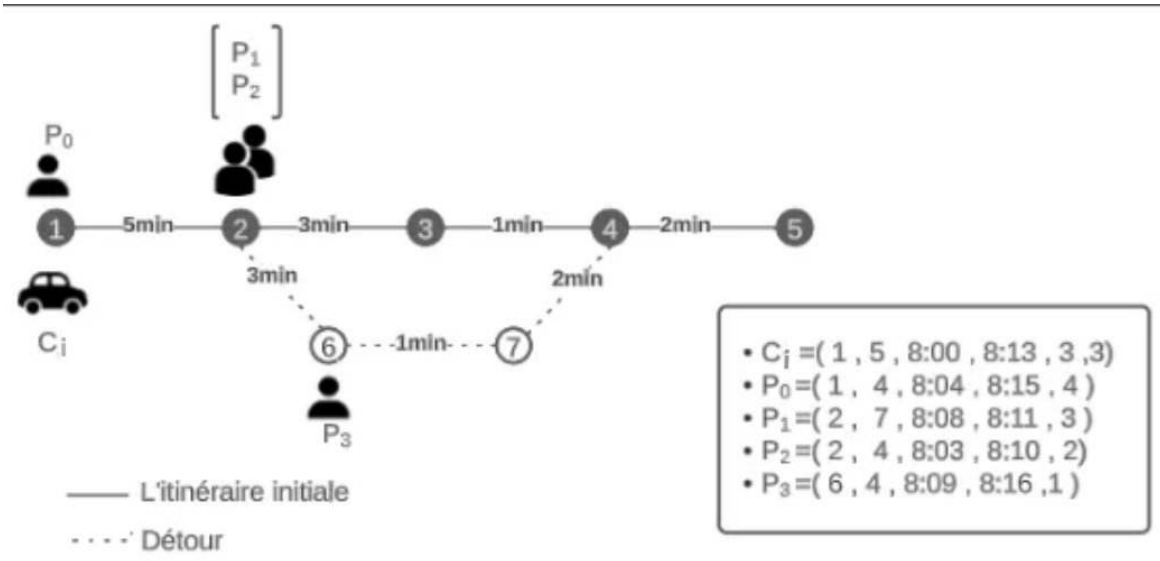

FIGURE 2.1 – Exemple illustratif d'un problème d'appariement.

2.4 Modélisation proposée

Dans le cadre d'un système de covoiturage en temps réel, nous abordons la question de la correspondance dynamique limitée dans l'espace et dans le temps entre les offres d'un conducteur et les demandes d'un passager. Nous proposons d'utiliser l'apprentissage par renforcement pour résoudre ce problème. En conséquence, nous devons construire les différentes composantes d'un MDP afin de décrire les caractéristiques du problème, ses contraintes, et les objectifs à optimiser.

Avant de présenter la modélisation proposée, il faut, tout d'abord, noter que ce travail est une continuité d'un travail de Master de l'année passée (TOUATI, 2022) . Lors de ce travail une première modélisation du problème d'appariement dynamique avec l'AR a été proposée. Cependant, le problème considéré ne prend en considération que les contraintes spatio-temporelles avec comme objectif la minimisation du temps d'attente des passagers. Des hypothèses sévères ont été posées sur ce problème, à savoir : affecter un seul passager par conducteur et il faut que le trajet du passager soit inclus dans le trajet du conducteur.

La figure 2.2 montre les améliorations que nous avons apportées et notre contribution par rapport au travail de l'année passée.

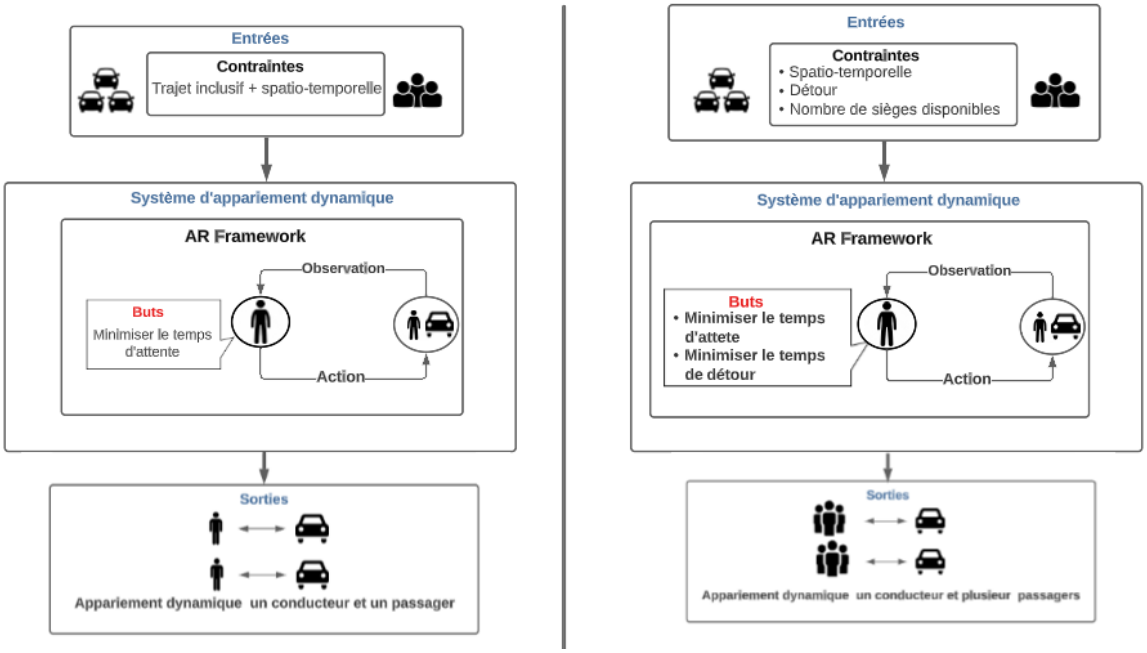

FIGURE 2.2 – Comparaison entre le travail de (TOUATI, 2022), à gauche et notre contribution, à droite.

Notre objectif principal dans cette étude est d'élargir le problème que nous avons traité l'année dernière en tenant compte de plusieurs passagers dans une seule affectation. En intégrant cette nouvelle dimension, nous souhaitons prendre en considération des trajets exclusifs et introduire la contrainte de détour. Ainsi, nous visons à optimiser deux objectifs : minimiser le temps d'attente des passagers et réduire le temps de détour pour les conducteurs. Cette approche nous permet de satisfaire les besoins des deux parties impliquées dans le système de covoiturage.

De plus, nous souhaitons également aborder le passage à l'échelle de notre solution. Avec l'expansion potentielle de l'utilisation du covoiturage, il est essentiel de garantir que notre approche puisse gérer efficacement un grand nombre de passagers et de conducteurs. Nous explorons donc des méthodes et des techniques qui nous permettront de gérer cette croissance et de fournir des solutions de covoiturage optimales même dans des scénarios à grande échelle. En prenant en compte ces aspects, nous cherchons à fournir une approche robuste et évolutive pour résoudre le problème d'appariement dynamique dans le covoiturage.

Nous présentons dans ce qui suit la modélisation des différents paramètres du problème par les différents paramètres de la base théorique de l'apprentissage par renforcement, à

savoir les processus décisionnels de Markov qui sont composés d'un espace d'état, d'un espace d'action, d'une fonction de transition et d'une fonction de récompense. Dans ce contexte, l'agent est considéré comme le conducteur.

2.4.1 Etat

L'état, s est capturé par les coordonnées géographiques au point l du conducteur, du passager, du point de destination suivant et du temps, c'est-à-dire
$s_t = (C_l, P_l, N_l, S_t, t, Dst_l, Arr_l, DP_l)$, ces éléments représentent :

- C_l :est la paire de coordonnées GPS (latitude, longitude) des points de départ et d'arrivée du conducteur,
- P_l : est la paire de coordonnées GPS (latitude, longitude) des points de départ et d'arrivée de la nouvelle demande,
- N_l : est la paire de coordonnées GPS (latitude, longitude) du point de destination suivant.
- t : est le temps.
- S_t : le nombre de siège disponible au moment t .
- Dst_l : l'ensemble de coordonnées GPS (latitude , longitude) des point d'arrivées des passagers à bord .
- Arr_l : l'ensemble des temps d'arrivées au plus tard des passagers à bord .
- DP_l : l'ensemble de coordonnées GPS (latitude , longitude) des points de l'itinéraire de détour .

2.4.2 Observation

L'observation, **o** est l'information extraite de l'environnement qui représente l'état actuel $\mathbf{s_t}$ au moment **t**. Nous définissons l'observation $\mathbf{o_t}$ au moment **t** comme un tuple
$\mathbf{o_t} = (ND_t, PW_t, PMW_t, MD_t, DT_t)$, où :

- $\mathbf{ND_t}$ est le temps d'arrivée à la prochaine destination.
- $\mathbf{PW_t}$ est le temps d'attente actuel du passager.
- $\mathbf{PMW_t}$ est le temps d'attente maximum du passager.
- $\mathbf{S_t}$ est le nombre de sièges disponibles au moment t.

- $\mathbf{MD_t}$ est le temps maximal de détour au memonet t.
- $\mathbf{DT_t}$ est la différence entre les durrées de l'itinéraire initial avec détour et sans détour.

Notre choix peut être motivé par le fait que notre objectif est de minimiser le temps d'attente des passagers et le temps de détour des conducteurs. Ainsi, retourner à l'agent l'ensemble d'information relatif à prendre la bonne décision par rapport au détour et par rapport à l'attente des passagers est nécessaire. Nous supposons que les paramètres temporels sont des paramètres tranchant pour la décision de l'agent avant même de considérer les emplacements.

2.4.3 Action

Deux actions sont disponilbes pour le conducteur qui reflètent sa décision :

- $a = 0 \rightarrow$ refuser la demande
- $a = 1 \rightarrow$ accepter la demande

2.4.4 Récompense

Afin de réaliser nos objectifs principaux du système, qui consistent à réduire le temps d'attente et le temps de détour, il est essentiel que le signal de récompense de notre problème contienne des informations qui guident l'action de l'agent. Par conséquent, nous proposons de définir le signal de récompense (r) en fonction de la diminution du temps d'attente des passagers et du temps de détour des conducteurs lors d'un trajet spécifique.

Ce signal de récompense mesurera le gain en termes du temps du côté passager et du côté conducteur. Nous définissons la récompense comme suit :

$$r = PMW_t - (PW_t + ND_t) + MD_t - DT_t$$

Où : PMW_t : est le temps d'attente maximum du passager, PW_t : est le temps d'attente actuel du passager , ND_t : est le temps d'arrivée à la prochaine destination , MD_t :est le temps maximal de détour et DT_t la différence entre les durrées de l'itinéraire initial avec détour et sans détour.

Exemple motivant

A travers cet exemple, nous allons motiver notre choix de récompense.

Dans cet exemple, nous supposons que le conducteur a reçu deux demandes P1 et P2 au même temps, au point 2. La destination du passager P1 est le point 7 et la destination du passager P2 est le point 6. Les deux points de destination nécéssitent de faire un détour pour pouvoir déposer les passagers.

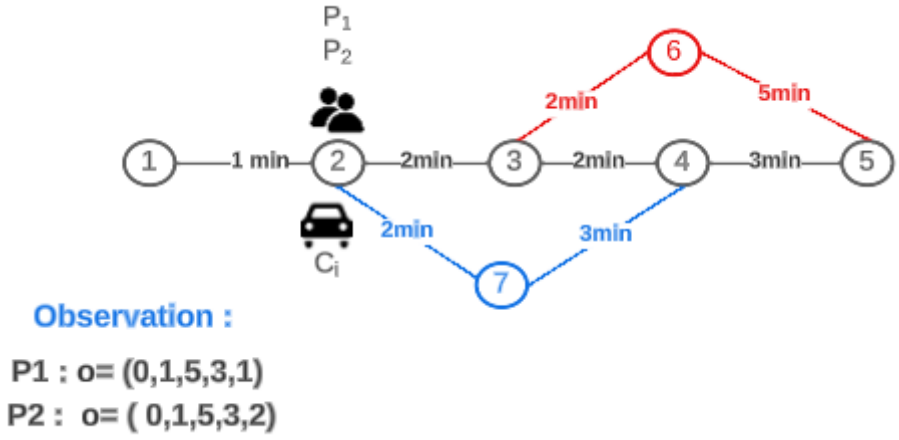

FIGURE 2.3 – Exemple motivant la récompense.

Si le conducteur décide de prendre le passager P1, la récompense sera : r = 5-(1+0)+3-1 = 6.

Si le conducteur décide de prendre le passager P2, la récompense sera : r = 5-(1+0)+3-2=5

Dans ce cas le conducteur va choisir de prendre le passager P1 , car il gagne plus de temps que s'il prend P2.

La figure 2.4 englobe les différents composants de notre modèle.

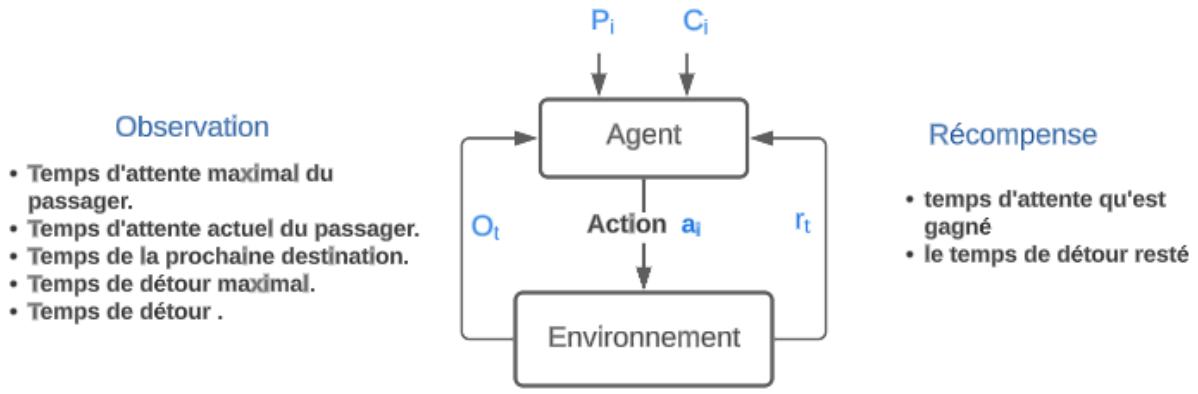

FIGURE 2.4 – Modélisation proposée.

2.4.5 Épisode

Un épisode, c'est le point du temps où l'agent prend une décision. Dans notre travail, chaque point où se trouve le conducteur représente un épisode (c'est à dire de C_l à C_{l+1} est un episode).

2.4.6 Transition d'état

Lorsqu'un passager est déposé à sa destination, cela entraîne un changement dans la liste des points d'arrivée des passagers présents dans l'état. Cette modification de la liste correspond à une transition vers un nouvel état. Le nouvel état reflète l'état actuel du système après avoir déposé le passager à sa destination.

2.5 Phase d'apprentissage

Une phase importante dans notre travail est de construire le modèle sus-présenté, l'injecter à l'agent et lui faire apprendre un comportement décisionnel qui répond aux différentes contraintes du problème et qui optimise les objectifs considérés.

Nous allons, à présent, présenter cette étape d'apprentissage avec ces deux sous-étapes : une première sous-étape qui a comme objectif d'initialiser l'environnement et une deuxième sous-étape qui concerne l'algorithme de décision de l'agent que nous proposons.

2.5.1 Initialisation de l'environnement

Pour construire notre modèle, nous avons besoin d'un ensemble de données contenant les informations essentielles correspondant à la description du problème, notamment les directions de chaque trajet. Pour cela, nous avons décidé d'utiliser la base de données Uber Pickup dans la ville de New York [1]. Cependant, afin d'obtenir un ensemble de données plus détaillé, nous allons utiliser l'API MapQuest pour générer un nouvel ensemble de données qui inclut les informations de direction dont nous avons besoin.

Une fois que nous avons généré cet ensemble de données, notre prochaine étape consiste à préparer l'environnement d'apprentissage en extrayant les informations pertinentes pour notre modèle et les contraintes du problème. Pour chaque conducteur, nous extrairons leurs points de départ et d'arrivée, ainsi que leurs temps de départ et d'arrivée. Nous prendrons également en compte la capacité de leur véhicule et le temps maximal de détour autorisé.

De plus, nous diviserons l'itinéraire de chaque conducteur en un ensemble de points de contrôle, qui représentent des emplacements spécifiques où le conducteur vérifiera la disponibilité des demandes de covoiturage.

De manière similaire, pour chaque passager, nous extrairons leurs points de départ et d'arrivée correspondants. De plus, nous récupérerons les temps de départ et d'arrivée au plus tard pour chaque passager. Le temps d'attente d'un passager sera défini comme la période entre le dépôt de la demande et le temps de départ au plus tard. Nous choisirons ce délai de manière aléatoire, dans une plage de 5 à 30 minutes, pour simuler les variations dans le temps d'attente des passagers.

Cette approche nous permet de modéliser et de résoudre efficacement le problème de covoiturage en prenant en compte les différentes contraintes et conditions des utilisateurs.

2.5.2 Décision de l'agent

Dans le processus de prise de décision, le conducteur peut recevoir deux types de demandes : celles qui s'inscrivent dans son itinéraire initial et celles qui nécessitent un détour. Les demandes sont évaluées en fonction de plusieurs critères, tels que le temps

d'attente actuel du passager, son temps d'attente maximal, le temps restant avant sa prochaine destination et le nombre de sièges disponibles dans le véhicule. Dans le cas des demandes nécessitant un détour, des facteurs supplémentaires entrent en jeu, notamment le temps d'arrivée au plus tard du conducteur, son temps maximal de détour, la destination du détour et le temps d'arrivée au plus tard des passagers déjà à bord. Sur la base de ces critères, l'agent prend la décision d'accepter ou de rejeter la demande.

Une fois qu'une action est sélectionnée, la récompense est calculée en fonction de l'état actuel de l'agent. Si le conducteur a déjà démarré le trajet ou si tous les sièges sont occupés, il se dirige vers sa prochaine destination. Sinon, il évalue s'il est temps de partir ou s'il doit encore attendre pour maximiser l'efficacité du covoiturage. Après avoir pris une décision et accompli la tâche, l'environnement renvoie un nouvel état qui reflète les nouvelles informations disponibles.

Ce processus de prise de décision en boucle est itéré jusqu'à ce que le conducteur atteigne sa destination finale. À chaque itération, l'agent évalue les actions prises et apprend des résultats. Si une décision s'est avérée peu favorable, l'agent prendra en compte cette expérience lorsqu'il sera confronté à des situations similaires à l'avenir, afin d'éviter de reproduire les mêmes erreurs. Grâce à cet apprentissage itératif, l'agent s'améliore progressivement dans ses choix et prend des décisions plus avantageuses au fil du temps.

La figure 2.5 décrit le processus de décision de l'agent à la réception d'une nouvelle demande.

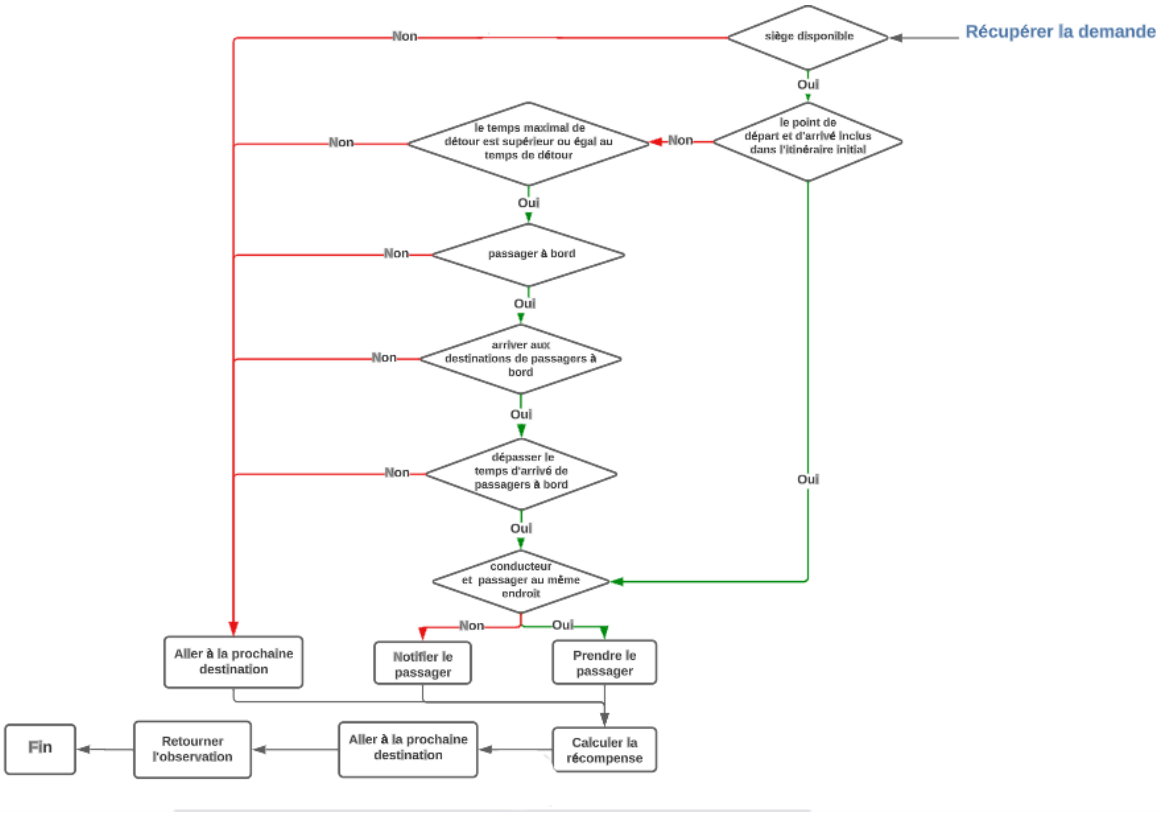

FIGURE 2.5 – Processus de décision

Nous donnons dans ce qui suit, l'algorithme de décision de l'agent que nous proposons.

Algorithme de décision de l'agent

Les entrées : $C_k(t)$, $P_i(t)$.

Les sorties : $(C_k(t), P_i(t))$

Si $S_k \neq 0$ **alors**

Si $Po_i \subset DP_l$ or $Pd_i \subset DP_l$ **alors**

Si $DT_t \leq MD_t$ **alors**

Si $Dst_l \subset DP_l$ **alors**

While $(Arr_i \in Arr_l)$ **faire**

Si $t + DT_t + ND_t > Arr_i$ **alors**

aller à la prochaine destination

Prendre le passager

Sinon aller à la prochaine destination

Sinon aller à la prochaine destination

Sinon Prendre le passager

Sinon aller à la prochaine destination

Retourner $(C_k(t), P_i(t))$

2.5.3 Apprentissage

Tout d'abord , pour créer un modèle, nous avons besoin d'un ensemble de données contenant les informations pertinentes liées à notre problème, telles que les directions de chaque trajet. Pour cela, nous utiliserons la base de données *Uber Pickup* [1] pour la ville de New York. Les données de ce répertoire couvrent plus de 4,5 millions de trajets Uber effectués dans la ville de New York entre avril et septembre 2014 .

Nous allons utiliser un fichier spécifique de ce répertoire appelé *"Federal_02216"*, qui contient des données brutes sur les prises en charge d'une entreprise de FHV non Uber. il contient les informations notamment : *Date* , *Time* , *PU_Adress*, *DO_Adress*, *Routing Details*, *Status* .

Nous avons choisi d'utiliser l'algorithme d'apprentissage par renforcement appelé Optimisation de la politique proximale (PPO). PPO est considéré comme l'un des algorithmes les plus simples et les plus populaires pour l'apprentissage par renforcement. Il s'agit d'une méthode sans modèle qui utilise le gradient de politique et peut être appliquée à des environnements avec des espaces d'action discrets ou continus. Cette technique consiste à interagir avec l'environnement pour collecter des données, puis à optimiser une fonction objective en utilisant le gradient de montéé (Schulman et al., 2017).

2.6 Conclusion

Dans ce chapitre, nous avons présenté la modélisation que nous avons proposée pour résoudre le problème d'appariement dynamique sous contraintes à l'aide de l'apprentissage par renforcement. Nous avons décrit en détail les différentes étapes que notre système traverse pour construire un modèle et ensuite exécuter le processus d'appariement.

Dans le chapitre suivant, nous allons fournir des informations détaillées sur l'implémentation de notre application.

Chapitre 3 : Implémentation

3.1 Introduction

Dans ce chapitre, nous allons aborder les aspects applicatifs de notre travail et fournir les détails de mise en œuvre de notre solution. Nous commençons par décrire les outils et le langage de programmation utilisés pour l'implémentation. Ensuite, nous présentons la base de données que nous avons utilisée pour entraîner et valider notre modèle. Nous expliquons ensuite en détail l'implémentation et le processus d'apprentissage et de validation de notre système. Enfin, nous abordons le simulateur que nous avons développé ainsi que les résultats obtenus lors de nos expérimentations.

3.2 Mise en oeuvre de la contribution

Dans cette section, nous allons commencer par spécifier les outils que nous avons employés lors du développement de notre application. Nous allons fournir, par la suite, une description détaillée de notre système, en expliquant ses composantes et son fonctionnement.

3.2.1 Langages et outils de développement

- ***Python*** **:** est un langage de programmation flexible connu pour sa lisibilité et sa facilité d'utilisation. Sa sémantique dynamique et ses structures de données intégrées le rendent idéal pour le développement rapide d'applications et les tâches de script. Python prend en charge les modules et les paquets, ce qui favorise la modularité et la réutilisation du code. L'interpréteur Python et la vaste bibliothèque standard sont disponibles gratuitement pour toutes les principales plateformes, ce qui le rend accessible aux développeurs du monde entier [2].
- ***Visual Studio Code :*** Visual Studio Code est un éditeur de code source puissant et léger qui fonctionne sur plusieurs plateformes, notamment Windows, macOS et

Linux. Il offre une prise en charge intégrée pour les langages JavaScript, TypeScript et Node.js, ainsi qu'un large éventail d'extensions pour d'autres langages et environnements d'exécution tels que C++, C#, Java, Python, PHP,etc [3] .

- ***Lucidchart :*** est une plateforme de travail visuelle qui intègre la création de diagrammes, la visualisation de données et des fonctionnalités de collaboration pour améliorer la communication, favoriser l'innovation et faciliter la coopération au sein d'une équipe entièrement distribuée [4].
- ***Les bibliothèques utilisées***
  1. **Gym :** est une bibliothèque Python open source qui facilite le développement et la comparaison d'algorithmes d'apprentissage par renforcement. Elle fournit une interface standardisée pour la communication entre les algorithmes d'apprentissage et les environnements, ainsi qu'un ensemble prédéfini d'environnements compatibles avec cette interface [5].
  2. **Pandas :** est une bibliothèque open source qui offre une analyse et une manipulation des données rapides, puissantes, flexibles et faciles à utiliser.Elle est basé sur le langage de programmation Python [6].
  3. **Numpy :** est une bibliothèque Python essentielle pour le calcul scientifique. Elle offre un objet tableau multidimensionnel, des objets dérivés et une variété de fonctions pour des opérations rapides sur les tableaux, telles que les opérations mathématiques, la manipulation de forme, le tri, etc [7] .
  4. **Stable baselines3 :** (SB3) est une collection d'implémentations fiable d'algorithmes d'apprentissage par renforcement qui sont spécifiquement conçus pour être utilisés avec le framework PyTorch [8] .
  5. **Tensor Board :** est une plateforme complète de visualisation et d'outils pour les tests d'apprentissage automatique . Il permet de suivre et de visualiser des métriques telles que la perte et la justesse, de visualiser le graphe du modèle, d'afficher des histogrammes des pondérations et des biais, de projeter des représentations vectorielles dans un espace de dimension réduite, d'afficher des images, du texte et des données audio, de profiler les programmes TensorFlow, et bien plus encore [9].
  6. **Tkinter :** Tkinter est une bibliothèque Python couramment utilisée pour créer des interfaces graphiques (GUI) en utilisant la boîte à outils Tcl/Tk. Il s'agit

de l'interface Python standard pour la création d'applications GUI. Tkinter est compatible avec la plupart des plateformes Unix, y compris macOS, ainsi que les systèmes Windows. [10] .

- ***Map API : MapQuest***
  - ***L'API Directions*** : est un service en ligne qui permet de demander des itinéraires entre des lieux en envoyant une requête HTTP. Les résultats de cette requête sont renvoyés au format JSON ou XML, fournissant ainsi des informations détaillées sur l'itinéraire entre les emplacements spécifiés [11].

3.3 Echantillon de données

Nous avons sélectionné aléatoirement un échantillon de données provenant du *Federal_02216* [1] afin de mener nos expérimentations . Cet échantillon est composé de 48 itinéraires, et nous avons utilisé l'API MapQuest Direction pour obtenir des informations plus détaillées sur les directions nécessaires à partir de l'ensemble de données d'itinéraire que nous avons . Cela nous permettra de générer un nouvel ensemble de données contenant les informations correspondantes (Figure 3.1) :

- ***Info :*** contient le copyright
- ***Route :*** liste contenant tous les itinéraires possibles, où chaque itinéraire comprend :
  - ***Time :*** la durée de l'itinéraire calculée, exprimée en secondes.
  - ***Distance :***la distance calculée de l'itinéraire.
  - ***Legs :*** contient des informations sur l'itinéraire :
    - ***Maneuvers :*** une liste qui contient des informations spécifiques entre les points, où chaque élément de la liste contient :
      1. ***Index :*** index de chaque point.
      2. ***Time :*** le temps entre un point et le point qui le suit.
      3. ***Distance :*** la distance entre un point et le point qui le suit.
      4. ***StartPoint :***la localisation du premier point, représenté par des coordonnées GPS (latitude *"lat"*, longitude *"lng"*).
  - ***Locations :*** liste contient à la fois l'adresse et les coordonnées GPS (latitude *"lat"*, longitude **"lng"**) des points de départ et d'arrivée.

- ***AlternateRoutes :*** liste qui cotient les mêmes informations de la ***"Route"***: *Time , Distance , Locations , Legs .*

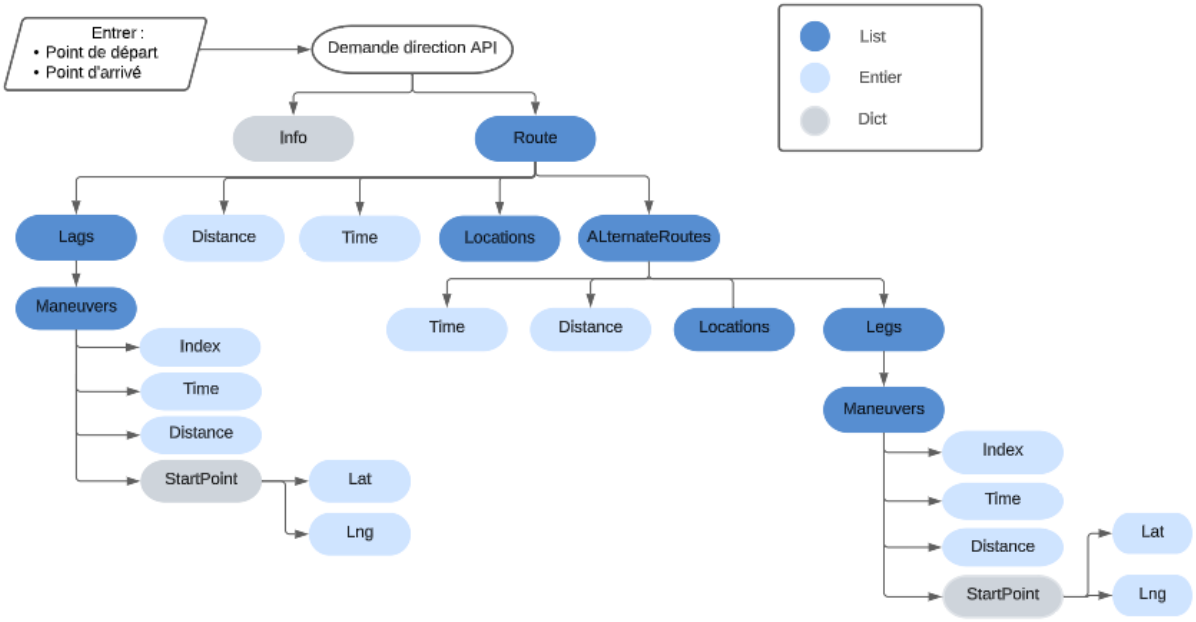

FIGURE 3.1 – Sortie d'une requête de MapQuest Direction API.

3.4 Apprentissage

Notre modèle PPO a été formé en utilisant un ensemble de données réelles de la base de données sus-présentée, contenant les enregistrements des trajets de taxi effectués dans la ville de New York en septembre 2016 .

Pour évaluer les résultats de l'apprentissage, nous avons considéré le trajet complet d'un conducteur comme un épisode afin de calculer la courbe de récompense. La durée totale de l'apprentissage s'élève à 800 000 épisodes avec gamma =0.9, au cours desquels l'agent cherche à maximiser les récompenses cumulées.

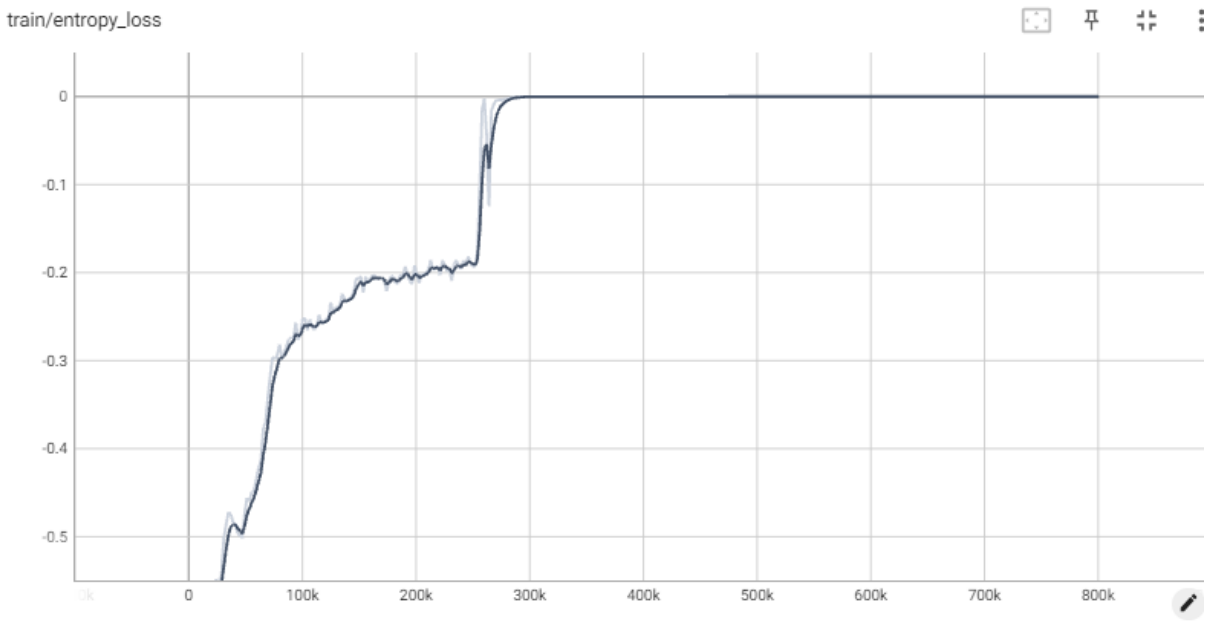

FIGURE 3.2 – Courbe de la perte d'entropie.

Avant de procéder à la simulation en temps réel pour évaluer les réactions de l'agent, nous avons effectué des tests préliminaires sur plusieurs scénarios. L'objectif était de vérifier si l'agent prenait les bonnes décisions de manière cohérente. Pour ce faire, nous avons soumis le modèle à diverses observations, afin d'évaluer ses prédictions dans des contextes variés. .

Les différentes observations fournies à l'agent englobaient un ensemble de facteurs cruciaux qui influencent ses décisions. Cela comprenait : le nombre de sièges disponibles, les temps d'attente des passagers, le temps maximal de détour du conducteur, ainsi que les points de départ et d'arrivée des trajets, qu'ils nécessitent ou non un détour. L'analyse de ces différentes variables nous a permis d'évaluer la capacité de l'agent à prendre des décisions pertinentes dans une variété de situations.

La figure 3.3 montre quelques exemples des différentes observations auxquelles notre agent a été soumis pour tester la robustesse de son comportement.

FIGURE 3.3 – Exemples de tests.

3.5 Simulateur

Pour initialiser l'environnement, nous avons lancé la simulation de 08 :00 à 21 :00. Dans le système, nous avons fixé l'intervalle de réception de nouvelles demandes et offres à 5 minutes ($\Delta t = 5$ min). Cela signifie que toutes les 5 minutes, le système reçoit un nombre aléatoire d'offres et de demandes. Nous avons fixé ce nombre entre 4 et 10 pour les offres des conducteurs et entre 10 et 18 pour les demandes des passagers.

Les emplacements initiaux de ces offres et demandes sont sélectionnés de manière aléatoire à partir de l'ensemble de données .

De plus, nous avons attribué un temps d'attente maximal à chaque passager de manière aléatoire, avec une durée comprise entre 10 et 30 minutes.

Du côté conducteur, nous avons affecté un temps de détour maximal de manière aléatoire entre 5 et 15 minutes. Quant au nombre de sièges disponibles des véhicules, nous l'avons fixé à 3 .

La figure 3.4 présente l'interface principale de notre simulateur.

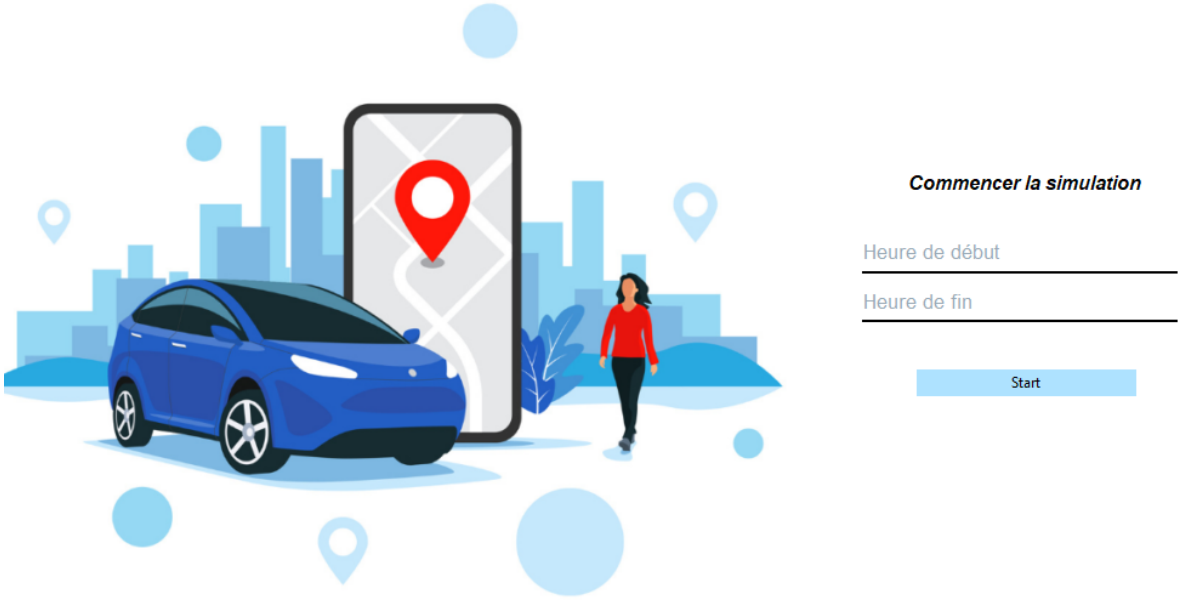

FIGURE 3.4 – Interface principal du simulateur.

Nous allons présenter, dans ce qui suit, les résultats relatifs à un échantillon de données de l'ensemble de données qui a été généré entre 08 :00 et 21 :00.

Les détails de l'échantillon de données sont donnés dans le tableau 3.1.

Offres	**Temps de détour maximal**
Conducteur 3	5 min
Conducteur 12	5 min
Conducteur 23	7 min
Conducteur 39	8 min
Conducteur 48	6 min

(a) Echantillon des conducteurs

Demandes	**Temps d'attente**	**Détour**
Passager 12	2 min	Non
Passager 18	5 min	Oui
Passager 25	1 min	Non
Passager 26	9 min	Oui
Passager 71	5 min	Oui
Passager 72	3 min	Non
Passager 75	2 min	Non
Passager 87	6 min	Non

(b) Echantillon des passagers

TABLE 3.1 – Echantillon de données.

Les résultats de l'appariement dynamique entre l'échantillon des conducteurs et l'échantillon des passagers est donné dans le tableau 3.2

Conducteurs	**Passagers**	**Heure d'appariement**
Conducteur 3	Passager 12	à 7h01
Conducteur 12	Passager 25	à 7h13
Conducteur 23	Passager 26 et Passager 18	à 7h16 et à 7h33
Conducteur 39	Passager 72 et Passager 71	à 7h22 et à 7h22
Conducteur 48	Passager 75 et Passager 87	à 7h26 et à 7h26

TABLE 3.2 – Résultats d'appariement dynamique.

Les statistiques de la simulation ainsi que la récompense total cumulée (gain en temps) sont données dans la figure 3.5 et le tableau 3.3.

Résultats de la simulation de 08 :00 à 21 :00	
Nombre de demandes reçues	2960
Nombre de demandes acceptées	1167
Nombre de demandes acceptées sans détour	965
Nombre de demandes acceptées avec détour	202
Récompense totale	20000
Temps de détour et d'attente totaux	16803

TABLE 3.3 – Les résultats de la simulation.

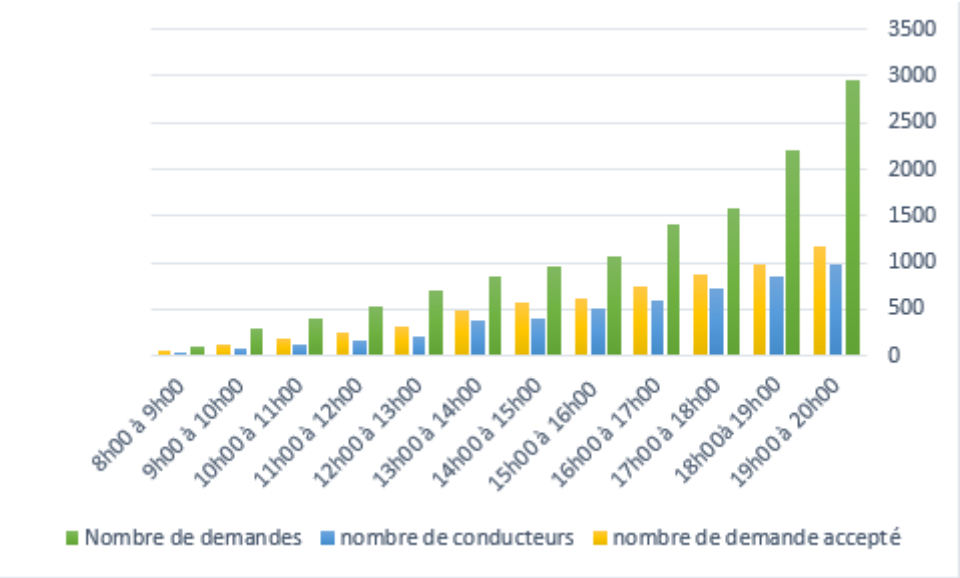

FIGURE 3.5 – Visualisation des résultats de la simulation.

L'interface relative à la visualisation des résultats de la simulation est donnée dans la figure 3.6.

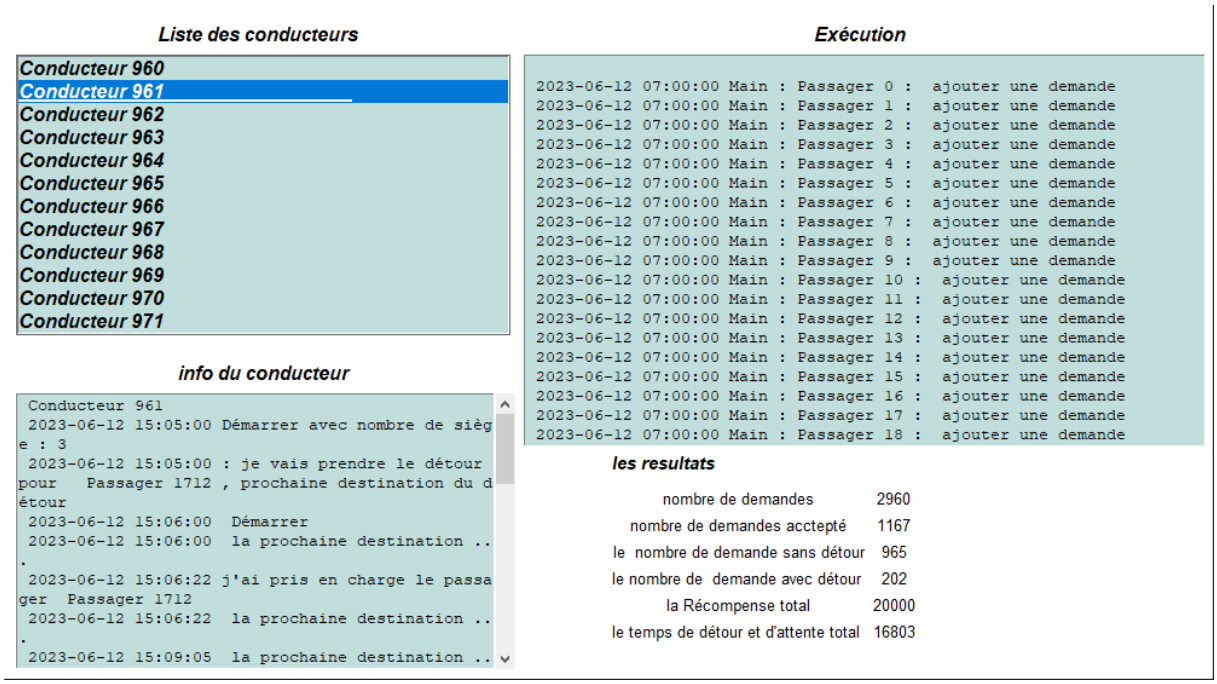

FIGURE 3.6 – Résultats de la simulation.

Les données obtenues à partir de la simulation, telles qu'elles sont exposées dans le tableau 3.3 et représentées dans les figures 3.6 et 3.5, mettent en évidence l'efficacité de notre approche.

La récompense totale obtenue, qui interprète le gain en temps résultant des décisions avisées prises par les conducteurs en ce qui concerne les détours et le temps d'attente des passagers, s'élève à 20000 minutes pour une simulation de 8h00 à 21h00. Ce chiffre remarquable démontre clairement les avantages significatifs de notre application.

En optimisant la décision de détour, notre modèle a permis aux conducteurs de minimiser le temps supplémentaire passé sur la route tout en respectant les besoins des passagers. De plus, en optimisant le temps d'attente des passagers, notre application a considérablement amélioré leur expérience en réduisant le temps global d'attente. Ces résultats soulignent l'efficacité de notre approche dans la prise de décisions optimales pour les conducteurs et dans l'amélioration de l'expérience des passagers.

3.6 Expérimentations

Dans le but d'évaluer plus efficacement la méthode proposée, des expérimentations ont

été réalisées dans différents environnements et configurations. Deux scénarios distincts ont été créés, chacun couvrant une plage horaire de 7h00 à 12h00. Les emplacements initiaux des offres de covoiturage et des demandes ont été choisis de manière aléatoire à partir de notre ensemble de données. Pour chaque conducteur, nous avons fixé le nombre de sièges disponibles à 3, tandis que le temps d'attente des passagers a été généré aléatoirement entre 10 et 25 minutes.

En outre, nous avons fait varier le nombre d'offres de covoiturage de 3 à 10, ainsi que le nombre de demandes de 9 à 15. Ces offres et demandes étaient générées toutes les 5 minutes et soumises au système.

La différence majeure entre les deux scénarios réside dans le temps de détour maximal, qui est le paramètre clé que nous souhaitons tester dans notre travail. Dans le premier scénario, nous avons fixé un temps de détour maximal relativement court, tandis que dans le deuxième scénario, nous avons augmenté ce temps de détour maximal pour évaluer la capacité de notre système à gérer des situations où les conducteurs sont prêts à effectuer des détours plus importants pour satisfaire les demandes des passagers.

Premier scénario : Dans ce scénario, nous avons fixé le temps de détour maximal de manière aléatoire, avec une plage allant de 5 à 10 minutes.

Le tableau 3.4 présente les résultats obtenus.

Plages horaires	**7h00-8h00**	**8h00-9h00**	**9h00-10h00**	**10h00-11h00**	**11h00-12h00**
Nombre des offres ajoutées	60	100	165	211	300
Nombre de demandes reçues	120	230	352	461	510
Nombre de demandes acceptées sans détour	37	81	134	172	238
Nombre de demandes acceptées avec détour	25	48	62	73	90
Récompense totale	1368 min	2725 min	4237 min	5800 min	6604 min
Temps de détour et d'attente totaux	1145 min	2434 min	3120 min	4012 min	5003 min

TABLE 3.4 – Résultats du premier scénario.

Deuxiéme scénario : Nous avons fixé le temps de détour maximal à 20 min . Le tableau 3.5 présente les résultats obtenus.

Plages horaires	**7h00-8h00**	**8h00-9h00**	**9h00-10h00**	**10h00-11h00**	**11h00-12h00**
Nombre des offres ajoutées	69	159	218	303	388
Nombre de demandes reçues	150	304	445	629	834
Nombre de demandes acceptées sans détour	43	80	129	171	213
Nombre de demandes acceptées avec détour	20	45	70	94	120
Récompense totale	1573 min	3662 min	5517 min	7194 min	9239 min
Temps de détour et d'attente totaux	1320 min	2844 min	4010 min	5020 min	7050 min

TABLE 3.5 – Résultats du deuxiéme scénario.

Les expérimentations que nous avons mené ont pour objectif d'évaluer la capacité d'adaptation de notre système face à des situations où les conducteurs peuvent être confrontés à des détours de durées variables.

En comparant les résultats obtenus pour différents intervalles de temps de détour maximal, nous avons pu constater que le nombre de demandes acceptées avec détour et sans détour ne révèle pas un grand écart (12 demandes pour le premier scénario et 23 demandes pour le deuxième). Cette constatation témoigne de l'efficacité de notre algorithme de décision et du raisonnement de l'agent. En effet, notre système parvient à prendre des décisions équilibrées en tenant compte des demandes nécessitant un détour et de celles pouvant être intégrées à l'itinéraire initial et le temps d'attente des passagers.

En outre, La variation du nombre d'offres et de demandes nous a permis d'explorer différentes charges de travail et de tester la capacité de notre système à traiter efficacement un volume croissant de données en temps réel. En analysant les performances du système dans ces différentes conditions, nous avons pu évaluer sa robustesse et sa capacité à s'adapter à des situations de demande variable.

3.7 Conclusion

Dans ce chapitre, nous avons détaillé l'implémentation de notre approche et de notre simulateur, en mettant en évidence les résultats obtenus à chaque étape de réalisation. De plus, nous avons mené des expérimentations afin d'évaluer les performances de notre système. Les résultats obtenus sont extrêmement encourageants et viennent confirmer la justesse de nos choix et de notre modélisation. Ils démontrent l'efficacité de notre approche dans la résolution du problème posé et renforcent notre confiance dans sa capacité à fournir des solutions pertinentes pour la gestion des trajets en covoiturage.

Conclusion générale

Le covoiturage dynamique offre une grande flexibilité d'utilisation. Il permet de trouver rapidement un trajet partagé en fonction de la position des conducteurs potentiels pour un itinéraire donné. Il s'agit d'un processus automatisé dans lequel un fournisseur de services met en relation des conducteurs et des passagers ayant des itinéraires et des horaires similaires afin de partager un trajet .

Les systèmes de covoiturage sont intrinsèquement dynamiques, leur complexité réside principalement dans l'appariement d'individus soumis à des contraintes spatio-temporelles. Ces contraintes doivent être précisées à la fois par les conducteurs et les passagers avant que le trajet souhaité puisse être établi et effectué.

Dans ce travail, nous avons proposé une modélisation des composants du problème d'appariement dynamique à contraintes par les composant du processus décisionnel de Markov, qui constitue la base théorique de l'apprentissage par renforcement. Notre modélisation se concentre sur la dimension temporelle et vise à minimiser le temps d'attente des passagers et le temps de détour des conducteurs. En réduisant ces paramètres, nous sommes en mesure de satisfaire les passagers et d'augmenter le nombre de demandes réalisées avec succès.

La validité de notre proposition a été établie en utilisant un simulateur que nous avons développé afin de capturer la dynamique du système. Les résultats obtenus à partir des différentes expérimentations ont démontré l'efficacité de notre système, ainsi que sa robustesse et sa capacité à répondre à diverses demandes soumises à différentes contraintes. Ces résultats ont confirmé notre hypothèse selon laquelle le temps joue un rôle crucial dans la prise de décision.

L'impact positif de notre application sur l'efficacité globale du système de transport est donc clairement établi grâce à ces résultats prometteurs. Il est encourageant de constater que notre modèle a su réduire le temps d'attente des passagers et minimiser les détours

des conducteurs, offrant ainsi une solution efficace et bénéfique pour toutes les parties impliquées dans le processus de covoiturage.

Des perspectives d'amélioration de notre travail restent, toutefois, indispensables.

Nous visons d'explorer encore les approches d'apprentissage par renforcement et plus précisément l'apprentissage par renforcement profond afin de réduire l'espace d'état et de recherche et pouvoir ainsi expérimenter des instances à grandes échelles.

De plus, nous envisageons aussi d'intégrer les préférence des utilisateurs. Actuellement, notre approche se concentre principalement sur l'optimisation du temps d'attente et du temps de détour. Une perspective intéressante serait d'intégrer les préférences des utilisateurs dans le processus de décision, en tenant compte de leurs critères individuels tels que le confort, le partage des frais, la proximité des itinéraires, etc.

Bibliographie

Agatz, N., Erera, A., Savelsbergh, M., & Wang, X. (2012). Optimization for dynamic ride-sharing : A review. *European Journal of Operational Research*, *223*(2), 295-303.

Agatz, N., Erera, A. L., Savelsbergh, M. W., & Wang, X. (2011). Dynamic ride-sharing : A simulation study in metro Atlanta. *Procedia-Social and Behavioral Sciences*, *17*, 532-550.

Al-Abbasi, A. O., Ghosh, A., & Aggarwal, V. (2019). Deeppool : Distributed model-free algorithm for ride-sharing using deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, *20*(12), 4714-4727.

Dangeti, P. (2017). *Statistics for machine learning*. Packt Publishing Ltd.

Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X., & Koenig, S. (2013). Ridesharing : The state-of-the-art and future directions. *Transportation Research Part B : Methodological*, *57*, 28-46.

Herbawi, W. (2013). *Solving the ridematching problem in dynamic ridesharing* (thèse de doct.). Universität Ulm.

Jin, J., Zhou, M., Zhang, W., Li, M., Guo, Z., Qin, Z., Jiao, Y., Tang, X., Wang, C., Wang, J., et al. (2019). Coride : joint order dispatching and fleet management for multi-scale ride-hailing platforms. *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1983-1992.

Jindal, I., Qin, Z. T., Chen, X., Nokleby, M., & Ye, J. (2018). Optimizing taxi carpool policies via reinforcement learning and spatio-temporal mining. *2018 IEEE International Conference on Big Data (Big Data)*, 1417-1426.

Ke, J., Xiao, F., Yang, H., & Ye, J. (2020). Learning to delay in ride-sourcing systems : a multi-agent deep reinforcement learning framework. *IEEE Transactions on Knowledge and Data Engineering*, *34*(5), 2280-2292.

Knox, W. B., & Stone, P. (2012). Reinforcement learning from simultaneous human and MDP reward. *AAMAS*, *1004*, 475-482.

Kullman, N. D., Cousineau, M., Goodson, J. C., & Mendoza, J. E. (2022). Dynamic ride-hailing with electric vehicles. *Transportation Science*, *56*(3), 775-794.

Li, M., Qin, Z., Jiao, Y., Yang, Y., Wang, J., Wang, C., Wu, G., & Ye, J. (2019). Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. *The world wide web conference*, 983-994.

Martins, L. d. C., de la Torre, R., Corlu, C. G., Juan, A. A., & Masmoudi, M. A. (2021). Optimizing ride-sharing operations in smart sustainable cities : Challenges and the need for agile algorithms. *Computers & Industrial Engineering*, *153*, 107080.

Mourad, A., Puchinger, J., & Chu, C. (2019). A survey of models and algorithms for optimizing shared mobility. *Transportation Research Part B : Methodological*, *123*, 323-346.

Plate, O. (2019). *Ridesharing with Multiple Riders* (thèse de doct.). Karlsruhe Institute of Technology Karlsruhe, Germany.

Qin, Z., Tang, X., Jiao, Y., Zhang, F., Xu, Z., Zhu, H., & Ye, J. (2020). Ride-hailing order dispatching at didi via reinforcement learning. *INFORMS Journal on Applied Analytics*, *50*(5), 272-286.

Qin, Z. T., Zhu, H., & Ye, J. (2021). Reinforcement learning for ridesharing : A survey. *2021 IEEE international intelligent transportation systems conference (ITSC)*, 2447-2454.

Qin, Z. T., Zhu, H., & Ye, J. (2022). Reinforcement learning for ridesharing : An extended survey. *Transportation Research Part C : Emerging Technologies*, *144*, 103852.

Santos, D. O., & Xavier, E. C. (2015). Taxi and ride sharing : A dynamic dial-a-ride problem with money as an incentive. *Expert Systems with Applications*, *42*(19), 6728-6737.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv :1707.06347*.

Singh, A., Al-Abbasi, A. O., & Aggarwal, V. (2021). A distributed model-free algorithm for multi-hop ride-sharing using deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, *23*(7), 8595-8605.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning : An introduction*. MIT press.

Tang, X., Zhang, F., Qin, Z., Wang, Y., Shi, D., Song, B., Tong, Y., Zhu, H., & Ye, J. (2021). Value function is all you need : A unified learning framework for ride hailing platforms. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 3605-3615.

Ting, K. H., Lee, L. S., Pickl, S., & Seow, H.-V. (2021). Shared mobility problems : a systematic review on types, variants, characteristics, and solution approaches. *Applied Sciences*, *11*(17), 7996.

TOUATI, I. (2022). Méthode d'optimisation pour la résolution du problème de covoiturage dynamique.

Wang, Z., Qin, Z., Tang, X., Ye, J., & Zhu, H. (2018). Deep reinforcement learning with knowledge transfer for online rides order dispatching. *2018 IEEE International Conference on Data Mining (ICDM)*, 617-626.

Wiering, M. A., & Van Otterlo, M. (2012). Reinforcement learning. *Adaptation, learning, and optimization*, *12*(3), 729.

Xu, Z., Li, Z., Guan, Q., Zhang, D., Li, Q., Nan, J., Liu, C., Bian, W., & Ye, J. (2018). Large-scale order dispatch in on-demand ride-hailing platforms : A learning and planning approach. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 905-913.

Zhou, M., Jin, J., Zhang, W., Qin, Z., Jiao, Y., Wang, C., Wu, G., Yu, Y., & Ye, J. (2019). Multi-agent reinforcement learning for order-dispatching via order-vehicle distribution matching. *Proceedings of the 28th ACM international conference on information and knowledge management*, 2645-2653.

Webographie

1. Kaggle [En ligne] . Available : https://www.kaggle.com/datasets/fivethirtyeight/uber-pickups-in-new-york-city .[Accès le 01 06 2023].
2. Python [En ligne] . Available : https://www.python.org/doc/essays/blurb/.[Accès le 01 06 2023].
3. visual studio [En ligne] . Available : https://code.visualstudio.com/docs.[Accès le 01 06 2023].
4. Lucidchart [En ligne] . Available : https://www.lucidchart.com/pages/product. [Accès le 01 06 2023].
5. Gym [En ligne] . Available : https://www.gymlibrary.dev/. [Accès le 01 06 2023].
6. Pandas [En ligne] . Available : https://pandas.pydata.org/. [Accès le 01 06 2023].
7. Numpy [En ligne] . Available : https://numpy.org/doc/stable/. [Accès le 01 06 2023].
8. Stable baselines3 [En ligne] . Available : https://stable-baselines3.readthedocs.io/en/master/. [Accès le 01 06 2023].
9. TensorBoard [En ligne] . Available : https://www.tensorflow.org/tensorboard?hl=fr. [Accès le 01 06 2023].
10. Tkinter [En ligne] . Available : https://docs.python.org/3/library/tkinter.html. [Accès le 10 06 2023].
11. Direction API [En ligne] . Available : https://developer.mapquest.com/documentation/directions-api/route/get. [Accès le 01 06 2023].