République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université de 8 Mai 1945 – 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 :

Méthode d'optimisation pour la résolution du problème de covoiturage dynamique

Encadré Par :

Dr. Abdelmoumène Hiba

Présenté par :

Touati Islem

Juin 2022

Remerciement

Je remercie dieu pour m'avoir donné santé, courage et patience afin de m'aider à réaliser ce travail.

Mes remerciements s'adressent à mon encadrante, l'enseignante « Dr. Abdelmoumène Hiba », pour avoir accepté de diriger ce travail, c'est un honneur pour moi.

Pour ses multiples conseils et les efforts pour assurer un travail de qualité, son soutien, ses compétences et sa clairvoyance m'ont été d'une aide inestimable.

Je tiens à remercier sincèrement les membres du jury pour l'intérêt qu'ils ont porté à notre projet en acceptant d'examiner le travail et de l'enrichir par leurs propositions.

Enfin, je remercie tous ceux qui de près ou de loin ont contribué à l'accomplissement de ce travail.

Dédicace

A

Mes parents

Les plus chers au monde Karim et Faiza qui ont œuvrés pour ma réussite et qui n'ont jamais cessé de m'encourager et de me soutenir. Puisse Dieu faire en sorte que ce travail porte son fruit ; Merci pour les valeurs nobles, l'éducation et le soutient permanent. Je vous souhaite une longue vie plein de bonheur.

A

Ma sœur et mes frères

Lina, Issam et Yahia qui ont toujours été là pour moi, des exemples de persévérance, de courage et de générosité, par leurs encouragements continus.

A

Tous amis

Et tous ceux que j'aime et me sont chers et que j'ai omis de citer, tout en leur souhaitant la réussite dans tout ce qu'ils entreprennent.

Touati Islem

Résumé

Le système de covoiturage dynamique est conçu pour mettre en relation des voyageurs ayant des itinéraires et des horaires similaires dans un court laps de temps. Ces systèmes peuvent offrir d'importants avantages sociaux et environnementaux en réduisant le nombre de voitures utilisées. Des techniques d'optimisation efficaces et efficientes qui mettent en relation conducteurs et passagers en temps réel sont l'un des éléments nécessaires à la réussite d'un système de covoiturage dynamique, c'est dans ce cadre que s'inscrit notre travail.

Dans ce travail, nous proposons de résoudre le problème d'appariement dynamique sous contraintes par l'apprentissage par renforcement. Nous avons proposé une modélisation nouvelle axée temps qui vise à minimiser le temps d'attente des passagers. Notre cadre est validé à l'aide de données spatio-temporelles de trajets réels provenant de l'ensemble de données publiques de New York City Taxi ainsi que sur un simulateur que nous avons développé. Les résultats obtenus viennent appuyés nos choix et ont montré l'efficacité et la robustesse de notre approche en temps réel.

Mots clés

Covoiturage dynamique, Appariement dynamique, Optimisation, Apprentissage par renforcement, Contraintes spatio-temporelle.

Abstract

The dynamic ridesharing system is designed to match travelers with similar itineraries and schedules in a short period of time. These systems can provide significant social and environmental benefits by reducing the number of cars used. Effective and efficient optimization techniques that match drivers and passengers in real time are one of the necessary components of a successful dynamic ridesharing system, it is within this framework that our work falls.

In this work, we propose to solve the dynamic matching problem under constraints by reinforcement learning. We have proposed a new time-based modeling that aims to minimize passenger waiting time. Our framework is validated using spatiotemporal data of real rides from the New York City Taxi public dataset as well as a simulator we developed. The results obtained support our choices and have shown the effectiveness and robustness of our approach in real time.

Keywords

Dynamic ridesharing, Dynamic Matching, Optimization, Reinforcement learning, Spatio-temporal constraints.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

L’homme a un besoin impérieux de se déplacer depuis l’allongement des distances et l’apparition des moyens de transports que ce soient publiques ou privés. Ce besoin de mobilité n’est pas sans conséquences et on peut le constater de la pollution atmosphérique, le changement climatique, etc.

Les dernières années ont été marquées par une augmentation significative des demandes de déplacements. Cette croissance nécessite le développement et l'amélioration continue des moyens de transport. Cependant, malgré son développement continu, le transport public reste moins compétitif en termes de demande des utilisateurs, notamment en milieu rural. En raison de leur disponibilité et de leur efficacité, les voitures particulières restent le mode de transport privilégié, en particulier dans ces zones non urbaines. Cependant, la mobilité humaine accrue et l'utilisation intensive des voitures particulières ont eu des effets négatifs sur l'environnement et soulevé des questions sur la qualité de vie.

En effet, l'utilisation excessive de la voiture particulière est source de pollution, d'embouteillages et de stress.

Le passage à de nouveaux modes de transport est une nécessité d’où la naissance du concept de covoiturage qui consiste à mettre en relation des conducteurs non professionnel ayant des offres de covoiturage avec des passagers potentiels ayant des demandes pour voyager ensemble.

Au départ, les plateformes de covoiturage correspondaient aux personnes dont les points de départ et d'arrivée de leurs trajets étaient relativement proches les uns des autres. De telles solutions statiques ont réussi pour certains types de solutions de covoiturage, mais ne fonctionnent pas bien en l'absence d'informations statiques disponibles avant la prise de décision.

De plus, de nos jours, les transports, comme de nombreux autres aspects de la vie quotidienne, sont transformés par la révolution des technologies de l'information. Ces progrès technologiques ont fait appel à un nouveau type de covoiturage appelé covoiturage dynamique qui doit être capable de répondre aux demandes en quelques millisecondes et de fournir des appariements automatiques

plus sophistiquées qu'une simple recherche radiale autour des origines et des destinations des trajets.

Le covoiturage dynamique se caractérise par une grande souplesse d'utilisation et moins d'interdépendance que le covoiturage statique.

Néanmoins, la majorité des systèmes opérationnels qui gèrent la comodalité intègrent le concept de covoiturage comme une simple alternative quand le transport en commun n'est pas disponible. Ce manque d'optimisation présente un frein au développement du covoiturage comme un mode de transport à part entière. En effet, ce processus est beaucoup plus compliqué qu'il le parait et se caractérise par un aspect dynamique qui doit assurer une flexibilité et une instantanéité et un ensemble de contraintes qui doivent être gérées.

C'est dans ce cadre que s'intègre notre travail, nous nous intéressons à explorer les méthodes d'optimisation afin de trouver une solution au problème de gestion dynamique d'un ensemble d'offres de covoiturage envoyées par les conducteurs et un ensemble de requêtes émises par les passagers. Notre objectif est de mettre en place un système qui doit optimiser les solutions d'appariement dynamique (un conducteur-un passager) en minimisant le temps d'attente, maximisant le nombre de requêtes appariées et achevées avec succès, et en respectant un ensemble de contraintes tel que la fenêtre temporelle des utilisateurs, leur localisation et leur chemine sans détour.

Une caractéristique principale du problème de covoiturage dynamique est sa nature spatio-temporelle. L'éligibilité d'un conducteur à correspondre à une requête d'un passager dépend en partie de sa proximité spatiale avec la demande. De plus, les réponses à des demandes de trajet prennent généralement un temps différent pour se terminer, et elles modifient les états spatiaux des conducteurs, affectant la distribution de l'offre pour une correspondance future.

Par conséquent, les décisions opérationnelles en matière de covoiturage sont de nature séquentielle et ont une forte dépendance spatio-temporelle, ce qui offre d'excellentes applications de l'apprentissage par renforcement, une piste que nous allons explorer lors de notre travail.

Ce mémoire est organisé comme suit :

Chapitre 1 : dans ce chapitre, nous allons donner un aperçu sur le covoiturage, ses composants et ses contraintes. Nous présentons, par la suite une liste non-exhaustive des travaux connexes.

Chapitre 2 : ce chapitre est consacré à notre contribution. Nous commençons par motiver et justifier nos choix, par la suite, nous présentons les concepts de base de l'apprentissage par renforcement et ces composants. Ensuite, nous présentons la modélisation proposée et nous détaillons chaque phase de réalisation de notre travail.

Chapitre 3 : nous présentons dans ce chapitre les détails d'implémentation de notre système. Nous discutons, par la suite, les résultats trouvés et les évaluations élaborées.

Chapitre 1 : Etat de l’art

1.1. Introduction

Le covoiturage est une solution de transport alternative à l’autosolisme qui permet d'augmenter les taux d'occupation des véhicules et par conséquent lutter contre la congestion automobile et réduire les émissions de Gaz à effet de serre. Ce mode de transport se caractérise par un aspect dynamique, un ensemble de contraintes et plusieurs considérations théoriques.
Dans ce chapitre, nous allons présenter quelques notions liées au covoiturage. Nous décrivons, par la suite, une liste de contraintes, de critères d’optimisation et de problématiques théoriques de ce système. Cet ensemble de paramètre va nous servir de critères de classification des travaux connexes.

1.2. Mobilité partagée

La mobilité partagée présente de nombreux avantages, tels que la réduction des niveaux de congestion et de pollution et la réduction des coûts de transport pour les personnes et les marchandises, mais elle présente également des défis qui freinent une adoption généralisée. Furuhata et al. (2013) ont identifié trois défis majeurs pour les agences proposant des covoiturages aux passagers. Il s'agit de : concevoir des mécanismes attrayants, un agencement de trajet approprié et instaurer la confiance parmi les passagers inconnus dans les systèmes en ligne. Ainsi, pour être adopté plus largement, un service de mobilité partagée doit être facile à mettre en place et permettre un déplacement sûr, efficace et économique. En tant que tel, il devrait être en mesure de concurrencer l'accès immédiat au transport de porte à porte offert par les voitures particulières (Agatz et al., 2012).

Le concept de mobilité partagée s'applique non seulement au transport de personnes mais aussi au transport combiné de personnes et de marchandises afin de mieux utiliser les ressources de transport disponibles. La littérature a présenté un certain nombre de variantes du problème de la mobilité partagée pour le transport de personnes et de marchandises (Abood et al., 2019).

Les systèmes de mobilité partagée pour le transport de personnes visent à minimiser le nombre de sièges vacants dans les véhicules afin de réduire le nombre de véhicules utilisés, et donc les embouteillages et la pollution. Cet objectif peut être atteint à l'aide d'un certain nombre de concepts, tels que : ***ridesharing, carpooling, vanpooling, car-sharing, dial-a-ride, etc***.

Il faut noter que le *ridesharing* et le *carpooling* sont deux concepts différents, cependant, ils sont tous les deux traduit en français par le mot covoiturage. Nous utilisons dans ce mémoire le mot covoiturage par référence au concept du *ridesharing* que nous considérons dans notre travail.

Nous présentons dans ce qui suit les différences entre ces concepts de mobilité partagée.

1.2.1. Problèmes de la mobilité partagée

1.2.1.1. Dial-a-Ride (DARP)

Le **DARP** fournit des trajets partagés entre n'importe quelle origine et destination en réponse à des demandes avancées de passagers dans une zone spécifique. Le DARP modélise un mode de transport répondant à la demande dans lequel l'objectif est de définir un ensemble d'itinéraires afin de satisfaire les demandes des passagers à des coûts minimisés (Masson et al.,2014 ; Ritzinger et al., 2016). Chaque demande consiste à transporter un passager de son lieu d'origine à son lieu de destination où les passagers ayant des préférences similaires en matière d'itinéraire et de temps peuvent partager le même véhicule tant qu'il y a de la capacité. En tant que tel, la résolution du DARP consiste à minimiser la distance totale du trajet, et donc le temps de trajet, tout en respectant les restrictions de temps spécifiées par le passager et toute contrainte de restriction du véhicule.

1.2.1.2. Covoiturage (Ridesharing)

Un problème de covoiturage appartient à un problème DARP classique. Le covoiturage permet à des personnes ayant des itinéraires et des horaires similaires de partager un véhicule pour un trajet, de telle sorte que les frais de déplacement de chacun (c'est-à-dire le carburant, le péage, les frais de stationnement, etc.) sont réduits (Furuhata et al., 2013). L'idée du covoiturage présente de nombreux avantages, notamment la réduction du coût et de la durée des déplacements, la diminution de la consommation de carburant et d'énergie, l'allègement des embouteillages et donc la réduction de la pollution atmosphérique. Il existe plusieurs variantes du problème du covoiturage, dont la plupart développent des systèmes de mobilité efficaces permettant aux voyageurs de partager leurs déplacements et d'améliorer ainsi leur expérience de voyage. La

planification des trajets en covoiturage peut être classée en deux catégories : le covoiturage pré-arrangé, ou statique où la demande des conducteurs et des passagers est connue à l'avance, et le covoiturage dynamique où les demandes des passagers et des conducteurs parviennent à la volée en temps réel (Furuhata et al., 2013).

1.2.1.3. Carpooling

Ce type de covoiturage a été introduit pour la première fois par les grandes entreprises dans le but d'encourager leurs employés à prendre leurs collègues en voiture pour se rendre au travail. L'idée était de minimiser le nombre de voitures se rendant chaque jour sur leurs sites (Baldacci et al., 2004). C'est vrai que ce concept est généralement utilisé pour les déplacements domicile-travail, mais il est devenu de plus en plus populaire pour des trajets plus longs.

1.2.1.4. Vanpooling

Dans ce problème, les passagers du covoiturage se rendent à un emplacement intermédiaire, appelé emplacement de stationnement relais, puis prennent une camionnette et se rendent ensemble à la destination cible. Le vanpooling, qui peut être exploité sur des bases quotidiennes ou à long terme (Calvo et al., 2004), fournit des moyens de transport réguliers et économiques cependant, ils ne s'adaptent pas aux changements d'horaire imprévus.

Les figures 1.1 et 1.2 montrent les relations et les différences entre les différents concepts de partage de véhicule.

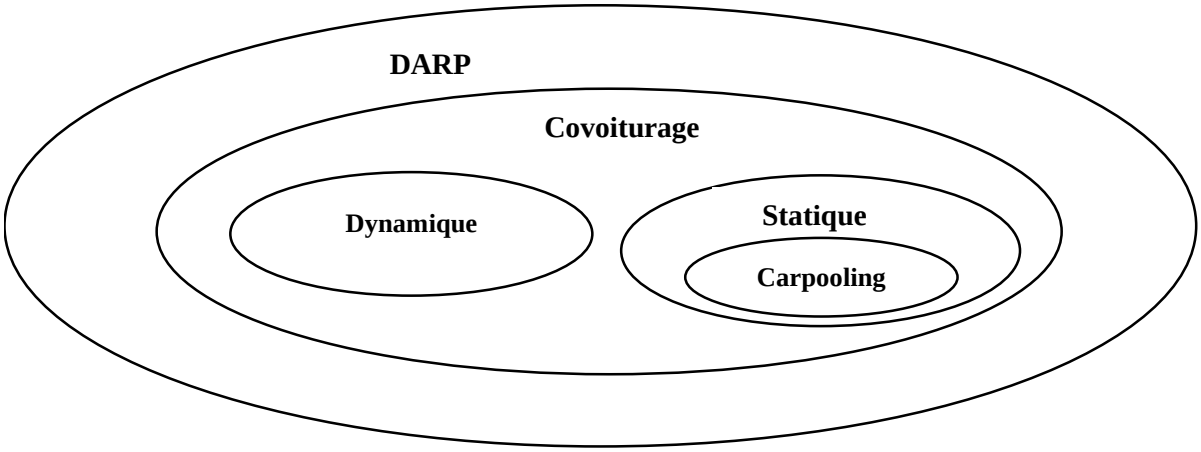


Figure 1.1 : *La relation entre les problèmes de mobilité partagée.*

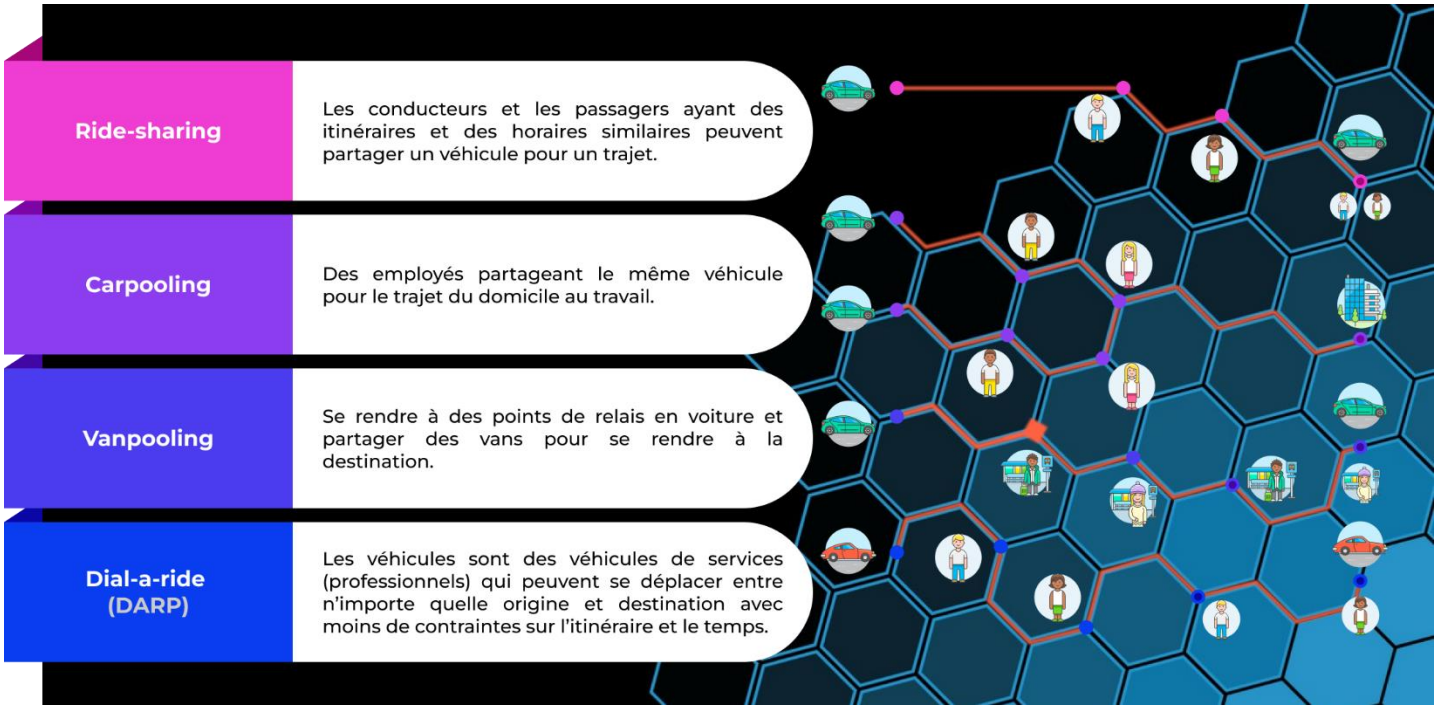


Figure 1.2 : *La différence entre les problèmes de mobilité partagée.*

Nous nous intéressons dans notre travail au concept du covoiturage (ridesharing). Nous allons donner dans ce qui suit des définitions plus détaillées de ce problème de trajet partagé ainsi que ses composants et ses variantes.

1.3. Covoiturage

1.3.1. Définition

Le covoiturage consiste en l'utilisation commune d'un véhicule par un conducteur non professionnel et un (ou plusieurs) passager(s) dans le but d'effectuer tout ou une partie d'un trajet commun. Cette nouvelle forme de transport a contribué à l'amélioration de l'image environnementale, sociale et économique de la voiture en réduisant les émissions de CO2, les dépendances à la voiture, les frais de transport engagés dans le cadre d'une mobilité individuelle, etc. Le covoiturage permet aussi de restaurer une certaine communication qui a disparu dans les

transports en commun en fortifiant les liens sociaux en regroupant des personnes dans la même voiture (Jeribi, 2012).

Furuhata et al., (2013) définissent le covoiturage comme un *« mode de transport dans lequel des voyageurs individuels partagent un véhicule pour un voyage et partagent les coûts liés au voyage tels que l'essence, les péages et les frais de stationnement avec les autres qui ont un itinéraire et un horaire similaire »*.

Cette définition du covoiturage inclut deux formes de covoiturage comme Furuhata et al. (2013) le mentionnent :

- Le covoiturage désorganisé où les participants s'organisent directement entre eux afin d'effectuer le covoiturage sans l'aide d'une tierce partie. Cette forme de covoiturage possède un long historique et implique généralement des membres de la famille, des voisins, des amis, des collègues de travail, voire même des inconnus.
- Le covoiturage organisé est opéré par des agences qui offrent des opportunités d'appariement entre des participants qui n'ont a priori aucun historique commun et qui désirent établir un groupe de covoiturage. Ce type de covoiturage permet un passage à grande échelle étant donné les participants potentiels n'appartiennent pas seulement à l'environnement direct des participants. Une partie importante de ce type de covoiturage est la présence d'arrangements préalables à l'activité de covoiturage qui sont effectués par l'agence afin de faire l'appariement entre les demandes et les offres de covoiturage (Gagnon, 2016).

La figure 1.3 montre un exemple d'un système de covoiturage où trois demandes peuvent être desservies par un seul conducteur. Ceci a pour conséquent de réduire le temps de voyage et d'alléger ainsi des embouteillages (Silwal et al., 2019).

Figure 1.3 : *Un système de covoiturage où trois demandes sont desservies par un seul conducteur de leurs sources (S) à leurs destinations (D) (Silwal et al., 2019).*

Il existe plusieurs variantes du problème du covoiturage, dont la plupart développent des systèmes de mobilité efficaces qui permettent aux voyageurs de partager leurs trajets et ainsi d'améliorer leur expérience de voyage. La planification de trajets en covoiturage peut être classée en covoiturage « statique » et en covoiturage « dynamique ».

Dans le covoiturage statique, l'offre et la demande des conducteurs et passagers, respectivement, est connue à l'avance (c'est-à-dire que les origines, les destinations, les heures de départ et d'arrivée des voyageurs sont données à l'avance) et peut ainsi être utilisée pour planifier leurs trajets partagés. Ces services préétablis sont principalement utilisés pour planifier des trajets quotidiens (domicile-travail) ainsi que des trajets longue distance partagés (par exemple, des trajets interurbains). Cependant, les trajets longue distance ont généralement des horaires plus flexibles que les trajets domicile-travail. Ceci fait appel à la deuxième variante du covoiturage, à savoir, le covoiturage dynamique qui permet d'apparier des pairs (offres, demandes) en temps réel ou quasiment.

1.3.2. Covoiturage statique

Le covoiturage statique est assuré par un système non automatisé accessible via le web, intégrant les fonctionnalités de gestion des offres et demandes de covoiturage. La majorité des systèmes existants font partie de cette catégorie. L'utilisation de tels systèmes nécessite une inscription préalable afin de pouvoir avoir accès aux différentes fonctionnalités offertes par le biais de ces supports. Malheureusement, ces fonctionnalités se résument dans le cas général à un simple dépôt ou consultation d'une offre ou demande de covoiturage selon que l'internaute soit conducteur ou passager. En effet, Il est supposé que toutes les offres et les demandes de conducteurs et de passagers sont connues à l'avance avant l'exécution d'un processus d'appariement (Cheikh, 2016).

De telles solutions statiques ont réussi pour certains types de solutions de covoiturage, mais ne fonctionnent pas bien en l'absence d'informations statiques disponibles avant la prise de décision. Le covoiturage dynamique vient apporter plus de flexibilité et de dynamisme à ce type de transport. Il présente un fort potentiel de développement du fait de la souplesse de service qu'il vise à apporter grâce à ses grands principes : le temps réel, l'optimisation du temps et des trajets et la garantie d'un service fiable.

1.3.3. Covoiturage dynamique

Le covoiturage dynamique se caractérise par une grande souplesse d'utilisation et moins d'interdépendance que le covoiturage statique. Il s'agit de pouvoir trouver un trajet partagé dans un délai rapide (moins d'une demi-heure par exemple) pour un trajet donné et en fonction de la position des véhicules des conducteurs potentiels. L'accès au service et son mode de fonctionnement doivent être faciles et souples.

C'est un processus automatisé dans lequel un fournisseur de services met en relation des conducteurs et des passagers ayant des itinéraires et des horaires similaires pour partager un trajet à court préavis dans un véhicule personnel, les systèmes de covoiturage sont naturellement dynamiques (Prieto et al., 2017). Leur complexité repose principalement sur l'appariement d'individus soumis à des contraintes spatio-temporelles, qui doivent être précisées de part et d'autre – c'est-à-dire conducteurs et passagers – avant que le trajet souhaité ne soit établi et exécuté. D'une part, les passagers demandent un trajet à une heure précise, d'une origine spécifique à une destination spécifique. En revanche, les conducteurs ont une trajectoire et une heure de départ fixes. Par conséquent, les systèmes de covoiturage nécessitent une certaine flexibilité puisque des

déviations peuvent être nécessaires à différents points de la trajectoire pour prendre et déposer des passagers, tant que ces distances de détour ne dépassent pas la distance de tolérance du conducteur (Cici et al., 2015).

Un scénario classique d'un service de covoiturage dynamique peut être décrit comme suit :

1- Le conducteur informe le service via une application de son Smartphone qu'il offre un trajet en tant que conducteur ; il indique son heure et son point de départ, sa destination ainsi que le nombre de passagers qu'il peut accueillir dans son véhicule ;
2- Quelques minutes avant son départ, le passager qui demande à effectuer un itinéraire en covoiturage contacte le service via la même application pour indiquer sa demande ;
3- Le service cherche alors un conducteur présent sur l'itinéraire demandé et qui respecte les contraints temporelles. Une fois trouvé, le service met en relation conducteur et passager. Il ne reste plus au conducteur qu'à finaliser l'itinéraire en déterminant le point de rendez-vous et en utilisant le géo positionnement par satellite (GPS) de son téléphone ;

Néanmoins, malgré les progrès technologiques et les efforts déployés dans ce sens, la plupart des systèmes existants implémentant un service de covoiturage en temps réel, sont restés au stade embryonnaire due principalement au manque de sécurité, et à la faille d'automatisation. En outre, comparés aux systèmes statiques, les plateformes dédiées au covoiturage dynamique, présentent l'avantage de consulter en temps réel la liste des offres des véhicules en circulation. Par contre, l'aspect optimisation est complétement ignoré. En effet, ces systèmes n'intègrent pas des algorithmes de recherche pour générer des affectations Véhicule/Requête, optimisant certains critères tels que le coût et la durée du voyage et respectant quelques contraintes telles que la capacité des véhicules et la synchronisation des transferts.

1.3.3.1. Composants d'un système de covoiturage dynamique

Un système de covoiturage est constitué d'un ensemble d'éléments relatifs à sa réalisation.

- Demande : il s'agit d'une requête de déplacement émise par un passager exprimant la volonté de se déplacer en voiture d'un point source (origine ; point de départ) à un point destination.
- Offre : elle exprime la volonté d'un conducteur de partager son trajet dans son propre véhicule avec d'autres personnes. Cette offre doit être accompagnée d'un ensemble de

paramètres spécifiant l''origine, la destination, le nombre de sièges vacants, heure d'arrivée, heure de départ, etc.

- Conducteurs : ce sont les propriétaires de véhicules proposant des offres de trajets.
- Passagers : ce sont les personnes qui ont émis une demande de covoiturage pour être transporter vers une destination précise.
- Contraintes d'appariement : faire apparier une demande à une offre est conditionnée par un ensemble de contraintes qui doivent être satisfaites. Ces contraintes peuvent concerner : l'emplacement (trajet identique, trajet du passager inclus dans le trajet du conducteur, trajets partiellement identiques, etc.) ; le temps (les heures de départ et d'arrivée doivent se coïncider) ; respect du nombre de sièges vacants ; etc.

La figure 1.4 décrit les composants d'un système de covoiturage dynamique ainsi que les interactions entre ces composants.

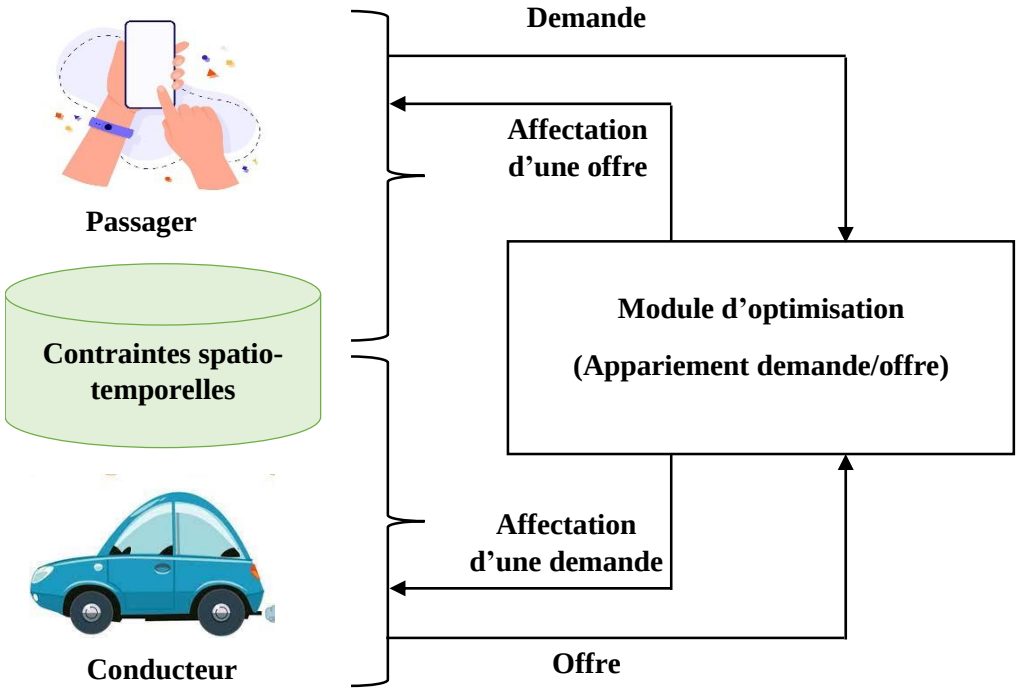


Figure 1.4 : *Les composants d'un système covoiturage.*

1.3.3.2. Contraintes sur le covoiturage dynamique

Dans un problème de covoiturage dynamique, l'objectif est de satisfaire un ensemble de demandes de transport (partage de trajets) tout en respectant un ensemble de contraintes. Ces contraintes peuvent être dérivées des caractéristiques du problème tel que le temps, l'emplacement, le coût, l'itinéraire, …, comme elles peuvent être dérivées des préférences et au confort social du passager et/ou du conducteur tel que l'âge, le sexe, …

Chaque contrainte peut simplifier les calculs et permettre de diminuer les espaces de recherche, mais rend également l'appariement entre conducteur et passager plus difficile et affecte les résultats.

Nous présentons dans cette section quelques contraintes et caractéristiques du covoiturage dynamique qui, à partir desquelles découlent plusieurs instances de ce problème et qui vont nous permettre, par la suite, de classer les travaux de la littérature de ce domaine.

Nous ne considérons pas toutes ces contraintes dans notre travail, mais il est important de mentionner que si un algorithme de covoiturage doit être fourni, il y a un grand nombre de contraintes qui peuvent être prises en compte.

- ***Contrainte de temps (CT)***

  En plus d'indiquer l'origine et la destination d'une demande de covoiturage, un trajet partagé doit également indiquer quand ce processus peut avoir lieu. Cela se fait généralement en associant une fenêtre temporelle à chaque demande de covoiturage. Cette fenêtre temporelle est généralement indiquée par le passager où il spécifie sa date de départ au plus tôt et sa date d'arrivée au plus tard. Ainsi, pour qu'un passager puisse participer à un trajet partagé, il doit être pris en charge à son origine et déposé à sa destination dans la fenêtre temporelle qu'il a spécifiée (Stiglic, et al. 2016 ; Mourad et al. 2019).

  De son côté, le conducteur aussi peut spécifier une fenêtre temporelle indiquant son temps de départ et d'arrivée. Dans ce cas, répondre à une requête de covoiturage ne dépendra pas seulement des origines et destinations des passagers et des conducteurs mais aussi de leurs fenêtres temporelles.

- ***Contrainte de capacité (CC)***

  La contrainte de capacité concerne le nombre de passagers qui sont permis de partager le même véhicule au même moment. Ceci est traduit par le nombre de sièges vacants dans ce véhicule précisé par le conducteur (Santos et Xavier. 2015). En plus de limiter le nombre de passagers qui peuvent partager le même trajet, le passager, à sa demande peut indiquer le nombre de personnes qui vont l'accompagner dans son voyage.

- ***Contrainte de coût (CO)***

  Les gens peuvent choisir de participer au covoiturage principalement pour des économies potentielles ; les dépenses liées au voyage, telles que le carburant et les péages, sont partagées. De plus, afin d'attirer plus de participants, les coûts de voyage dans les systèmes de covoiturage devraient être compétitifs par rapport aux autres modes de transport.

  Dans certains cas, un participant à un système de covoiturage peut spécifier un coût maximum qu'il est prêt à payer pour le trajet partagé, et devrait donc être apparié à des trajets partagés qui restent sous le montant maximum spécifié (Mourad et al. 2019).

- ***Contrainte de partage (CP)***

  Les conducteurs proposant un trajet peuvent vouloir prendre un seul passager ou accepter de prendre plusieurs passagers. De même, les passagers qui demandent un trajet peuvent vouloir le partager avec un seul conducteur ou peuvent être disposés à le partager avec plusieurs conducteurs. Cela veut dire qu'une demande peut être affectée à plusieurs véhicules afin d'atteindre sa destination finale où chaque conducteur va intervenir sur seulement une partie du trajet du passager. Ce type de covoiturage est appelé « sauts multiples, *Multi-Hop* ».

Ainsi, nous pouvons distinguer quatre variantes d'un système de covoiturage de base, comme indiqué dans le tableau 1.1.

Tableau 1.1 : *Les différentes variantes d'un système de covoiturage liées à la contrainte de partage.*

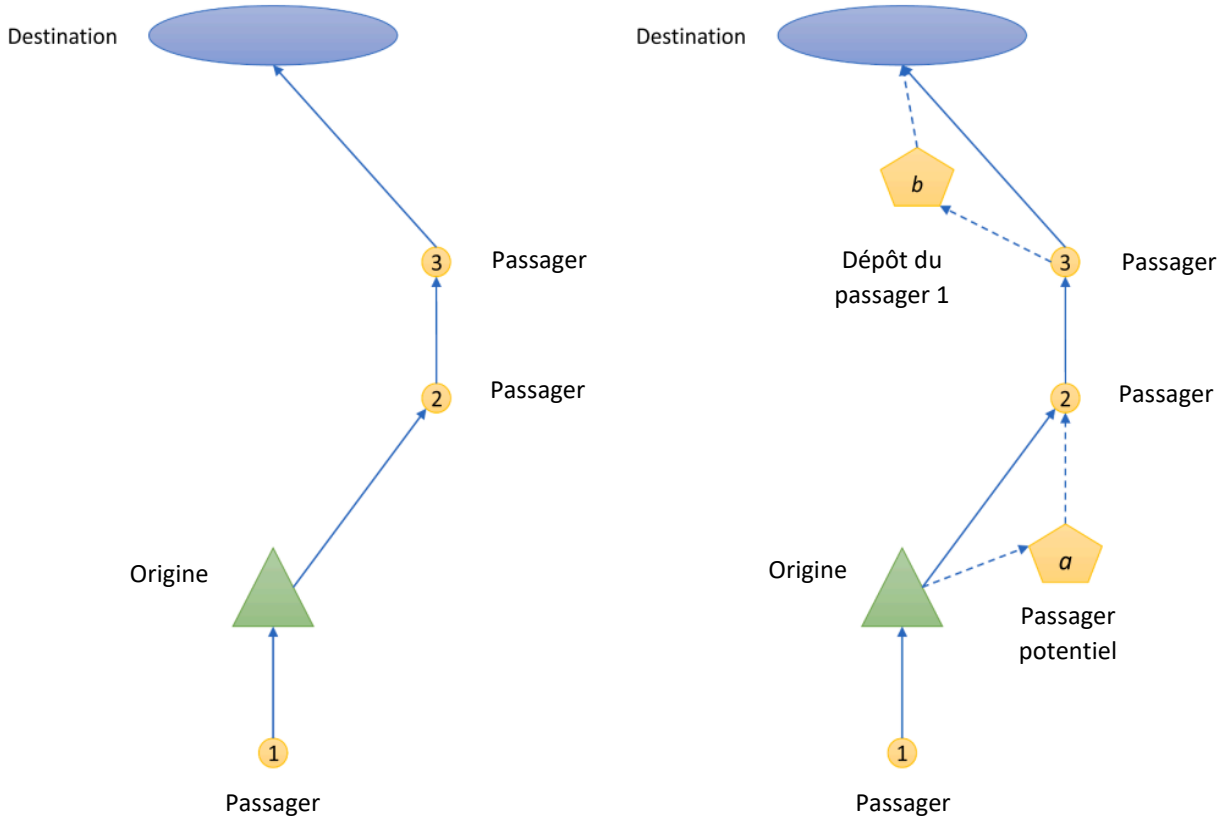
	Passager unique	**Plusieurs passagers**
Conducteur unique	Problème d'appariement des couples (conducteur, passager). Nous appelons cette variante : **1C→1P**	Le conducteur peut répondre à plusieurs demandes. Au problème d'appariement s'ajoute un problème de routage. Nous appelons cette variante : **1C→NP**
Plusieurs conducteur	Un seul passager va être transférer entre un conducteur à un autre afin d'atteindre sa destination. Au problème d'appariement s'ajoute un problème de routage. Nous appelons cette variante : **NC→1P**	Cette variante combine (1C→NP et NC→1P) où plusieurs conducteurs vont servir plusieurs passagers en partageant le même véhicule et acceptant des sauts d'un véhicule à un autre. Au problème d'appariement s'ajoute un problème de routage. Nous appelons cette variante : **NC→NP**

❖ ***Contrainte de détour (CD)***

Dans certains systèmes de covoiturage, les déviations de route sont autorisées afin de récupérer et/ou déposer des passagers. Ceci est toléré dans le but de desservir un maximum de demandes tout en limitant le nombre de voitures. La figure 1. 5 montre deux scénarios de covoiturage, le premier sans détour et le deuxième avec détour (Martins et al., 2021).

La considération de cette contrainte est liée au respect de la distance et du temps estimés pour les trajets. En effet, Un trajet peut être associé à une voiture, uniquement si un certain taux de

détour de tous les trajets, y compris le nouveau trajet reçu et les trajets déjà attribués à la voiture, est satisfait (Shen et al., 2016).



(a)Trajet du conducteur sans détours pour récupérer et déposer des passagers.

(b)Trajet du conducteur avec possibilité de détours pour récupérer et déposer des passagers.

Figure 1.5 : *Une représentation de deux scénarios de covoiturage : sans détours et avec détours (Martins et al., 2021).*

❖ ***Temps d'attente (TA)***

Cette contrainte interprète la flexibilité en termes de temps des conducteurs et des passagers. En plus de la fenêtre temporelle qu'ils peuvent spécifier, chacun de ces acteurs peut spécifier un temps maximal d'attente. Du côté du passager, cette marge temporelle indique qu'il est prêt de démarrer après sa date de départ au plus tôt. Du côté du conducteur, ce temps d'attente est lié aux potentiels détours qu'il peut effectuer afin de récupérer certains passagers.

❖ ***Contraintes de confort social (CS)***

Bien que les contraintes suscitées aient une influence directe sur la qualité et la performance d'un système de covoiturage, le confort social des passagers et des conducteurs reste un facteur important dans la réussite de tels systèmes.

La façon dont les utilisateurs perçoivent les dimensions sociales du partage du temps et de l'espace avec des étrangers n'est toujours pas claire. Certains passagers peuvent apprécier positivement la possibilité d'interagir avec de nouvelles personnes, tandis que d'autres peuvent considérer ces interactions comme gênantes, dangereuses ou même comme une expérience au cours de laquelle ils sont victimes de discrimination de la part des autres passagers (Sarriera et al., 2017).

Du point de vue des passagers, leur sentiment de satisfaction est principalement déterminé par le confort social avec le conducteur et les autres passagers (par exemple, le sexe, l'intérêt ou le fait d'avoir des amis communs). En outre, le confort social a tendance à augmenter si deux personnes partagent un intérêt commun, par exemple, un amateur de musique rock peut préférer partager un trajet avec quelqu'un qui a un goût similaire. De plus, le souci de sécurité est également un facteur important pour le covoiturage. Par exemple, une conductrice peut souhaiter prendre le véhicule avec une conductrice pour des raisons de sécurité tard dans la nuit.

Le conducteur, de son côté, joue un rôle important afin d'apporter un certain degré de confiance et de sécurité pour les passagers. Ce dernier peut même exiger certaines restrictions lors de la proposition du trajet à partager. Ces restrictions peuvent concerner l'âge, le sexe, l'interdiction de fumer, l'interdiction d'animaux domestiques, … Par conséquent, une tendance claire pour les systèmes de covoiturage est de fournir des services en tenant compte à la fois des facteurs sociaux et des problèmes de confiance, ainsi que de maximiser le nombre de demande desservies (Fu et al., 2019).

1.3.3.3. Critères d'optimisation

La plupart des objectifs que les problèmes de covoiturage dynamique visent à optimiser peuvent être classés en deux grandes catégories : des **objectifs opérationnels** et des **objectifs liés à la**

qualité. Les objectifs opérationnels visent généralement à optimiser les coûts d'exploitation de l'ensemble du système, par exemple en minimisant les kilométrages des véhicules et le temps de transport, en maximisant le nombre de demandes satisfaites, en minimisant le nombre de véhicules nécessaires, etc. Les objectifs liés à la qualité visent à améliorer la qualité du service fourni. Par exemple, la minimisation du temps total de déplacement ou d'attente des passagers peut donner lieu à de meilleures performances du point de vue des passagers.

Nous donnons ci-dessous, une liste non-exhaustive des objectifs qu'on peut optimiser dans un système de covoiturage dynamique.

1.3.3.3.1. Objectifs opérationnels

L'ensemble des objectifs opérationnels a été défini par (Agatz et al., 2012) et ils sont relatifs à la distance, au temps et au nombre de participants.

- ***Minimiser la distance parcourue (Min_D)***

  Cette mesure représente le nombre total de kilométrages parcourus par tous les participants se rendant à leur destination. Cet objectif est important d'un point de vue social puisqu'il contribue à réduire la pollution et la congestion.

- ***Minimiser le temps de voyage (Min_T)***

  Cette mesure représente le temps passé dans le véhicule pendant le trajet entre l'origine (la source) et la destination. Ce critère est important d'un point de vue environnemental et sociétal puisqu'il est lié à l'émission des véhicules de gaz à effet de serre qui ne se rapporte pas seulement aux distances mais aussi à la vitesse des véhicules.

- ***Maximiser le nombre de participants (Max_P)***

  Cet objectif maximise le nombre de conducteurs et de passagers satisfaits dans le système. Il peut être avantageux pour un fournisseur privé de covoiturage dont les revenus sont liés au nombre d'ententes de covoiturage réussies. De plus, le taux de réussite correspondant peut également être un indicateur de performance important pour les utilisateurs d'un service de covoiturage particulier, et un taux de réussite élevé peut stimuler un plus grand nombre de participants à l'avenir.

1.3.3.3.2. Objectifs de qualité

- ***Minimiser le temps d'attente (Min_TA)***

  Un critère important lié à la qualité d'un système de covoiturage est le temps d'attente d'un passager avant d'être récupérer par un conducteur. Optimiser cette métrique a pour objectif de satisfaire les passagers desservis et ainsi améliorer le service.

- ***Minimiser le temps de réponse (Min_TR)***

  Par temps de réponse, on entend le temps dépensé afin de trouver des appariements entre les conducteurs et les passagers. Ceci peut aussi être capturé, aussi, par un taux de réponse qui est le rapport entre les demandes appariées et l'ensemble des demandes de déplacement. Etant donné que la probabilité d'annulation avant l'appariement est principalement fonction du temps de réponse (temps d'attente avant l'appariement), le temps de réponse total est une mesure alternative au temps de réponse (Qin et al., 2021).

- ***Maximiser le nombre de demandes desservies (Max_ND)***

  Un indicateur important pour la réussite d'un système de covoiturage est le nombre de demandes de trajets accomplies avec succès. Ce critère indique que le processus d'appariement est fonctionnel et répond bien aux contraintes des deux acteurs. Ceci peut aussi être capturé par un taux de satisfaction qui est le rapport entre les demandes terminées (desservies/accomplies) et toutes les demandes. Le taux de satisfaction n'est pas supérieur au taux de réponse. L'écart est dû à l'annulation post-appariement, généralement, en raison de l'attente de la récupération. Par conséquent, la distance moyenne de récupération est également une quantité pertinente à observer (Qin et al., 2021).

De plus, certains objectifs plus spécifiques au problème ont été considérés dans la littérature, tels que ; la minimisation du nombre de véhicules requis (Al-Abbasi et al., 2019), la maximisation du taux d'occupation des passagers (Armant & Brown, 2020), la maximisation de la fiabilité du système (Huang et al., 2014), la minimisation du temps d'appariement (Yang et al., 2020), la minimisation du nombre de véhicules nécessaires au fonctionnement du système et la minimisation du temps d'attente total des demandes avant d'être desservies (Singh et al., 2021).

Une grande partie de la recherche sur le covoiturage dynamique est axée sur l'optimisation d'un seul objectif, mais certains travaux considèrent des systèmes à objectifs multiples combinant des

objectifs opérationnels et des objectifs liés à la qualité. Dans les systèmes à objectif unique, les considérations relatives à la qualité du service sont représentées comme des contraintes dans le modèle afin de garantir un niveau de service minimum (Molenbruch et al., 2017).

1.3.3.4. Problèmes théoriques

Le covoiturage dynamique a reçu beaucoup d'intérêt ces dernières années. Cet intérêt est motivé par le fait que, de nos jours, on a besoin de nouveaux modes de transport qui facilitent le déplacement mais qui, en même temps, respectent les contraintes environnementale et sociétale. Dans ce cadre, le covoiturage dynamique attire de plus en plus l'attention des chercheurs dans le monde entier surtout que beaucoup de considérations théoriques découlent de ce type de covoiturage. Parmi ces considérations théoriques figurent la confiance et la sécurité, le routage, la tarification et l'appariement dynamique.

- ***Confiance et sécurité (C)***

  Les covoiturages sont généralement coordonnés entre des participants qui ne se connaissent pas. Cela pose le problème de la sécurité et de la confiance. Afin d'avoir un covoiturage confortable, les participants doivent se sentir en sécurité avec des étrangers et ils doivent se faire confiance. Parmi les approches pour améliorer la confiance et la sécurité, il y a l'utilisation du système de réputation (comme celui utilisé sur eBay), la limitation du covoiturage à la communauté proche et l'utilisation des informations fournies par les réseaux sociaux (Herbawi, 2013 ; Stach & Brodt, 2011).

- ***Routage (R)***

  Le covoiturage dynamique est une généralisation du problème de routage de véhicules (VRP). En effet, au problème de base d'un système de covoiturage dynamique qui est l'appariement des conducteurs et des passagers, s'ajoute un problème de routage qui fournit la navigation au conducteur avant d'affecter les passagers à ces véhicules. Ce routage peut être appliqué à plusieurs niveaux selon la variante de covoiturage considérée. Si le problème considéré est 1C→1P, alors le routage va déterminer les itinéraires efficaces. Si le problème considéré est 1C→NP, alors, l'objectif de routage devient à trouver les itinéraires à suivre afin de récupérer et déposer les passagers. Quand il s'agit de plusieurs conducteurs et plusieurs passagers, le routage implique aussi le transfert entre les conducteurs (Qin et al., 2021).

- ***La tarification (T)***

  Lorsqu'un passager potentiel soumet une demande de voyage, le module de tarification propose un devis, que le passager accepte ou refuse.

  Étant donné que le tarif du trajet détermine non seulement le prix que le passager doit payer pour le trajet, mais également le revenu du conducteur, les décisions de tarification influencent à la fois la distribution de l'offre et de la demande en fonction de la sensibilité des utilisateurs au prix, par exemple, l'utilisation de la tarification majorée pendant les heures de pointe. Le problème de tarification dans la littérature sur le covoiturage est dans la plupart des cas une tarification dynamique, qui ajuste les prix des trajets en temps réel compte tenu de la l'évolution de la demande et de l'offre (Qin et al., 2021).

- ***Appariement dynamique (A)***

  La mise en correspondance automatisée des conducteurs et des passagers qui nécessite un minimum d'effort de la part des participants est au cœur de tout système de covoiturage dynamique. Le problème d'appariement du covoiturage (Yan et al., 2020 ; Özkan & Ward, 2020 ; Qin et al., 2021) peut apparaître sous différents noms dans la littérature, par exemple, répartition des commandes, affectation des commandes aux conducteurs (*order dispatching, order-driver assignment*). Il s'agit d'un problème d'appariement bipartite en ligne où l'offre et la demande sont dynamiques, l'incertitude provenant des arrivées de la demande, des temps de trajet et du comportement d'entrée-sortie des conducteurs. L'appariement peut être effectué en continu ou à des fenêtres fixes (c'est-à-dire par lots, *fixed review windows* ou *batching*).

  L'appariement des demandes en ligne n'est pas entièrement unique au covoiturage. En effet, l'appariement du covoiturage entre dans la famille des problèmes d'appariement dynamique plus généraux pour des offres – demandes des marchés (Hu & Zhou, 2020). Un trait distinctif du problème du covoiturage est sa nature spatio-temporelle. L'éligibilité d'un conducteur à correspondre et à répondre à une demande de déplacement dépend en partie de sa proximité spatiale avec la demande. Les demandes de trajet prennent généralement un temps différent pour se terminer, et elles modifient les états spatiaux des conducteurs, affectant la distribution de l'offre pour une correspondance future. Les conducteurs et les passagers présentent généralement des comportements de sortie asymétriques dans la mesure où les conducteurs restent généralement dans le système pendant une période de

temps prolongée, alors que les demandes des passagers sont perdues après une période d'attente beaucoup plus courte en général.

1.3.3.5. Techniques de résolution

Comme mentionné ci-dessus, le problème de mobilité partagée est une généralisation du problème de routage de véhicules (VRP) qui est NP-difficile, en général. En outre, les variantes simplifiées du problème (par exemple, avec un seul conducteur, un seul passager, une seule récupération et un seul dépôt ou une variante à objectif unique) sont toujours NP-difficile (Gu, Liang et Zhang. 2016). La résolution de ces problèmes devient plus complexe lorsqu'ils comportent des paramètres dynamiques et des données d'entrée stochastiques. Ainsi, des approches de solutions exactes et heuristiques ont été introduites dans la littérature.

Un système de covoiturage en temps réel vise à réunir des voyageurs dans un délai très court. Ainsi, il peut avoir besoin d'être réoptimisé à intervalles réguliers, à mesure que de nouveaux voyageurs entrent ou sortent du système. Par conséquent, les voyageurs déjà en route doivent être informés de tout changement de plan à chaque fois que le système est réoptimisé, car leurs itinéraires originaux peuvent être modifiés. Ce processus automatisé nécessite des modèles et des algorithmes efficaces pour mettre en relation les conducteurs et les usagers dans des temps de calcul très courts.

De nombreuses études récentes sur les systèmes de covoiturage en temps réel se sont concentrées sur le développement d'approches heuristiques, car elles peuvent fournir des solutions de bonne qualité dans des temps de calcul relativement courts. Néanmoins, de tels systèmes peuvent également être traités par des algorithmes d'énumération (exacts) (comme le branch-and-bound) (Mourad et al., 2019).

Nous passons ici en revue quelques travaux qui ont traité le covoiturage dynamique, les études les plus récentes et leurs approches de solution. Nous présentons principalement les travaux qui ont utilisés les métaheuristiques (méthodes approchées), les méthodes exactes et l’apprentissage par renforcement, car ces techniques ont été largement utilisées dans ce cadre.

❖ ***Les métaheuristiques (MH)***

Ces techniques ont été largement utilisées dans le cadre du covoiturage dynamique. (Manna & Prestwick, 2014) ont proposé un algorithme génétique et une heuristique d'insertion qui modifie la solution de l'algorithme génétique pour résoudre le problème du taxi partagé, afin de maximiser le profit attendu pour le conducteur. Herbawi & Weber (2012) ont proposé un algorithme génétique pour résoudre DARP avec sauts et sous contraintes temporelles, l'objectif était de minimiser la distance totale et la durée totale du déplacement, et de maximiser le nombre de demandes desservies.

Ben Cheikh et al. (2015) ont proposé un algorithme coopératif évolutif basés sur les agents pour résoudre le problème du covoiturage dynamique avec sauts afin de minimiser le temps d'attente du conducteur et du passager, le temps de retard et le temps total de déplacement du véhicule. Dans un autre travail (Ben Cheikh et al., 2017) ont proposé une approche basée sur des opérateurs génétiques contrôlés (MACGeO) pour le même problème et le même objectif. En conséquence, le modèle satisfait 100 % des demandes en utilisant seulement 46 % des véhicules disponibles. Ben Cheikh & Hammadi (2016) ont proposé une autre approche basée sur l'algorithme évolutionnaire avec un système multi-agents (E2AIA). En conséquence, E2AIA est capable de fournir de meilleurs résultats en matière d'appariement avec un score plus faible que l'approche MACGeO.

Pour résoudre le problème de covoiturage dynamique avec détour, (Ma et al., 2018) ont créé un modèle basé sur les algorithmes génétiques ayant comme objectif de minimiser la distance de voyage du véhicule. Comme résultat de cette approche, le revenu du véhicule est augmenté d'environ 36.7%. L'optimisation par colonies de fourmis (ACO) a aussi été utilisée dans le cadre du covoiturage dynamique, l'objectif était de maximiser le nombre total de passagers appariés et les taux d'utilisation des sièges (Huang et al., 2018).

Cheikh-Graiet et al., (2020) ont proposé une nouvelle métaheuristique basée sur la recherche tabou pour résoudre le problème de covoiturage dynamique multicritères où le temps de voyage, le coût et l'émission du CO2 doivent être minimiser. L'algorithme proposé utilise un système de mémoire explicite et plusieurs stratégies de recherche originales développées pour prendre automatiquement des décisions optimales. Afin d'augmenter la satisfaction des utilisateurs, l'approche proposée permet de gérer les sauts multiples ainsi que de pouvoir déposer le passager dans un endroit proche de sa destination ou à un point de relais.

D'autres heuristiques et métaheuristiques ont été largement utiliser pour résoudre différentes variantes du problème, à l'image du recuit simulé (Jung et al, 2016), algorithme d'abeilles (Masmoudi et al., 2020), GRASP (Greedy Randomized adaptive search procedure), …

- ***Méthodes exactes (ME)***

(Armant & Kenneth, 2014) ont proposé une approche basée sur les techniques de linéarisation et de brisure de symétrie pour résoudre le problème covoiturage dynamique avec un problème de changement de rôle, afin de minimiser la distance totale du déplacement, Par la suite, les même auteurs (Armant & Kenneth, 2020) ont proposé une approche basée sur le graphe pour résoudre le problème DARP avec aussi le problème de changement de rôle, afin de minimiser la distance totale du déplacement et de maximiser le nombre de demandes desservies.

Huang et al., (2013) ont proposé un algorithme branch-and-bound pour résoudre le problème de covoiturage en temps réel. Ils ont introduit un algorithme d'arbre cinétique pour planifier les requêtes dynamiques et ajuster les itinéraires à la volée. Liu et al., (2015) ont proposé un algorithme branch-and-cut pour résoudre un DARP réaliste avec plusieurs trajets et types de requêtes et une flotte hétérogène de véhicules.

Wang et al., (2018) ont, tout d'abord, proposée une méthode exacte pour résoudre de petites instances du problème. Ensuite, ils ont présenté une heuristique de recherche tabou pour les problèmes. Pour comparer la méthode heuristique à la situation optimale, ils utilisent un ratio : la valeur objective de la solution optimale divisée par la valeur objective de sortie de la méthode heuristique et afin d'évaluer les bénéfices d'un appariement, ils calculent les économies de coût par rapport au fait de voyager seul.

(Alisoltani et al., 2019) ont fixé comme objectif de trouver une solution optimale globale pour le problème de covoiturage dynamique sans incertitude. Dans ce problème, un système de voitures fonctionne pour servir les passagers qui souhaitent partager leur voyage avec d'autres passagers. L'algorithme proposé est basé sur *integar linear programming* et résout un problème d'optimisation contraint pour minimiser le temps de trajet total et la distance pour les véhicules et le temps de trajet total et le temps d'attente pour les passagers. Les contraintes considérées dans le problème portent sur la capacité, la fenêtre temporelle, le nombre de partage (un nombre défini par les passagers pour montrer leur volonté de partager leur trajet) et la qualité du service.

❖ ***Apprentissage par renforcement (AR)***

Des approches basées sur l'AR ont été développées pour la tarification dynamique ainsi que l'appariement dynamique. En termes de formulation MDP (Markov Decision Process), la modélisation de l'agent en tant que conducteur est un choix pratique pour sa définition simple de l'état, de l'action et de la récompense, contrairement à la modélisation au niveau du système où l'espace d'action est exponentiel. Dans ce cas, la plateforme de covoiturage est naturellement un système multi-agent avec un objectif global. Une approche commune consiste à utiliser les trajectoires d'expérience de tous les conducteurs pour former un agent unique et l'appliquer à tous les conducteurs pour générer leurs politiques de correspondance (Xu et al., 2018 ; Wang et al., 2018 ; Tang et al., 2019).

(Chen et al., 2019) intègrent des bandits contextuels et le réseau de valeurs spatio-temporelles développé dans (Tang et al., 2019) pour l'appariement afin d'optimiser conjointement les décisions de tarification et d'appariement. En particulier, les actions de tarification sont les changements de pourcentage de prix discrétisés et sont sélectionnées par un algorithme de bandits contextuel, où les valeurs à long terme apprises par le réseau de valeur sont incorporées dans les récompenses des bandits. Dans (Turan et al., 2020), l'agent AR détermine à la fois le prix pour chaque couple origine-destination et les décisions de repositionnement/récupération pour chaque véhicule. L'état contient des informations globales telles que le prix dans chaque zone, la longueur de la file d'attente des passagers pour la paire origine-destination, le nombre de véhicules dans chaque zone et leurs niveaux d'énergie. La récompense tient compte des revenus du voyage, des pénalités pour les files d'attente et des coûts opérationnels pour la recharge et le repositionnement.

L'objectif de (Jindal et al., 2018) était de maximiser l'efficacité du transport afin que moins de voitures soient nécessaires pour répondre à la demande de déplacement donnée. Pour ce faire, ils ont développé un réseau de neurone spatio-temporel (deep ST-NN : Spatio-Temporal Neural Network) afin de prédire le temps du trajet à partir des données GPS. Le réseau construit apprend le temps de trajet et la distance à partir des coordonnées GPS d'une origine et d'une destination et le temps du jour. Ils ont, par la suite, développé un simulateur pour l'apprentissage par renforcement et en utilisant les sorties du ST-NN. L'objectif est de construire une politique qui indique au conducteur quand accepter une demande de covoiturage afin de maximiser l'efficacité du transport à long terme et de réduire la congestion routière.

La récompense reçue par l'agent consiste à la distance effective parcourue par le conducteur lors d'un trajet en covoiturage. Par conséquent, plus la distance parcourue par un conducteur au cours d'un trajet (pour satisfaire une plus grande demande de trajet) est grande, plus l'efficacité est élevée et moins il y aura de congestion.

Outre les décisions d'appariement conducteur-passager, des recherches ont également été menées sur l'utilisation de l'AR pour apprendre quand apparier une demande (ou un lot de demandes). Cela peut être fait du point de vue de la demande elle-même (Jintao et al., 2020), un agent est modélisé comme une demande de trajet. Un réseau d'agents est formé de manière centralisée en utilisant l'expérience commune de tous les agents pour décider s'il faut ou non reporter la mise en correspondance d'une demande à la prochaine fenêtre d'examen, et tous les agents partagent la même politique. Pour encourager la coopération entre les agents, une fonction de récompense de forme spéciale est utilisée pour tenir compte de la rétroaction de la récompense locale et globale.

L'utilisation de l'apprentissage par renforcement pour résoudre les différentes considérations théoriques du covoiturage temps réel est un axe très prometteur et très actifs. Beaucoup d'autres recherches récentes ont été réalisées dans ce cadre et que nous n'avons pas analyser dans ce manuscrit tels (song et al., 2020 ; Haliem et al., 2020 ; Qin et al., 2020 ; Lu et al., 2020 ; …).

Un ensemble de constatations qu'on puisse faire sur l'ensemble de travaux qui ont résout le problème en utilisant l'apprentissage par renforcement est que la majorité de travaux considère le conducteur comme agent. Les informations capturées dans les états concernent, généralement, l'emplacement, l'heure de départ et d'arrivée, et les contraintes exigées tels le nombre de sièges vacants, … Quant à l'observation reçue de l'environnement, la majorité de travaux, considère les coordonnées de l'emplacement du conducteur et du passager ainsi que le temps ce qui a comme conséquence une augmentation de l'espace d'états. La récompense est, en général, une fonction du critère à optimiser car son rôle est d'aider l'agent à prendre les bonnes décisions et éviter les mauvaises.

1.3.4. Classification des travaux

Nous allons classer dans cette section certains des travaux analysés dans la section 1.3.3.5 et ce en basant sur l'ensemble de contraintes, critères d'optimisation, problèmes théoriques et les techniques de résolution, présentés dans les sections précédentes.

Référence	Contraintes							Objectifs						Problématique	Solution
	CT	CC	CO	CP	CD	TA	CS	Min_D	Min_T	Max_P	Min_TA	Min_TR	Max_ND		
(Herbawi & Weber 2012)	✔	✔		✔	✔	✔		✔	✔			✔	✔	R, A	MH
(Huang et al., 2013)	✔				✔			✔	✔					A, T	ME
(Armant & Kenneth, 2014)	✔	✔			✔			✔						A, R	ME
(Manna & Prestwick, 2014)			✔		✔			✔	✔					A, T	MH
(Ben Cheikh et al. 2015)	✔	✔		✔	✔	✔		✔	✔	✔	✔	✔		R, A	MH
Liu et al., (2015)	✔	✔			✔				✔					R	ME
(Ben Cheikh & Hammadi 2016)	✔	✔		✔	✔	✔		✔	✔	✔	✔	✔		R, A	MH
(Ben Cheikh et al., 2017)	✔	✔		✔	✔	✔		✔	✔	✔	✔	✔		R, A	MH
(Filcek, et al. 2017)	✔	✔			✔		✔	✔	✔					C, T, A	MH
(Huang et al., 2018)	✔	✔			✔			✔		✔			✔	A, R	MH
(Jindal et al., 2018)	✔				✔			✔	✔				✔	R	AR
(Ma et al., 2018)	✔				✔			✔						T, R	MH
(Xu et al.,2018)	✔		✔		✔			✔					✔	R, T	AR
(Wang et al., 2018)	✔		✔		✔			✔	✔				✔	R, T	ME
(Al-Abbasi, et al.,2019)	✔	✔		✔	✔	✔		✔	✔			✔		R	AR
(Alisoltani et al., 2019)	✔	✔		✔			✔	✔	✔		✔			C, A	ME
(Chen et al., 2019)	✔		✔		✔			✔						A, T, R	AR
(Jin et al. 2019)	✔		✔		✔			✔				✔		R, A, T	AR
(Tang et al., 2019)	✔	✔			✔	✔		✔			✔			A, R, T	AR
(Singh, et al. 2019)	✔	✔		✔	✔	✔		✔	✔			✔		A, R	AR
(Zhou et al. 2019)	✔				✔			✔				✔		R, A, T	AR
(Armant & Kenneth, 2020)	✔	✔			✔			✔					✔	R	ME
Cheikh-Graiet et al., 2020)			✔	✔	✔			✔	✔				✔	A, T	MH
(Jintao et al., 2020)	✔					✔		✔	✔			✔	✔	A	AR

(Masmoudi et al., 2020)	✔	✔			✔			✔	✔	✔				R, T	MH
(Qin et al., 2020)	✔		✔		✔	✔		✔	✔		✔	✔		T	AR
(Turan et al., 2020)	✔		✔		✔	✔		✔			✔			T, R	AR
(Haliem, et al. 2020)	✔	✔	✔		✔			✔	✔			✔		A, T	AR

Tableau 1.2 : *Classification des travaux connexes.*

1.4. 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 et nous avons donné une classification de ces travaux.

Dans le chapitre suivant, nous allons exposer notre contribution pour résoudre le problème d'appariement dynamique dans un système de covoiturage.

Chapitre 2 : Contribution

2.1. Introduction

Nous nous intéressons dans ce chapitre à la modélisation et à la résolution du problème d'appariement dynamique sous contraintes dans un système de covoiturage.

Nous proposons de résoudre le problème considéré par l'apprentissage par renforcement (AR). L'aspect dynamique, le caractère spatio-temporel, le problème de décision (apparier ou non), les contraintes liées à l'environnement, tous ces facteurs se manifestent dans l'apprentissage par renforcement avec son aspect de prise de décision séquentielle.

Ce chapitre est consacré, dans un premier temps, à la présentation de la modélisation proposée. Cette modélisation est axée temps et l'objectif que nous fixons consiste à minimiser le temps d'attente de passagers. Ceci aura comme impact de satisfaire les passagers et ainsi maximiser le nombre de demandes achevées.

Dans un second temps, nous allons décrire les différentes étapes de réalisation de notre travail.

2.2. Problématique et objectifs de notre approche

Les villes d'aujourd'hui sont confrontées à des défis en termes de congestion, de manque d'espace, de croissance démographique, de qualité de l'air, de bruit, de santé et de développement économique. Les citoyens veulent être mobiles et se déplacer d'un lieu à un autre - dans et entre les villes - facilement, pas cher, intelligemment et proprement.

Pour pallier à ces problèmes, la mobilité partagée a gagné en popularité ces dernières années et a donné naissance au concept de covoiturage qui consiste à mettre en relation des conducteurs et des passagers qui ont des besoins de déplacement similaires afin d'effectuer des trajets communs. Nous nous intéressons dans notre travail à ce nouveau mode de transport : le covoiturage (*Ridesharing*).

Ce mode de transport alternatif consiste à mettre en relation deux personnes (un conducteur non professionnel et un passager) ou plus (1C→1P ; NC→NP), qui souhaitent voyager depuis des points de départ individuels vers des destinations individuelles.

Faire correspondre ces deux acteurs et leurs trajets leur permettra de partager un même véhicule. Cela a, évidemment, le potentiel de réduire le trafic, la consommation de carburant et la pollution, mais cela permet également aux utilisateurs de partager leurs frais de déplacement. Par conséquent, ce type de covoiturage est un sujet important dans la société moderne.

Au départ, les plateformes de covoiturage correspondaient aux personnes dont les points de départ et d'arrivée de leurs trajets étaient relativement proches les uns des autres. La recherche est très restrictive et la correspondance est fournie en répertoriant les conducteurs à une certaine distance autour de l'origine et de la destination de la personne à la recherche d'un trajet (le passager). En effet, dans ce type de covoiturage, les offres et les demandes des conducteurs et des passagers sont connues à l'avance, bien avant l'exécution d'un processus d'appariement.

De telles solutions statiques ont réussi pour certains types de solutions de covoiturage, mais ne fonctionnent pas bien en l'absence d'informations statiques disponibles avant la prise de décision.

De plus, de nos jours, les transports, comme de nombreux autres aspects de la vie quotidienne, sont transformés par la révolution des technologies de l'information. La généralisation des appareils mobiles ; la connexion accessible à tout le monde ce qui fait que les personnes peuvent facilement proposer et demander des voyages quand ils le souhaitent et où qu'ils soient et le développement du Global Positioning System (GPS), ont permis à tous les opérateurs de transport d'adapter en temps réel l'offre de transport à la demande de déplacements. Ces nouvelles technologies ont apporté des changements considérables aux différents modes de transport.

En effet, Ces options peuvent donner la possibilité d'avoir accès à la position des véhicules à tout moment et d'effectuer le processus d'appariement du covoiturage en temps réel. Ces possibilités ont conduit au développement et au progrès d'un nouveau type de covoiturage appelé covoiturage dynamique qui doit être capable de répondre aux demandes en quelques millisecondes et de fournir des appariements automatiques plus sophistiquées qu'une simple recherche radiale autour des origines et des destinations des trajets.

Le covoiturage dynamique se caractérise par une grande souplesse d'utilisation et moins d'interdépendance que le covoiturage statique. Il s'agit d'un service ou un processus automatisé qui fait apparier des offres de trajets des conducteurs et des demandes de trajets des passagers dans un délai très court en fonction de la position, du temps, … En d'autres termes, les nouveaux conducteurs, proposant des trajets, et les passagers, demandant des trajets, peuvent entrer et sortir du système à tout moment, et le système essaie alors de faire correspondre leurs trajets à court terme (ou même en cours de route).

Néanmoins, la majorité des systèmes opérationnels qui gèrent la multimodalité intègrent le concept de covoiturage comme une simple alternative quand le transport en commun n'est pas disponible. Ce manque d'optimisation présente un frein au développement du covoiturage comme un mode de transport à part entière. En effet, ce processus est beaucoup plus compliqué qu'il le parait et se caractérise par un aspect dynamique qui doit assurer une flexibilité et une instantanéité et un ensemble de contraintes qui doivent être gérées.

C'est dans ce cadre que s'intègre notre travail. Nous nous intéressons à explorer les méthodes d'optimisation afin de trouver une solution au problème de gestion dynamique d'un ensemble d'offres de covoiturage envoyées par les conducteurs et un ensemble de requêtes émises par les passagers.

Notre objectif est de mettre en place un système qui doit optimiser les solutions d'appariement dynamique *(un conducteur – un passager 1C→1P),* ayant comme objectif la *minimisation du temps d'attente des passagers*.

Comme nous l'avons présenté dans le chapitre précédent, le temps d'attente s'inscrit dans la liste des critères de qualité. La raison derrière notre choix de ce critère d'optimisation est qu'il a une influence directe sur la satisfaction des passagers puisque le système va répondre à leurs demandes le plus tôt possible prenant en compte le temps d'attente qu'ils tolèrent. Par conséquent, le taux d'annulation de demandes va diminuer et le nombre de demandes desservies va augmenter et ainsi améliorer la qualité du service offert.

Considérer l'aspect dynamique du covoiturage nous mène à assurer une certaine flexibilité dans le processus d'appariement. Ceci s'interprète par un ensemble de contraintes liées aux préférences

des conducteurs et des passagers et à l'environnement que nous devons satisfaire lors de la recherche des appariements (conducteur, passager).

En effet, Une caractéristique principale du problème de covoiturage dynamique est sa nature spatio-temporelle. L'éligibilité d'un conducteur à correspondre à une demande d'un passager dépend en partie de sa proximité spatiale avec la demande. De plus, les réponses à des demandes de trajet prennent généralement un temps différent pour se terminer, et elles modifient les états spatiaux des conducteurs, affectant la distribution de l'offre pour une correspondance future.

Par conséquent, les décisions opérationnelles en matière de covoiturage sont de nature séquentielle et ont une forte dépendance spatio-temporelle, ce qui offre d'excellentes applications de l'apprentissage par renforcement, une piste que nous allons explorer lors de notre travail.

Beaucoup de travaux ont considérés l'apprentissage par renforcement (Sutton & Barto, 2018) en réponse à ce type de problème (Qin et al., 2020 ; Zhou et al., 2019 ; …). La contribution majeure de chaque travail concerne la modélisation des différents paramètres du problème par les différents paramètres du fondement théorique de l'apprentissage par renforcement, à savoir, les processus décisionnels de Markov (MDP : Markov Decision Processes) (Putterman, 1994). Ainsi, notre objectif est de proposer une bonne modélisation du temps et de l'espace afin de répondre au mieux aux demandes des passagers en minimisant leur temps d'attente et en respectant leurs fenêtres temporelles et leurs points de départ et d'arrivée ainsi que celles des conducteurs (fenêtres temporelles et points de départ et d'arrivée).

2.3. Description du problème

Nous considérons le problème d'appariement dynamique dans un système de covoiturage temps réel. L'objectif est de trouver des appariements entre les couples (conducteur, passager). Le conducteur propose un trajet à partager quelques minutes avant son départ tout en spécifiant les paramètres nécessaires relatifs à son déplacement. Le passager, de son côté, dépose une demande de trajet quelques minutes avant son départ tout en spécifiant les paramètres nécessaires relatifs à son déplacement.

Notre problème se caractérise par un ensemble de contraintes qui doivent être gérées et respectées lors de sa résolution. L'ensemble de contraintes que nous considérons est le suivant :

1. **Contraintes imposées aux variables de décision concernant l'appariement des conducteurs/passagers.** A un moment donné, un conducteur ne peut avoir qu'un seul passager à bord, c'est-à-dire, chaque conducteur ne peut avoir qu'un seul passager à servir. Il ne peut pas servir un autre passager avant d'avoir déposé le passager qu'il a à bord.
2. **Contraintes imposées aux variables de décision concernant l'itinéraire.** Nous considérons un modèle de covoiturage inclusif (Furuhata et al., 2013). Ceci veut dire que l'origine et la destination d'un passager se trouvent toutes deux sur le trajet d'un itinéraire original du conducteur. Il faut noter que si un conducteur va servir plusieurs passagers ceci n'implique pas que les passagers ont des trajets identiques au conducteur.
3. **Contraintes imposées aux variables de décision concernant l'aspect spatio-temporel.** Le système que nous voulons mettre en œuvre, vise à mettre en relation des voyageurs ayant des itinéraires et des horaires similaires dans des délais très courts. Chaque acteur a une fenêtre temporelle dans laquelle son trajet doit avoir lieu.

2.3.1. Notation et terminologie

Nous donnons dans, dans cette section, une formalisation du problème sus-cité du côté du passager et du conducteur.

Passager. Nous notons $P = \cup_{i=1}^{n}\{P_i\}$ un ensemble de n passagers et $P_{nd}(t) = \{P_1, P_2, \ldots, P_i, \ldots, P_{nd}\}$ l'ensemble des nd demandes déposées par ces passagers au temps t et qui sont instantanément reçues sur le système avec $nd \leq n$.

Une demande d'un passager que nous notons P^l est définie par : $P_i(t) = (P_i^{l+}, P_i^{l-}, d_i, a_i)$, où :

- P_i^{l+}, est le point de départ (origine) du passager i.
- P_i^{l-}, est le point d'arrivée (destination) du passager i.
- d_i, indique le temps de départ au plus tard du passager i.
- a_i, spécifie le temps d'arrivée au plus tard du passager i.

Conducteur. Nous notons $V = \cup_{k=1}^{m}\{V_k\}$ un ensemble de m véhicules exprimant un ensemble

d'offres de partage de trajet (offre de véhicules) $V_m(t) = \{V_1, V_2, \ldots, V_k, \ldots, V_m\}$ déposées par les conducteurs au temps t et qui sont instantanément reçues sur le système.

Une offre d'un conducteur que nous notons V^k est définie par : $O_{V_k}(t) = (V_k^{l+}, V_k^{l-}, d_i, a_i)$, où :

- V_k^{l+}, est le point de départ (origine) du conducteur k.
- $, V_k^{l-}$, est le point d'arrivée (destination) du conducteur k.
- d_k, indique le temps de départ au plus tard du conducteur k.
- a_i, spécifie le temps d'arrivée au plus tard du conducteur k.

La figure 2.1 décrit un scénario de base de notre problème d'appariement dynamique où un conducteur propose de partager son véhicule sur un trajet du point A au point D. Au point A, une demande du passager 1 a été appariée à l'offre du conducteur 1. Ce passager va partager le trajet de A à C avec le conducteur 1 (son trajet est inclus dans le trajet du conducteur).

Au point B, sur le trajet du conducteur 1, le système reçoit une nouvelle demande de la part d'un deuxième passager. Cette demande ne peut être satisfaite car le véhicule est déjà occupé (1C → 1P). Au point C, le conducteur dépose le passager 1 et le système reçoit une nouvelle demande de la part du passager 3 qui son trajet est inclus dans le trajet du conducteur. Sa demande est appariée à l'offre du conducteur qui le récupère du point C et le dépose à sa destination (point D).

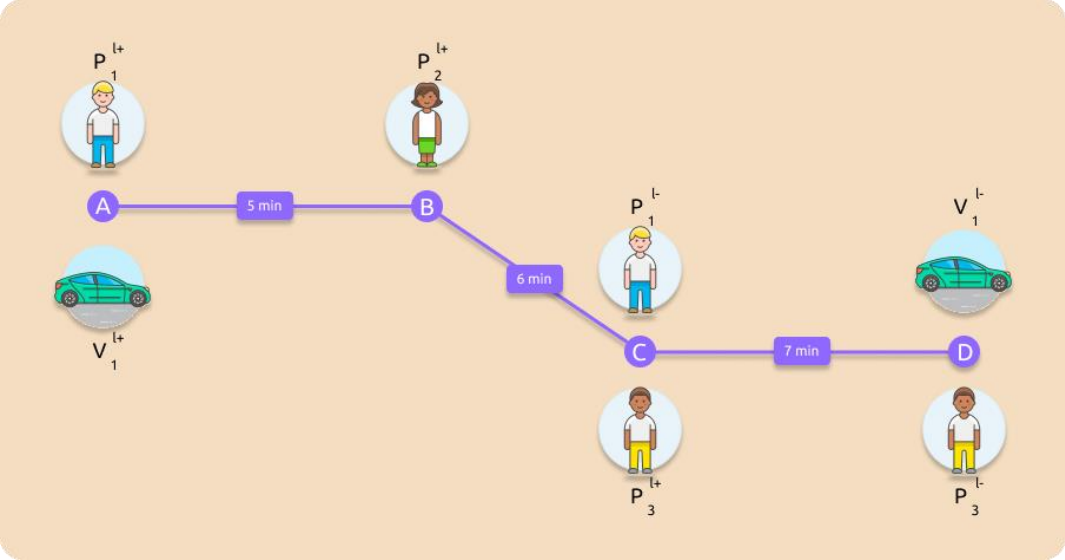


Figure 2.1 : *scénario d'appariement des trajets.*

Nous proposons de résoudre le problème d'appariement dynamique considéré par l'apprentissage par renforcement (AR). L'aspect dynamique, le caractère spatio-temporel, le

problème de décision (apparier ou non), les contraintes liées à l'environnement, tous ces facteurs se manifestent dans l'apprentissage par renforcement avec son aspect de prise de décision séquentielle.

Notre problématique est, à présent, comment modéliser les paramètres présentés précédemment par les différents paramètres de l'apprentissage par renforcement. En effet, l'AR trouve son fondement théorique dans les processus décisionnels de Markov (MDP). Un MDP est composé d'un ensemble d'états, un ensemble d'actions, une fonction de transition et une fonction de récompense. La modélisation du problème de décision d'appariement dynamique à contraintes par un MDP revient à construire chaque composant du MDP afin de prendre en considération l'information nécessaire à la décision.

Nous allons présenter dans ce qui suit les concepts de base de l'apprentissage par renforcement. Par la suite, nous allons présenter avec détails la modélisation du problème que nous proposons.

2.4. Apprentissage par renforcement

L'apprentissage par renforcement (Sutton & Barto, 2018) fait référence à une classe de problèmes d'apprentissage automatique, dont le but est d'apprendre, à partir d'expériences, ce qu'il faut de faire en différentes situations, de façon à optimiser une récompense numérique au cours du temps.

L'agent doit apprendre tout seul quelles actions entreprendre suite à des états de l'environnement. Son comportement est guidé par une récompense (positive ou négative) qui va lui aider à découvrir les actions qui rapportent le plus en les essayant.

L'agent cherche, au travers d'expériences itérées, un comportement décisionnel (appelé *stratégie* ou *politique*, et qui est une fonction associant à l'état courant l'action à exécuter) optimal, afin qu'il maximise la somme des récompenses au cours du temps (Wiering & Van Otterlo, 2012).

L'AR trouve son fondement théorique dans les processus décisionnels de Markov (MDP : Markov Decision Processes) (Putterman, 1994). Les MDPs sont un formalisme intuitif et fondamental pour la planification théorique de la décision (DTP), l'apprentissage par renforcement

(RL) et d'autres problèmes d'apprentissage dans les domaines stochastiques. Dans ce modèle, un environnement est modélisé comme un ensemble d'états et d'actions pouvant être effectué pour contrôler l'état du système. Le but est de contrôler le système de manière à ce que certains critères de performance soient maximisés. De nombreux problèmes tels que les problèmes de planification (stochastiques), l'apprentissage du contrôle des robots et les problèmes de jeu ont été modélisés avec succès en termes de MDP. En fait, les MDP sont devenus le formalisme standard de facto pour apprendre la prise de décision séquentielle (Sutton & Barto, 2018 ; Wiering & Van Otterlo, 2012).

2.4.1. Éléments de l'apprentissage par renforcement

Un système basé sur l'apprentissage par renforcement se caractérise par (Dangeti, 2017) :

- **Environnement :** tout système possédant des états, et des mécanismes de transition entre les états.
- **Agent :** Il s'agit d'un système automatisé qui interagit avec l'environnement.
- **État :** L'état de l'environnement ou du système est l'ensemble des variables ou des caractéristiques qui décrivent cet environnement.
- **Action :** elle est interprétée l'interaction entre l'agent et l'environnement. Cela définit la transition entre les états
- **Politique :** Elle définit l'action à sélectionner et à exécuter pour tout état de l'environnement. En d'autres termes, la politique est le comportement de l'agent ; c'est une carte de l'état à l'action. Quand il s'agit d'une politique optimale, elle fournit la meilleure action à entreprendre dans chaque situation.
- Le Facteur d'escompte γ, qui est un nombre compris entre 0 et 1. Ce facteur décrit dans quelle mesure un agent préfère les récompenses actuelles aux récompenses futures.
- **Transitions :** Les probabilités de transition caractérisent la dynamique de l'état du système. Elles caractérisent les réactions de l'environnement aux actions émises par l'agent.

- **Récompenses :** Elle quantifie l'interaction positive ou négative de l'agent avec l'environnement. Les récompenses sont généralement des gains immédiats réalisés par l'agent atteignant chaque état.

$$r(s, a, s') = E[R_t | S_{t-1} = s, A_{t-1} = a, S_t = s'] \quad (1)$$

Où, $r(s, a, s')$ est la récompense reçue par l'agent à l'état s' après avoir exécuté l'action a à l'état s.

- **Fonction de valeur :** Une fonction de valeur $V_\pi(s)$ est une prédiction des récompenses futures de chaque état. Elle est utilisée pour évaluer le bien/mal des états, sur la base duquel l'agent choisira/agira pour sélectionner le meilleur état suivant :

$$V_\pi(s) = E[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots | S_t = s] \quad (2)$$

- **Épisode :** Il définit le nombre d'étapes nécessaires pour atteindre l'état cible à partir d'un état initial. Les épisodes sont également connus sous le nom d'essais.
- **Horizon :** Il s'agit du nombre d'étapes ou d'actions futures utilisées dans la maximisation de la récompense. L'horizon peut être infini, auquel cas les récompenses futures sont actualisées afin que la valeur de la politique converge.
- **Fonction État-Action :** fonction Etat-Action, $Q_\pi(s, a)$ représente la récompense cumulative qu'un agent va recevoir lorsqu'il prend l'action A dans l'état S, et se comporte selon la politique $\pi(a, s)$.

$$Q_\pi(s, a) = E[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots | S_t = s, A_t = a] \quad (3)$$

Les méthodes d'apprentissage par renforcement sont dites indirectes ou directes selon que l'on maintient ou non un modèle explicite de la dynamique du PDM que l'on cherche à contrôler (déterminé par les probabilités de transition $p(s' \mid s, a)$ et les récompenses $r(s, a)$).

Les méthodes de résolution de l'apprentissage par renforcement peuvent être classées en deux grandes classes. Les méthodes basées sur modèle ce sont des méthodes qui maintiennent un modèle explicite de la dynamique du MDP, elles cherchent à apprendre les fonctions de transition et de récompense, puis à utiliser, entre autres, la programmation dynamique sur ces approximations afin d'améliorer la politique (Sutton & Barto, 2018).

La deuxième catégorie est sans modèle appelée : *model-free*. Ces méthodes ne reposent pas sur la disponibilité de modèles de transition et de récompense connus à priori, c'est-à-dire un modèle du MDP. L'absence de modèle génère un besoin d'échantillonner le MDP pour recueillir des connaissances statistiques sur ce modèle inconnu.

Il existe de nombreuses techniques RL sans modèle qui sondent l'environnement en effectuant des actions, estimant ainsi le même type de fonctions de valeur d'état et de valeur d'état-action que les techniques basées sur un modèle tels que : Q-learning, la différence temporelle, méthode de Monte Carlo, etc (Sutton & Barto, 2018).

2.5. Modélisation proposée

Nous adressons le problème d'appariement dynamique sous contraintes spatio-temporelles entre les offres d'un conducteur et les demandes d'un passager, dans le cadre d'un système de covoiturage temps réel. Nous proposons de résoudre ce problème par l'apprentissage par renforcement. Par conséquent, nous devons construire les différents composants d'un MDP afin de modéliser les caractéristiques du problème, ses contraintes et l'objectif à optimiser.

Nous allons, dans ce qui suit, présenter la modélisation proposée par un MDP.

La modélisation du problème présenté à la section 2.3 par un MDP revient à définir les différents éléments qui composent un MDP (espace d'états, espace d'actions, fonction de transition et fonction de récompense).

Nous proposons que le conducteur soit au cœur de notre modélisation et qu'il représente l'agent, ainsi, il va apprendre à décider s'il prend un passager ou non, à un endroit précis et à un instant précis. Cette décision doit optimiser le temps d'attente du passager dans notre système.

La transition correspond à la terminaison d'un trajet : lorsqu'une demande est affectée à un conducteur et se termine avec succès tenant compte des contraintes d'itinéraire. Le conducteur, par la suite, gagne une récompense immédiate grâce à cette transition.

2.5.1. Etat

L'Etat, s est capturé par les coordonnées géographiques du conducteur, du passager, du point de destination suivant et du temps, c'est-à-dire $s_t = (V^l, P^l, N^l, t)$ ces éléments représentent:

- V^l – est la paire de coordonnées GPS (latitude, longitude) du conducteur,
- P^l – est la paire de coordonnées GPS (latitude, longitude) du passager,
- N^l – est la paire de coordonnées GPS (latitude, longitude) du point de destination suivant.
- t – est le temps.

Il est possible qu'il n'y ait pas de passager (demande) à l'endroit où se trouve le conducteur ou au point de destination suivant. Dans ce cas, P^l est égale à nul.

2.5.2. Observation

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

- ND_t – le temps d'arrivée à la prochaine destination,
- PW_t – le temps d'attente actuel du passager,
- PMW_t – le temps d'attente maximum du passager

Il faut noter que lorsque le conducteur est état d'attente et non pas en chemin $ND_t = 0$. En outre, lorsqu'il n'y a pas de passager, $PW_t = PMW_t = 0$.

Notre choix de ces trois paramètres est justifié par :

(1) Notre objectif global qui est *la minimisation du temps d'attente des passagers* ainsi, le temps représente une information nécessaire pour la prise de décision et une précision dynamique qui assure une stabilité et une rapidité de réponse de notre système.

(2) Après avoir fait le tour des travaux qui ont utilisé l'AR pour résoudre ce type de problème, que nous avons présenté dans le chapitre précédent, nous avons constaté que tous ces travaux transmettent la localisation des conducteurs et des passagers à l'agent pour pouvoir prendre une décision ce qui augmente l'espace. Or, dans notre travail, nous avons supposé que le temps est une information suffisante pour la prise de décision et proche de la réalité. En effet, le temps est un paramètre tranchant dans la décision : ***si le temps ne permet pas de prendre le passager, il est inutile de voir la localisation.*** Cette situation figure dans le cas où le temps d'attente actuel du passager dépasse son temps d'attente maximum.

2.5.3. But

Le critère d'optimisation que nous considérons consiste à minimiser le temps d'attente des passagers. En effet, le temps d'attente s'inscrit dans la liste des critères de qualité. La raison derrière notre choix de ce critère d'optimisation est qu'il a une influence directe sur la satisfaction des passagers puisque le système va répondre à leurs demandes le plus tôt possible prenant en compte le temps d'attente qu'ils tolèrent. Par conséquent, le taux d'annulation de demandes va diminuer et le nombre de demandes desservies va augmenter et ainsi améliorer la qualité du service offert.

2.5.4. Action

L'action, a est la réponse du conducteur à un passager particulier, qui est simplement définie par la décision du conducteur qui est :

- $a = 0 \rightarrow$ refuser l'offre
- $a = 1 \rightarrow$ accepter l'offre

Exemple. On va se référer dans notre exemple au scénario présenté dans la figure 2.1.

Le conducteur V_1^{l+} et le passager P_1^{l+} se trouvent au même endroit, comme illustré dans Figure. Nous allons supposer que le passager a déjà attendu 10 minutes et que son temps d'attente maximum est égale 30 min, l'environnement renvoie l'état actuel :

$$s_0 = (0\,,30\,,10\,)$$

L'action dans ce cas est $a = 1$.

Nous allons, à présent, considéré le conducteur V_1^{l+} et le passager P_2^{l+}. Nous allons supposer que P_2^{l+} a déjà attendu 16 min et que son temps d'attente maximum est égale à 20, donc l'état retourné est le suivant :

$s_0 = (5, 20, 16)$.

L'action dans ce cas est $a = 0$.

Il faut noter que lorsque le temps de la prochaine destination est égal à zéro, cela signifie que le conducteur et le passager se trouvent au même endroit, sinon le passager se trouve à la prochaine destination du conducteur.

2.5.5. Récompense

Le signal de récompense dans notre problème doit présenter une information qui guide le comportement de l'agent afin d'atteindre l'objectif principal du système qui est la minimisation du temps d'attente. De ce fait, nous proposons que ce signal, r, correspond au temps d'attente gagné par les passagers un trajet et est une fonction de l'état actuel (temps de la prochaine destination, temps d'attente actuel du passager et le temps d'attente maximal du passager) et est définie par :

$$r = PMW_t - (PW_t + ND_t) \quad (4)$$

Lorsque l'action correspond au refus ($a = 0$) alors la récompense est égale à zéro.

Le cas où le conducteur accepte une demande vide est aussi considéré. Par exemple, si le conducteur V_1^{l+} du scénario de la figure 2.1 est au point A et il ne reçoit aucune demande, donc l'état renvoyé est : $s = (5, 0, 0)$. Lorsque l'agent accepte une telle demande vide, alors la récompense sera : $r = 0 - (0 + 5) = -5$. Dans ce cas le conducteur apprendra qu'il a fait un mauvais choix pour l'éviter la prochaine fois.

Le signal de récompense que nous proposons dans notre modélisation signifie que moins il y a de temps d'attente, plus la récompense est élevée. L'agent doit donc agir de manière à maximiser la récompense, afin d'atteindre notre objectif de minimiser le temps d'attente.

2.5.6. Episode

Un épisode est un trajet complet du conducteur, de V_k^{l+} à V_k^{l-}. Par conséquent, un état terminal est un état correspondant à V_k^{l-}.

2.5.7. Fonction état-action

La fonction de valeur état-action, $Q(s,a)$ est la récompense cumulative attendue que le conducteur gagnera à la fin d'un épisode s'il commence à l'état s et effectue une action a. Mathématiquement :

$$Q(s,a) = E\left[\sum_{t=0}^{T} \gamma^t R(S_t, A_t) | S_t = s, A_t = a\right] \quad (5)$$

Où S_t est l'état au temps t, A_t est l'action exécutée au temps t et R est la récompense reçue quand l'agent choisit d'exécuter A_t dans S_t. T est le nombre d'étapes de transition jusqu'à l'état terminal et γ est le facteur d'actualisation des récompenses futures.

2.5.8. Politique

La politique, $\pi(a|s)$ est une fonction qui fait correspondre à chaque état s, une action a. La politique avide par rapport à un $Q(s,a)$ appris est donnée par $\pi(s) := argmax_a\, Q(s,a)$.

2.6. Phase d'apprentissage

Afin de réaliser notre système, deux étapes de réalisation se distinguent. La première consiste à une phase d'apprentissage qui a comme objectif de construire un modèle où l'agent va apprendre un comportement répondant aux contraintes et en optimisant l'objectif considérés. La deuxième concerne le lancement de ce modèle dans un simulateur. Nous allons présenter, dans cette section, la phase d'apprentissage. Nous réservons la discussion du simulateur avec détails pour le chapitre implémentation.

2.6.1. Données

Afin de construire un modèle, nous avons besoin d'un ensemble de données contenant les informations nécessaires correspondant à notre description du problème aux directions de chaque

trajet. Pour ce faire, nous allons utiliser la base de données *Uber Pickup*[1] dans la ville de New York.

Ce répertoire contient des données sur plus de 4,5 millions de prises en charge Uber dans la ville de New York d'avril à septembre 2014, et 14,3 millions de prises en charge Uber supplémentaires de janvier à juin 2015. Des données au niveau des trajets sur 10 autres entreprises de véhicules de location (FHV), ainsi que des données agrégées pour 329 entreprises de FHV, sont également incluses. Tous les fichiers sont tels qu'ils ont été reçus les 3 août, 15 septembre et 22 septembre 2015.

A partir de ce répertoire, nous allons travailler avec un fichier portant le nom "*Federal_02216"* qui contient des données brutes sur les pick-ups d'une entreprise de FHV non Uber. La figure 2.2 présente un fragment du fichier *Federal_02216.* Ce fichier contient les informations suivantes :

- *Date* : la date de dépôt de la demande,
- *Time* : l'heure de dépôt de la demande,
- *PU_Adress* : l'adresse de récupération,
- *PU_Adress* : l'adresse de dépôt,
- *Routing Details* : Détails du routage,

  Status : statut de la demande.

Date	Time	PU_Address	DO_Address	Routing D...	PU_Address	Status
07/01/2014	07:15 AM	Brooklyn Museum, 200 Eastern Pkwy., BK NY;	1 Brookdale Plaza, BK NY;	PU: Brooklyn Museum, 200 Eastern Pkwy., BK NY; DO: 1 Brookdale Plaza, BK NY;	Brooklyn Museum, 200 Eastern Pkwy., BK NY; DO: 1 Brookdale Plaza, BK NY;	Cancelled
07/01/2014	07:30 AM	33 Robert Dr., Short Hills NJ;	John F Kennedy International Airport, vitona Airlines;	PU: 33 Robert Dr., Short Hills NJ; DO: John F Kennedy International Airport, vitona Airlines;	33 Robert Dr., Short Hills NJ; DO: John F Kennedy International Airport, vitona Airlines;	Arrived
07/01/2014	08:00 AM	60 Glenmore Ave., BK NY;	2171 Nostrand Ave., BK NY;	PU: 60 Glenmore Ave., BK NY; DO: 2171 Nostrand Ave., BK NY;	60 Glenmore Ave., BK NY; DO: 2171 Nostrand Ave., BK NY;	Assigned

Figure 2.2 : *Fichier informatif de trajets Federal_02216.*

[1] https://www.kaggle.com/datasets/fivethirtyeight/uber-pickups-in-new-york-city.

A partir de cet ensemble des trajets, nous allons générer un nouvel ensemble de donnée qui contient plus de détails sur les directions dont nous avons besoin, en utilisant l'API de Google Direction.

Plus de détails concernant cette phase seront donnés dans le prochain chapitre.

2.6.2. Initialisation de l'environnement

Dans cette sous-étape, à partir de l'ensemble de données généré à la sous- étape précédente, nous allons choisir un trajet aléatoire pour le conducteur et un trajet aléatoire pour le passager afin d'extraire les données nécessaires à la préparation de l'environnement (figure 2.3).

Pour un conducteur V_k, nous allons extraire ses points de départ et d'arrivées V_k^{l+}, V_k^{l-}, respectivement ; ses temps de départ et d'arrivée au plus tard d_k, a_k, respectivement.

En effet, du temps de dépôt de l'offre au temps de départ au plus tard s'inscrit le temps que peut attendre un conducteur avant de démarrer. Nous choisissons ce temps d'une manière aléatoire de l'intervalle 15 à 30 minutes.

De plus, le temps d'arrivée au plus tard peut être calculé en ajoutant le temps du trajet au temps de départ au plus tard.

Nous définissons également un nombre de points de contrôle qui représente l'ensemble de points dans lesquels le conducteur vérifie la disponibilité des demandes.
Pour un passager P_i, nous allons extraire ses points de départ et d'arrivée P_i^{l+}, P_i^{l-}, respectivement ; ses temps de départ et d'arrivée au plus tard d_i, a_i , respectivement.

De même, du temps de dépôt de la demande au temps de départ au plus tard s'inscrit le temps que peut attendre un passager. Nous choisissons ce temps d'une manière aléatoire de l'intervalle 15 à 30 minutes.

De plus, le temps d'arrivée au plus tard peut être calculé en ajoutant le temps du trajet au temps de départ au plus tard.

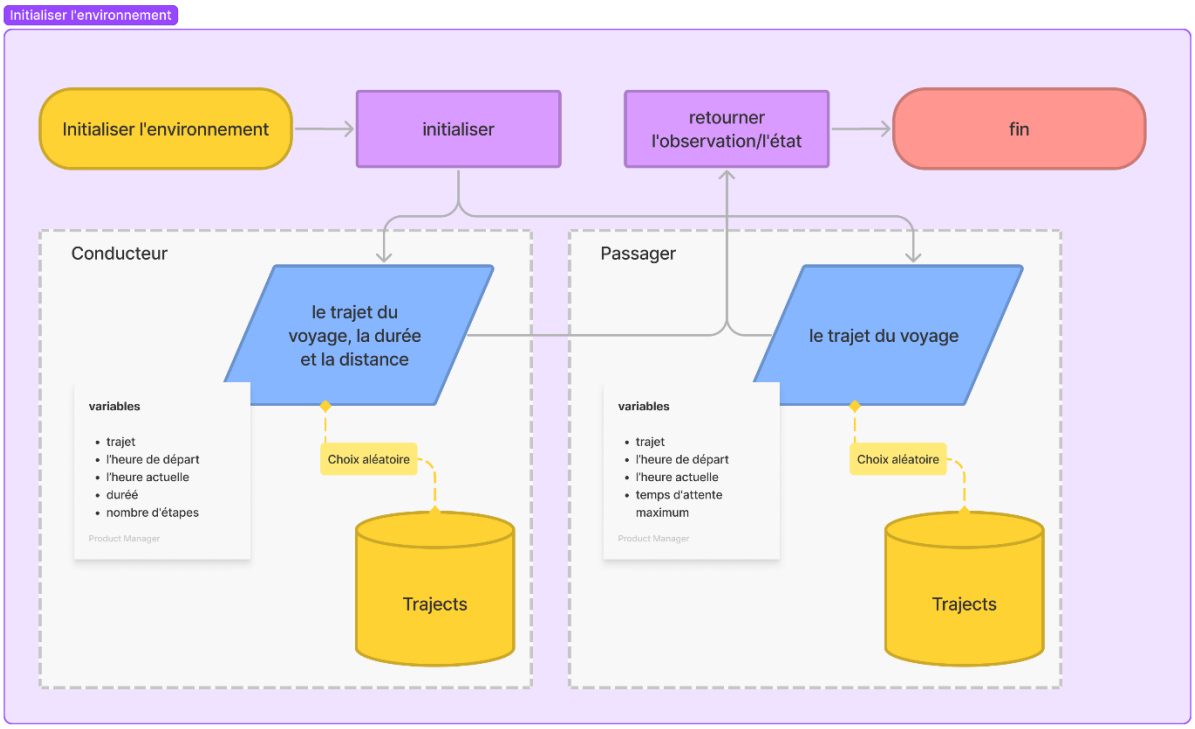


Figure 2.3 : *initialisation de l'environnement.*

2.6.3. Décision de l'agent

Après avoir choisi une action, l'agent fait un pas et exécute cette action. Une fois l'exécution est achevée, l'agent peut savoir si cette action représente une bonne décision. Dans le cas contraire, l'agent évitera de prendre cette même décision s'il sera face à la même situation (si le même état se produit dans le futur).

- **Processus d'exécution** : Premièrement, le conducteur vérifie s'il a accepté un passager mais qu'il ne l'a pas encore récupéré, c'est-à-dire que le conducteur a seulement notifié le passager, alors le conducteur va récupérer le passager.

  Si le passager n'a pas encore été notifié, donc deux possibilités sont disponibles pour le conducteur : soit le passager est déjà avec lui, soit il n'a pas encore reçu de demande. Dans le cas où le passager est à bord, le conducteur va le déposer à sa destination. Dans le cas contraire, le conducteur va vérifier s'il y a de nouvelles demandes ou non.
- **Processus de décision** : Quand le conducteur reçoit une nouvelle demande, cette demande va être traitée en se basant sur le temps d'attente, le temps actuel et le temps de la prochaine destination, et ainsi l'agent va décider s'il accepte la demande ou non. Dans le cas d'acceptation, il va soit prendre le passager, soit le notifier.

Après avoir calculé la récompense, si le conducteur est déjà sur la route ou s'il a une demande à servir, alors il va à la prochaine destination ; sinon il vérifie si c'est le moment de partir, sinon il attend. Après avoir accompli cette tâche, l'environnement renvoie le nouvel état en fonction de ces nouvelles informations.

Ce processus tourne en boucle jusqu'à ce que le conducteur arrive à sa destination. Les deux phases : exécution et décision sont présentées dans la figure 2.4.

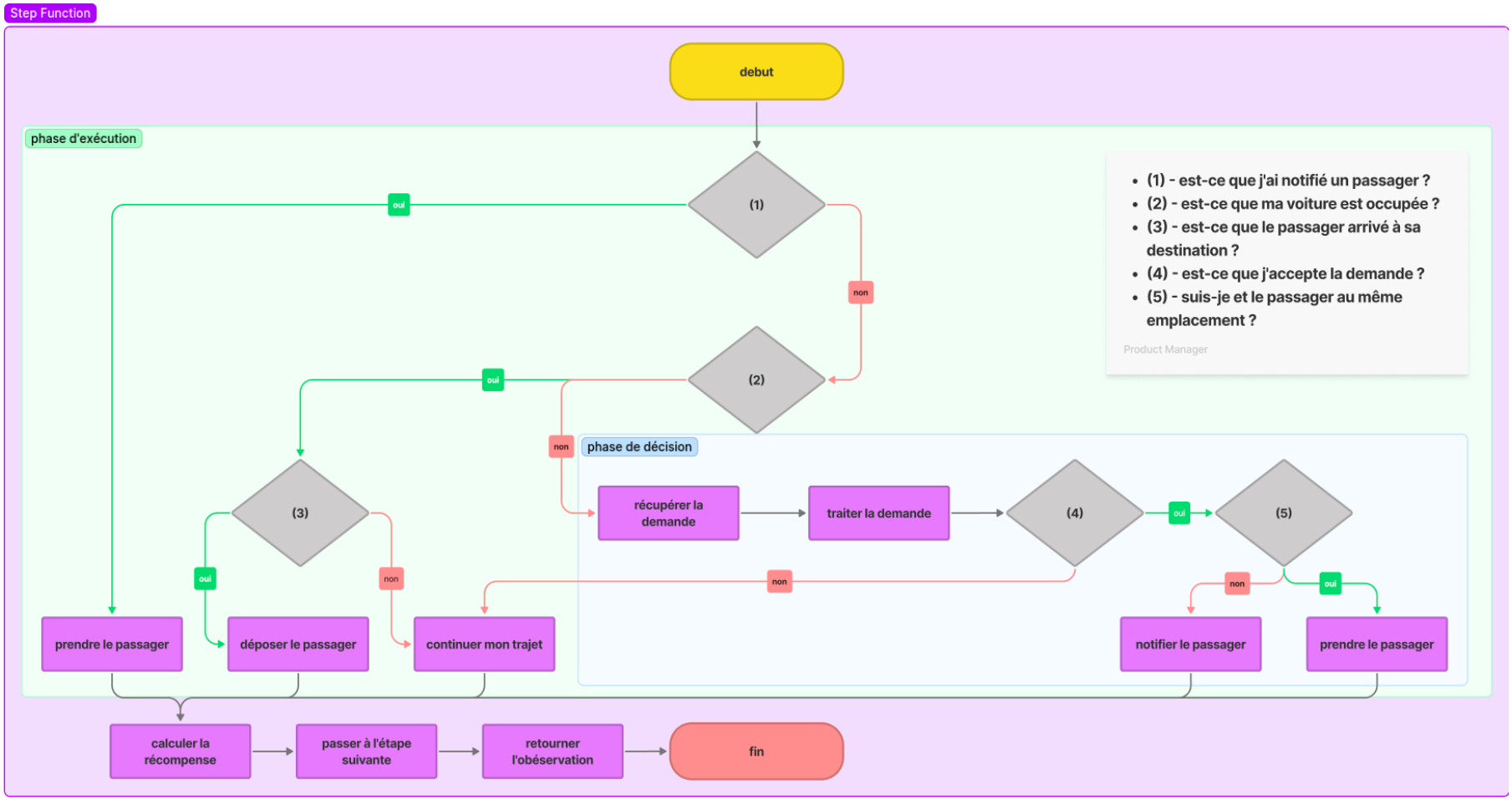


Figure 2.4 : *Décision/Exécution.*

Exemple illustratif

Prenons un exemple pour voir comment cela fonctionne. En se basant sur le scénario présenté dans la figure 2.1, nous supposons que le conducteur V_1 et les passagers P_1, P_2 postent leurs offres et demandes à 8:00, et le passager P_3 à 8 :01.

L'exécution de ce scénario nous donne plusieurs plans d'exécution possibles donnés dans la figure 2.5. Dans le tableau 2.1, nous donnons la récompense cumulée de chaque plan.

Tableau 2.1 : *Récompense cumulée du scénario d'exécution.*

Scénario	**Récompense cumulée**
(Oui, Oui, Oui, Oui, Oui)	12
(Oui, Non, Oui, Oui, Oui)	18
(Oui, Non, Oui, Non, Oui)	-3
(Oui, Non, Oui, Oui, Non)	24
(Oui, Non, Non, Oui, Oui)	23
(Oui, Non, Non, Non, Oui)	3
(Oui, Non, Non, Oui, Non)	30
…	…
(Non, Oui, Non, Oui, Non)	23
…	…
(Non, Non, Non, Non, Non)	0

Donc le meilleur plan est (Oui, Non, Non, Oui, Non).

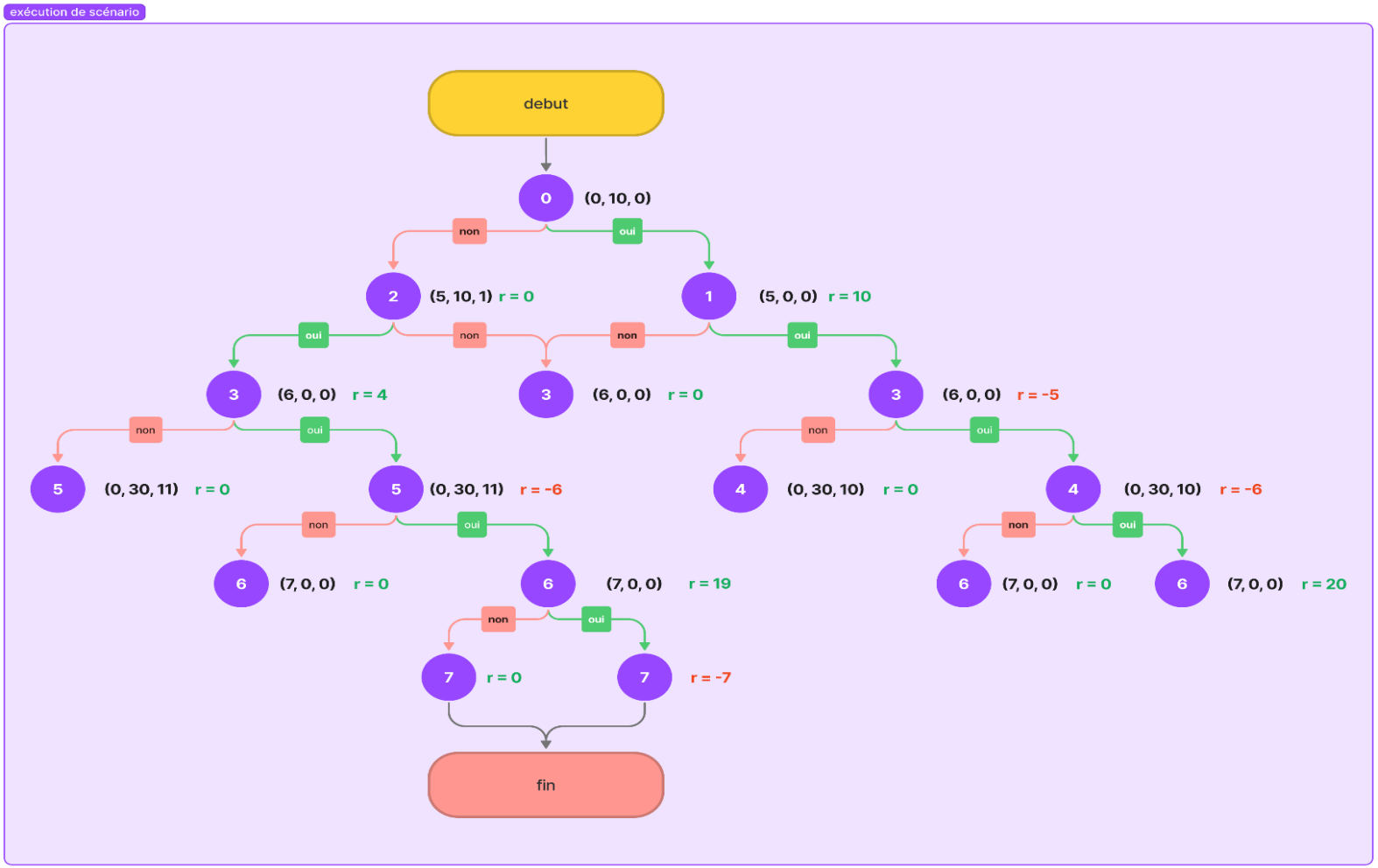


Figure 2.5 : *Scénario d'exécution.*

2.6.4. Apprentissage

Notre choix d'algorithme d'apprentissage par renforcement s'est orienté vers l'algorithme : Optimisation de la politique proximale (PPO) (Schulman et al., 2017). PPO est l'un des plus simples et des plus populaires des algorithme d'AR.

PPO est une méthode *model-free*, de gradient de politique et peut être utilisée pour des environnements avec des espaces d'action discrets ou continus. Cette technique alterne entre l'échantillonnage de données par interaction avec l'environnement et l'optimisation d'une fonction objective « substitut » à l'aide du gradient de montée. Alors que les méthodes de gradient de politique standard effectuent une mise à jour de gradient par échantillon de données, la fonction objective de PPO permet plusieurs époques de mises à jour de mini-lots (Schulman et al., 2017).

PPO est devenu l'algorithme d'apprentissage par renforcement par défaut d'OpenAI en raison de sa facilité d'utilisation et de ses bonnes performances.

Principe. Tout d'abord, PPO recueille un ensemble de trajectoires pour chaque époque en échantillonnant la dernière version de la politique. Ensuite, les récompenses à venir et les estimations de l'avantage sont calculées afin de mettre à jour la politique et d'ajuster la fonction de valeur. La politique est mise à jour via un optimiseur à gradient ascendant, tandis que la fonction de valeur est ajustée via un algorithme à gradient descendant. Cette procédure est appliquée pendant de nombreuses époques jusqu'à ce que l'environnement soit résolu [16].

L'idée principale est qu'après une mise à jour, la nouvelle politique ne doit pas être trop éloignée de l'ancienne. Pour cela, PPO utilise le clipping (coupure) pour éviter une mise à jour trop importante [17].

Après la création de l'environnement et la mise en place de la décision de l'agent lorsqu'il reçoit une observation, nous lançons L'étape d'apprentissage. Cette étape vise à fournir une compréhension quantitative des modèles spatio-temporels de l'offre des conducteurs et de la demande des passagers sur toutes les données.

Compte tenu des données historiques, nous construisons un processus de décision de Markov (MDP) avec un agent représentant un conducteur individuel dans le système. Les fonctions de

valeur apprises produisent la "valeur" pour chaque état spatio-temporel, qui est ensuite utilisée dans l'étape de planification en ligne suivante.

Nous allons discuter cette étape plus en détails dans le prochain chapitre.

2.7. Conclusion

Nous avons présenté dans ce chapitre la modélisation que nous avons proposée pour le problème d'appariement dynamique sous contraintes par les différents paramètres de l'apprentissage par renforcement. Nous avons détaillé toutes les étapes par lesquelles passent notre système afin de construire un modèle et lancer par la suite l'exécution.

Nous présentons dans le chapitre suivant, les détails d'implémentation de notre application.

Chapitre 3 : Implémentation

3.1. Introduction

Dans ce chapitre, nous décrivons les travaux applicatifs et les détails de mise en œuvre de notre travail. Nous allons décrire, d'abord, les outils et langage d'implémentation. Par la suite, nous présentons la base de données utilisée pour valider et entraîner notre système. Nous détaillons, ensuite, l'implémentation et l'apprentissage de notre système, et nous terminons par présenter le simulateur, les résultats obtenus et les expérimentations.

3.2. Mise en œuvre de la contribution

Dans cette section, nous allons spécifier, dans un premier temps, les outils utilisés pour développer notre application. Dans un second temps, nous allons décrire notre système.

3.2.1. Langages et outils de développement

- ***Le langage Python***

Nous avons choisi le langage python comme langage de programmation. Python est un langage de programmation interprété, c'est-à-dire qu'il n'est pas nécessaire de le compiler avant de l'exécuter. Il est gratuit, mais il peut être utilisé sans restriction dans des projets commerciaux. Il est portable, non seulement sur diverses variantes d'Unix, mais également sur des systèmes d'exploitation propriétaires : Mac OS, BeOS, NeXTStep, MS-DOS et diverses variantes de Windows [1]. La syntaxe de Python est très simple, combinée à des types de données avancés (listes, dictionnaires...), permet de générer des programmes très compacts et très lisibles. Python est un langage évolutif, soutenu par une communauté d'utilisateurs enthousiastes et responsables, dont la plupart sont des partisans du logiciel libre (Swinnen, 2012).

- ***Le navigateur Anaconda***

Le navigateur Anaconda est une interface utilisateur graphique (GUI) de bureau inclus dans la distribution Anaconda qui nous permet de lancer des applications et de gérer facilement les

packages, les environnements et les canaux conda sans utiliser des commandes de ligne de commande [2].

- ***Jupyter NoteBook***

Jupyter Notebook et son interface flexible étendent le notebook au-delà du code à la visualisation, au multimédia, à la collaboration, etc. En plus d'exécuter le code, il stocke le code et la sortie, ainsi que les notes de démarque, dans un document modifiable appelé bloc-notes. Lorsqu'il est enregistré, celui-ci est envoyé du navigateur au serveur du bloc-notes, qui l'enregistre sur le disque en tant que fichier JSON avec une extension.ipynb. [3]

- ***Visual Studio Code***

Visual Studio Code est un éditeur de code source léger mais puissant qui s'exécute sur votre bureau et est disponible pour Windows, macOS et Linux. Il est livré avec un support intégré pour JavaScript, TypeScript et Node.js et dispose d'un riche écosystème d'extensions pour d'autres langages (tels que C++, C#, Java, Python, PHP, Go) et moteurs d'exécution (tels que .NET et Unity). [4]

- ***Spyder***

Spyder est un environnement scientifique gratuit et open source écrit en Python, pour Python, et conçu par et pour les scientifiques, les ingénieurs et les analystes de données. Il offre une combinaison unique des fonctionnalités avancées d'édition, d'analyse, de débogage et de profilage d'un outil de développement complet avec l'exploration de données, l'exécution interactive, l'inspection approfondie et les magnifiques capacités de visualisation d'un progiciel scientifique. [5]

➔ ***Les bibliothèques utilisées***

- ***Pandas***

Pandas est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles. [6]

- ***Numpy***

NumPy est une bibliothèque pour le langage de programmation Python, ajoutant la prise en charge de grands tableaux et matrices multidimensionnels, ainsi qu'une grande collection de fonctions mathématiques de haut niveau pour opérer sur ces tableaux. [7]

- ***KivyMD***

Il s'agit d'une collection de widgets conformes au Material Design à utiliser avec le cadre graphique multiplateforme Kivy, un cadre pour les applications graphiques tactiles multiplateformes. L'objectif du projet est de se rapprocher le plus possible de la spécification Material Design de Google sans sacrifier la facilité d'utilisation ou les performances des applications [8].

- ***Gym***

Gym est une boîte à outils pour le développement et la comparaison d'algorithmes d'apprentissage par renforcement. Il ne fait aucune hypothèse sur la structure de votre agent et est compatible avec toute bibliothèque de calcul numérique, comme TensorFlow ou Theano. [9]

- ***Stable Baselines3***

Stable Baselines3 (SB3) est un ensemble d'implémentations fiables d'algorithmes d'apprentissage par renforcement dans PyTorch. [10]

- ***TensorBoard : La boîte à outils de visualisation de TensorFlow***

TensorBoard (Figure 3.1) fournit la visualisation et les outils nécessaires à l'expérimentation de l'apprentissage automatique [11] :

- Suivi et visualisation des métriques telles que la perte et la précision,
- Visualisation du graphique du modèle (opérations et couches),
- Visualisation d'histogrammes de poids, de biais ou d'autres tenseurs en fonction de leur évolution dans le temps,
- Projection des incorporations dans un espace de dimension inférieure,
- Affichage d'images, de textes et de données audio,
- Profiler les programmes TensorFlow,
- Et bien d'autres choses encore.

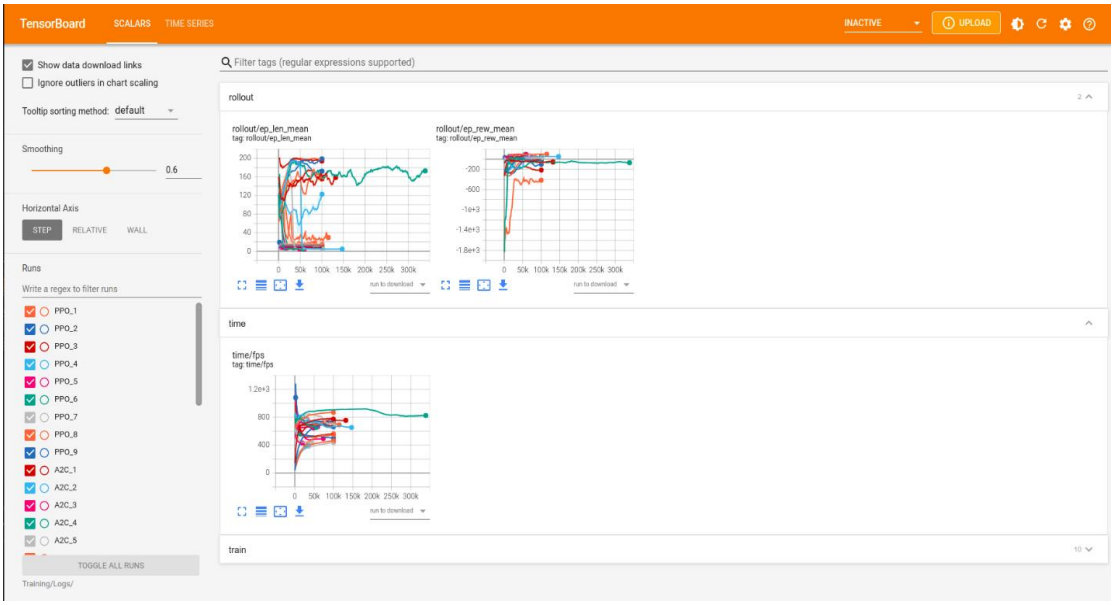


Figure 3.1 : *Interface de TensorBoard.*

➔ ***Maps API - Développeurs Google***

- **Directions API**

L'API Directions est un service web qui utilise une requête HTTP pour renvoyer des instructions au format JSON ou XML entre des lieux [12].

➔ ***Outil de désign***

- **Figma**

Figma est un éditeur de graphiques vectoriels et un outil de prototypage qui est principalement basé sur le web, avec des fonctionnalités hors ligne supplémentaires activées par des applications de bureau pour macOS et Windows. L'ensemble des fonctionnalités de Figma se concentre sur l'utilisation dans la conception d'interfaces et d'expériences utilisateur, en mettant l'accent sur la collaboration en temps réel [13].

3.3. Base de données

3.3.1. Description de la base

Dans notre travail, nous avons utilisé la base de données Uber Pickup dans la ville de New York, les données sont disponibles à [14]. Ce répertoire contient des données sur plus de 4,5 millions de prises en charge Uber dans la ville de New York d'avril à septembre 2014, et 14,3 millions de prises en charge Uber supplémentaires de janvier à juin 2015. Des données au niveau

des trajets sur 10 autres entreprises de véhicules de location (FHV), ainsi que des données agrégées pour 329 entreprises de FHV, sont également incluses. Tous les fichiers sont tels qu'ils ont été reçus les 3 août, 15 septembre et 22 septembre 2015.

Dans ce répertoire, nous avons choisi un fichier portant le nom *"Federal_02216"*. Ce fichier contient des données brutes sur les pick-ups d'une entreprise de FHV non Uber. Les informations sur le voyage varient d'une entreprise à l'autre, mais peuvent inclure le jour du voyage, l'heure du voyage, le lieu de départ, le lieu d'arrivée, les détails de l'itinéraire et le statut.

3.3.2. Echantillon de données

Pour effectuer nos expérimentations, nous avons choisi d'une manière aléatoire un échantillon de données du *Federal_02216*. Cet échantillon est constitué de 93 trajets, puis nous avons utilisé l'API de Google Direction pour obtenir les détails de la direction. Chaque demande renvoie ces informations (figure 3.2) :

- ***Geocoded-waypoints*** : liste des points de passage
- ***Routes*** : liste de tous les itinéraires possibles, chaque itinéraire contient
  - ***Boundes*** : limites
  - ***Copyright***
  - ***Legs*** : contient des informations sur le trajet
    - ***Distance*** : la distance du trajet
    - ***Duration*** : la durée du trajet
    - ***Start_location*** : emplacement de départ présenté par des paires de coordonnées GPS (latitude, longitude)
    - ***Start_address*** : adresse du lieu de départ
    - ***End_location*** : emplacement de destination présenté par des paires de coordonnées GPS (latitude, longitude)
    - ***End_address*** : adresse de l'emplacement final (destination)
    - ***Steps*** : liste qui contient des informations entre les points, chacun contient :
      - ***Polyline*** : la ligne entre deux points
      - ***Distance*** : la durée entre ces deux points
      - ***Duration*** : la durée entre ces deux points

- ***Start_location*** : localisation du premier point présenté par des paires de coordonnées GPS (latitude, longitude)
- ***End_location*** : localisation du 2ème point présenté par des paires de coordonnées GPS (latitude, longitude)

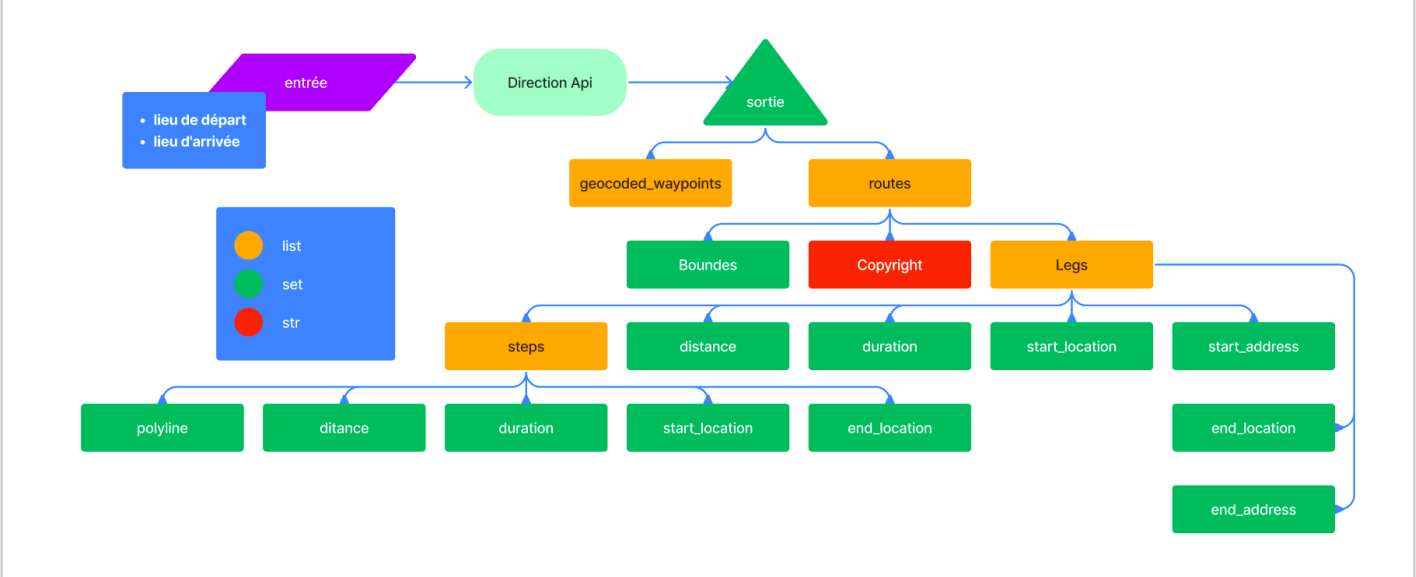


Figure 3.2 : *sortie de la demande Direction API.*

3.4. Apprentissage

Nous avons entraîné notre modèle PPO en utilisation un ensemble de données réelles sur les enregistrements des trajets de taxi dans la ville de New York pour le mois de septembre 2016. Nous avons utilisé un facteur d'escompte $\gamma = 0{,}9$. Nous avons constaté que ce prétraitement est nécessaire pour un entrainement stable. Pour les résultats de l'apprentissage, nous avons utilisé le trajet complet d'un conducteur comme un épisode pour calculer la courbe de récompense et la durée totale de l'apprentissage est de 100000 épisodes (Figure 3.3), où l'agent maximise les récompenses cumulées.

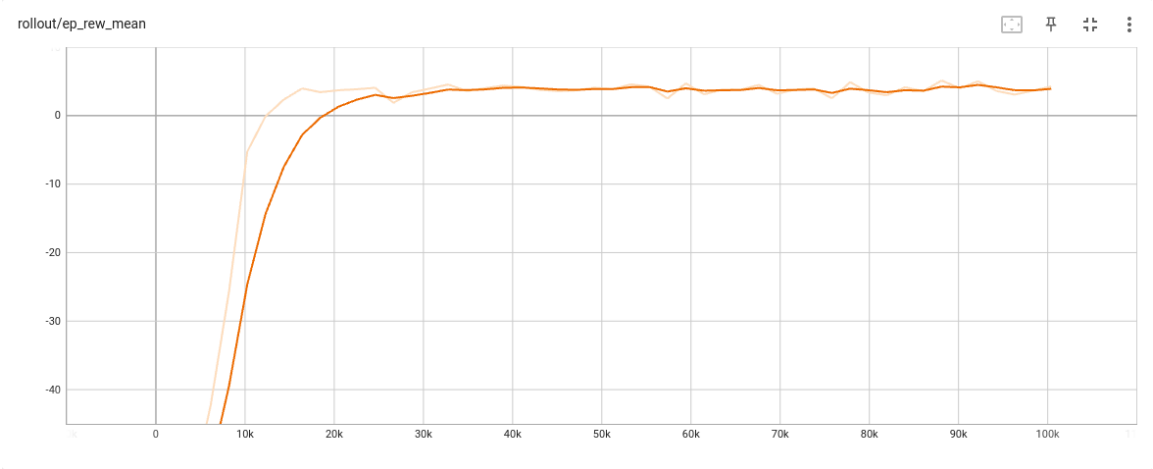


Figure 3.3 : *Courbe d'apprentissage de PPO.*

A la figure 3.4, nous présentons la courbe de perte en entropie. Nous avons constaté que cette courbe est croissante au début et ensuite elle converge vers zéro, ce qui signifie que l'agent est plus sûr des actions à entreprendre.

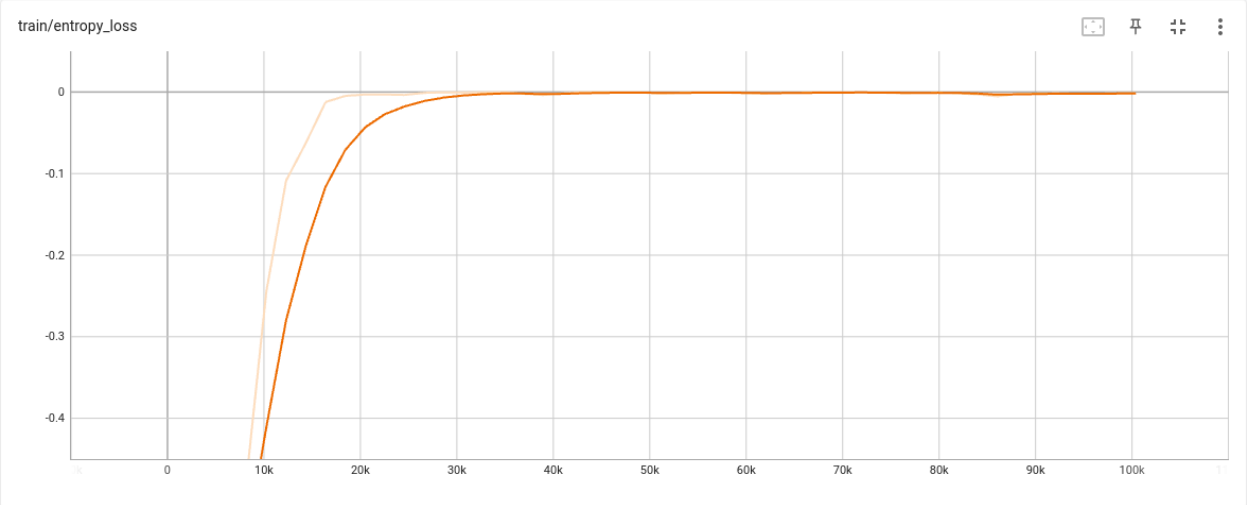


Figure 3.4 : *Courbe de la perte d'entropie.*

Avant de lancer le modèle dans le simulateur afin de tester les réactions de l'agent en temps réel, nous avons, tout d'abord, testé le modèle sur plusieurs scénarios.

L'objectif est de voir si l'agent fait le bon choix à chaque fois ou non. Nous avons, donc, testé sa prédiction quand nous lui donnons, à chaque fois, une observation différente. Ci-dessous quelques exemples :

- ***Premier cas***, il n'y a pas de demandes des passagers dans l'emplacement actuel et la destination suivante. Il faut noter que lorsque le conducteur attend au premier endroit, le temps de la prochaine destination est égal à 0 (figure 3.5).

```
obs = dict({
            'next distination time': np.array([0]),
            'rider max waiting time': np.array([0]),
            'rider current waiting time' : np.array([0]),
        })
action, _states = model.predict(obs)
print(action)
```

```
0
```

Figure 3.5 : *Cas de demande vide.*

- ***Deuxième cas***, lorsque le conducteur a reçu une demande pour la prochaine destination, mais qu'il arrivera en retard à cet endroit par rapport au temps d'attente maximum et au temps d'attente actuel de ce passager (Figure 3.6).

```
obs = dict({
            'next distination time': np.array([5]),
            'rider max waiting time': np.array([25]),
            'rider current waiting time' : np.array([24]),
        })
action, _states = model.predict(obs)
print(action)
```

```
0
```

Figure 3.6 : *Cas où l'agent arrive en retard.*

- ***Troisième cas***, lorsque le conducteur a reçu une demande pour la prochaine destination, mais dans ce cas, le conducteur arrivera au bon moment à cet endroit par rapport au temps d'attente maximum et au temps d'attente actuel de ce passager (Figure 3.7).

```
obs = dict({
            'next distination time': np.array([5]),
            'rider max waiting time': np.array([30]),
            'rider current waiting time' : np.array([25]),
        })
action, _states = model.predict(obs)
print(action)
```

```
1
```

Figure 3.7 : *Cas où l'agent viendra au bon moment.*

- ***Dans le dernier cas***, il y a deux possibilités : soit le conducteur arrivera tôt à la prochaine destination, soit il est déjà au même endroit que le passager (Figure 3.8).

```
obs = dict({
            'next distination time': np.array([5]),
            'rider max waiting time': np.array([30]),
            'rider current waiting time' : np.array([0]),
        })
action, _states = model.predict(obs)
print(action)
```

```
1
```

```
obs = dict({
            'next distination time': np.array([0]),
            'rider max waiting time': np.array([30]),
            'rider current waiting time' : np.array([0]),
        })
action, _states = model.predict(obs)
print(action)
```

```
1
```

Figure 3.8 : *Dernier cas.*

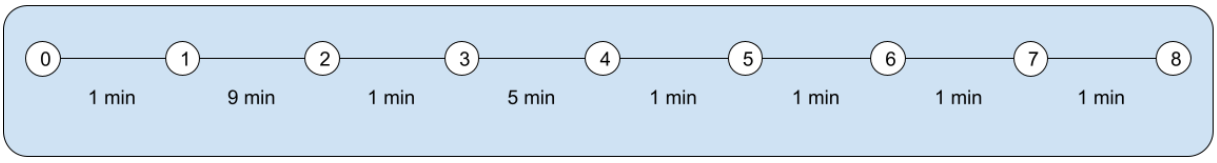


Figure 3.9 : *Trajet illustré.*

Par la suite, nous avons testé des petits scénarios illustrés dans la figure 3.9.

- *Dans le premier scénario*, disons que le conducteur et le passager présentent une offre et une demande au même endroit et au même moment, donc à ce moment-là, le conducteur acceptera la demande et n'attendra pas, le résultat est donné à la figure 3.10.

```
2020-12-12 09:00:00 Main : Rider 0 : Add his trip
2020-12-12 09:00:00 Main : Driver 0 : Add his trip
2020-12-12 09:00:00 Driver 0: trip information :
2020-12-12 09:00:00 Driver 0: start location 200 Eastern Pkwy, Brooklyn, NY 11238, USA to -->
2020-12-12 09:00:00 Driver 0: end location 1 Brookdale Plaza, Brooklyn, NY 11212, USA
2020-12-12 09:00:00 Driver 0: duration 17 mins | distance 3.7 mi
2020-12-12 09:00:00 Driver 0: i am taking with me the Rider 0
2020-12-12 09:00:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:01:00 Driver 0: Next Distination.... 9 min
2020-12-12 09:10:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:11:00 Driver 0: Next Distination.... 5 min
2020-12-12 09:16:00 Driver 0:Rider 0 arrived on his distination
2020-12-12 09:16:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:17:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:18:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:19:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:20:00 Driver 0: Finished The trip : 30.0
```

Figure 3.10 : *Scénario 1.*

- *Dans le deuxième scénario*, le conducteur et le passager postent leur offre et leur demande en même temps mais maintenant la position du conducteur est dans le 1er point et la position du passager est dans le 2ème point. Donc à ce moment, le passager n'attend qu'une minute jusqu'à ce que le conducteur arrive. Le résultat est donné à la figure 3.11.

```
Wrapping the env with a `Monitor` wrapper
Wrapping the env in a DummyVecEnv.
2020-12-12 09:00:00 Main : Rider 0 : Add his trip
2020-12-12 09:00:00 Main : Driver 0 : Add his trip
2020-12-12 09:00:00 Driver 0: trip information :
2020-12-12 09:00:00 Driver 0: start location 200 Eastern Pkwy, Brooklyn, NY 11238, USA to -->
2020-12-12 09:00:00 Driver 0: end location 1 Brookdale Plaza, Brooklyn, NY 11212, USA
2020-12-12 09:00:00 Driver 0: duration 17 mins | distance 3.7 mi
2020-12-12 09:00:00 Driver 0: i am goint to take Rider 0 in the next destination
2020-12-12 09:00:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:01:00 Driver 0:i am taking with me theRider 0
2020-12-12 09:01:00 Driver 0: Next Distination.... 9 min
2020-12-12 09:10:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:11:00 Driver 0: Next Distination.... 5 min
2020-12-12 09:16:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:17:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:18:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:19:00 Driver 0:Rider 0 arrived on his distination
2020-12-12 09:19:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:20:00 Driver 0: Finished The trip : 29.0
```

Figure 3.11 : *scenario 2.*

- *Dans le troisième scénario*, est identique que deuxième sauf que le conducteur poste son offre à 9h00 et le passager poste sa demande à 9h05.

  Dans ce cas, le conducteur attendra au premier point jusqu'à ce que le passager poste sa demande, le passager aussi dans ce scénario n'attendra qu'une minute. Le résultat est donné à la figure 3.12.

```
2020-12-12 09:00:00 Main : Driver 0 : Add his trip
2020-12-12 09:00:00 Driver 0: trip information :
2020-12-12 09:00:00 Driver 0: start location 200 Eastern Pkwy, Brooklyn, NY 11238, USA to -->
2020-12-12 09:00:00 Driver 0: end location 1 Brookdale Plaza, Brooklyn, NY 11212, USA
2020-12-12 09:00:00 Driver 0: duration 17 mins | distance 3.7 mi
2020-12-12 09:01:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:02:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:03:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:04:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:05:00 Main : Rider 0 : Add his trip
2020-12-12 09:05:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:05:00 Driver 0: i am goint to take Rider 0 in the next destination
2020-12-12 09:05:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:06:00 Driver 0:i am taking with me theRider 0
2020-12-12 09:06:00 Driver 0: Next Distination.... 9 min
2020-12-12 09:15:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:16:00 Driver 0: Next Distination.... 5 min
2020-12-12 09:21:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:22:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:23:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:24:00 Driver 0:Rider 0 arrived on his distination
2020-12-12 09:24:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:25:00 Driver 0: Finished The trip : 24.0
```

Figure 3.12 : *scenario 3.12*

- Dans le quatrième, nous ajoutons un autre passager en même temps que le premier passager au troisième point, et nous définissons ce point comme étant le point d'arrivée du premier passager.

  Donc le conducteur attend que le premier ait posté sa demande, il notifie le passager qu'il va venir le servir. Quand le conducteur dépose le premier passager, il reçoit une autre notification qu'à ce moment-là il y a une autre demande à servir qu'il accepte. Le premier passager attend 1 min, et le deuxième attend 9 min jusqu'à ce que le conducteur arrive (figure 3.13).

```
2020-12-12 09:00:00 Main : Driver 0 : Add his trip
2020-12-12 09:00:00 Driver 0: trip information :
2020-12-12 09:00:00 Driver 0: start location 200 Eastern Pkwy, Brooklyn, NY 11238, USA to -->
2020-12-12 09:00:00 Driver 0: end location 1 Brookdale Plaza, Brooklyn, NY 11212, USA
2020-12-12 09:00:00 Driver 0: duration 17 mins | distance 3.7 mi
2020-12-12 09:01:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:02:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:03:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:04:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:05:00 Main : Rider 0 : Add his trip
2020-12-12 09:05:00 Main : Rider 1 : Add his trip
2020-12-12 09:05:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:05:00 Driver 0: i am goint to take Rider 0 in the next destination
2020-12-12 09:05:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:06:00 Driver 0:i am taking with me theRider 0
2020-12-12 09:06:00 Driver 0: Next Distination.... 9 min
2020-12-12 09:15:00 Driver 0:Rider 0 arrived on his distination
2020-12-12 09:15:00 Driver 0: i am taking with me the Rider 1
2020-12-12 09:15:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:16:00 Driver 0: Next Distination.... 5 min
2020-12-12 09:21:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:22:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:23:00 Driver 0:Rider 1 arrived on his distination
2020-12-12 09:23:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:24:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:25:00 Driver 0: Finished The trip : 38.0
```

Figure 3.13 : *scenario 4.*

- *Dans le cinquième scénario*, nous voulons prouver et vérifier le dynamisme du système. Nous avons prévu que le deuxième passager ajoute sa demande au moment où il dépose le premier passager, c'est-à-dire qu'il poste sa demande à 9h15. Donc le seul qui attend est le premier passager (figure 3.14).

```
2020-12-12 09:00:00 Main : Driver 0 : Add his trip
2020-12-12 09:00:00 Driver 0: trip information :
2020-12-12 09:00:00 Driver 0: start location 200 Eastern Pkwy, Brooklyn, NY 11238, USA to -->
2020-12-12 09:00:00 Driver 0: end location 1 Brookdale Plaza, Brooklyn, NY 11212, USA
2020-12-12 09:00:00 Driver 0: duration 17 mins | distance 3.7 mi
2020-12-12 09:01:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:02:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:03:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:04:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:05:00 Main : Rider 0 : Add his trip
2020-12-12 09:05:00 Driver 0 :Still waitting.... 1 min
2020-12-12 09:05:00 Driver 0: i am goint to take Rider 0 in the next destination
2020-12-12 09:05:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:06:00 Driver 0:i am taking with me theRider 0
2020-12-12 09:06:00 Driver 0: Next Distination.... 9 min
2020-12-12 09:15:00 Main : Rider 1 : Add his trip
2020-12-12 09:15:00 Driver 0:Rider 0 arrived on his distination
2020-12-12 09:15:00 Driver 0: i am taking with me the Rider 1
2020-12-12 09:15:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:16:00 Driver 0: Next Distination.... 5 min
2020-12-12 09:21:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:22:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:23:00 Driver 0:Rider 1 arrived on his distination
2020-12-12 09:23:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:24:00 Driver 0: Next Distination.... 1 min
2020-12-12 09:25:00 Driver 0: Finished The trip : 53.0
```

Figure 3.14 : *scenario 5.*

3.5. Simulateur

Afin d'évaluer la modélisation que nous avons proposée, nous avons développé un simulateur (figure 3.15) de répartition plus complexe et plus réaliste. Dans notre application, les résultats du simulateur hors ligne sont parfois aussi importants que les expériences en ligne, étant donné que le système en ligne est hautement dynamique.

Pour initialiser l'environnement, nous lançons la simulation de 08:00 à 21:00.

Nous avons fixé l'intervalle de réception de nouvelles demandes et offres par le système à 4 minutes ($\Delta t = 4\,min$). Ainsi, chaque 4 minutes, le système reçoit un nombre aléatoire entre 1 et 5 d'offres et de demandes.

Les emplacements initiaux de ces offres et demandes sont choisis au hasard dans l'ensemble de données.

Nous rappelons que nous considérons un trajet inclusif du passager dans le trajet du conducteur.

Nous avons donné le temps d'attente de chaque conducteur et passager d'une manière aléatoire et entre 15 à 30 minutes.

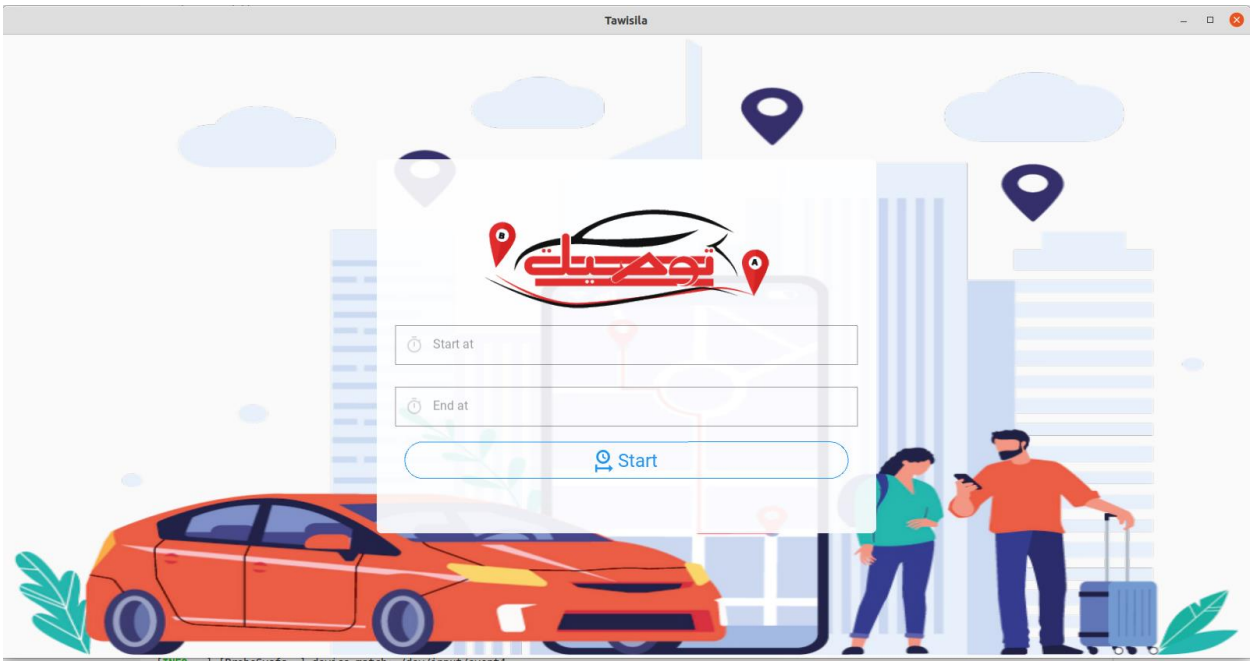


***Figure 3.15** : L'interface de notre simulateur.*

Une demande du passager peut être abandonnée dans le processus de mise en file d'attente. La raison en est l'impatience des passagers dont le temps d'attente pour être mis en correspondance est long. Dans notre simulation, nous supposons que chaque passager dispose d'un temps d'appariement maximal et qu'un passager abandonnera la file d'attente si son temps d'attente cumulé est supérieur au temps d'appariement maximal. Ce processus a été mis en place car notre objectif est de minimiser le temps d'attente du passager.

Nous notons que l'inadéquation entre l'offre et la demande est reflétée dans notre simulateur par la mesure du taux d'acceptation. Cette métrique est calculée comme le nombre de trajets acceptés divisé par le nombre total de demandes par heure.

Lorsque on lance le simulateur, et en respectant les paramètres mentionnés ci-dessus, le simulateur affiche tous les conducteurs, les passagers, et les contraintes sur les trajets de chacun d'entre eux (figure 3.16). Les résultats du simulateur sont présentés dans le tableau 3.1.

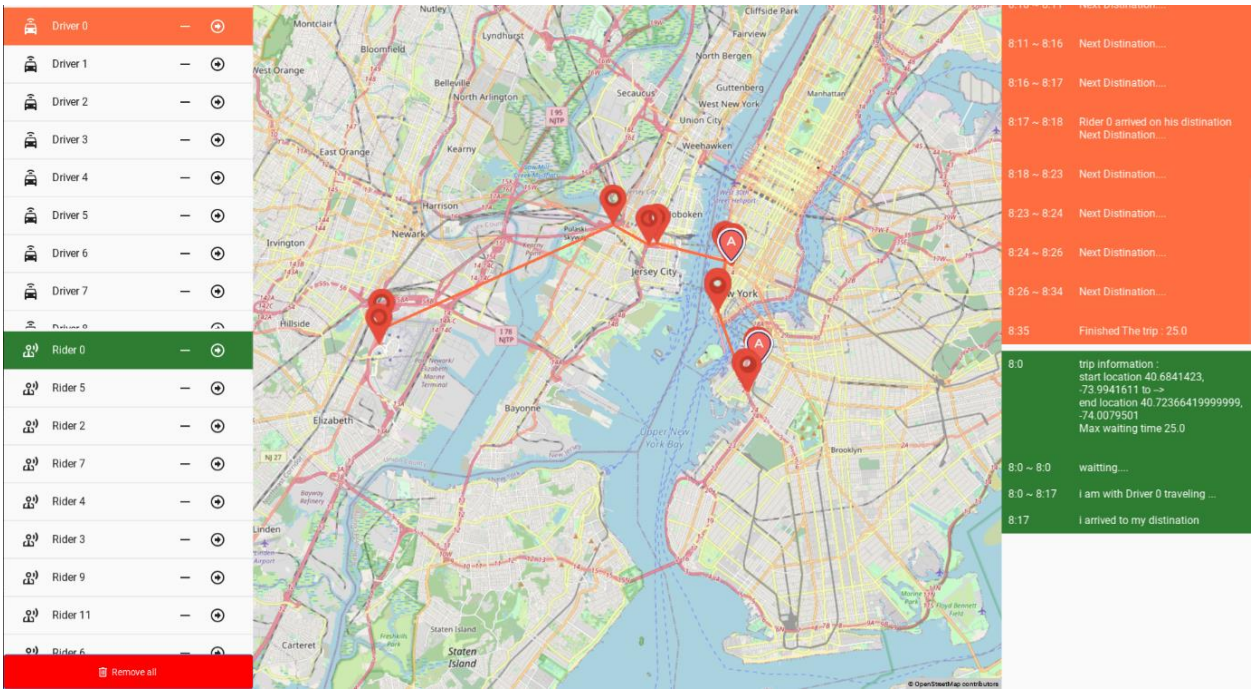


Figure 3.16 : *Interface de visualisation des résultats.*

Tableau 3.1 : *Résultats de la simulation.*

HEURE	**08:00 - 09:00**	**09:00 - 10:00**	**10:00 - 11:00**	**11:00 - 12:00**	**12:00 - 13:00**	**13:00 - 14:00**	**14:00 - 15:00**
N° CONDUCTEURS	50	96	136	186	235	276	328
N° REQUÊTES EFFECTUÉES	30	74	116	156	195	233	273
N° DEMANDES ACCEPTÉES	24	50	80	105	133	159	188
TAUX	0,8	0,68	0,69	0,67	0,68	0,68	0,69
HEURE	**15:00 - 16:00**	**16:00 - 17:00**	**17:00 - 18:00**	**18:00 - 19:00**	**19:00 - 20:00**	**20:00 - 21: 00**	**21:00 - 21:30**
N° CONDUCTEURS	372	421	469	512	561	612	615
N° REQUÊTES EFFECTUÉES	326	377	427	478	523	564	582
N° DEMANDES ACCEPTÉES	218	251	284	310	341	371	384
TAUX	0,67	0,67	0,67	0,65	0,65	0,66	0,66
RÉCOMPENSE TOTALE (TEMPS PROFITABLE)				8145 MIN			
TEMPS D'ATTENTE TOTAL				5115 MIN			

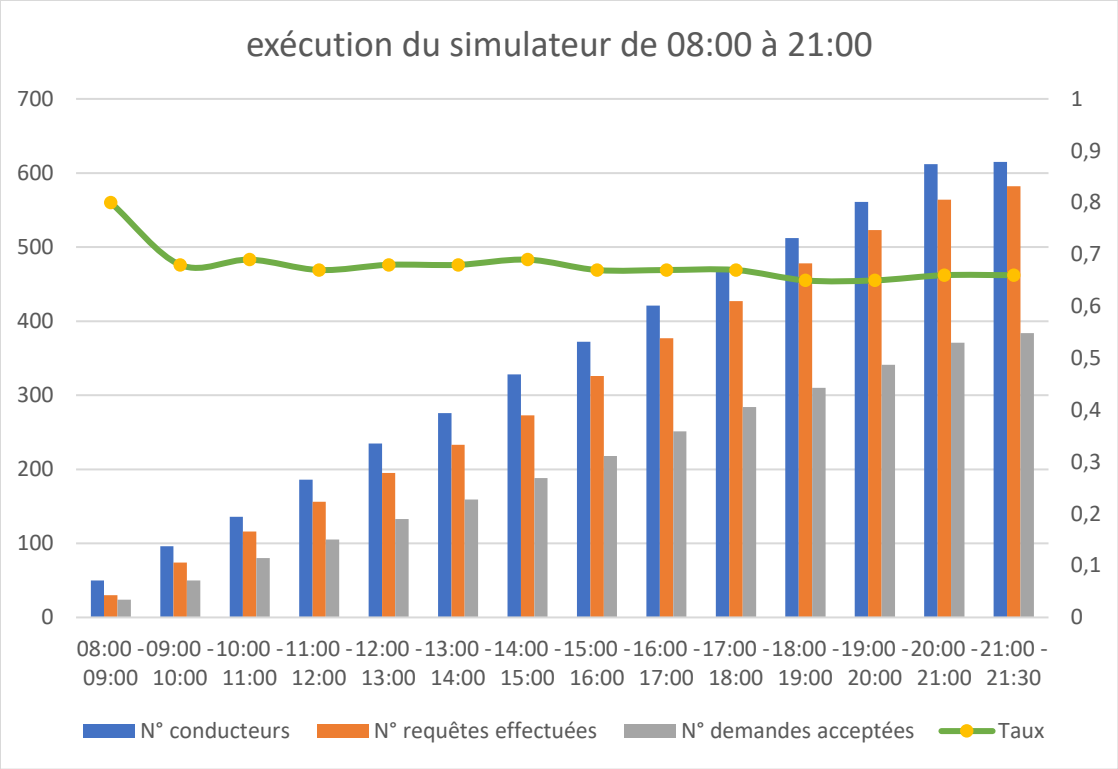


Figure 3.17 : *Visualisation des résultats de la simulation.*

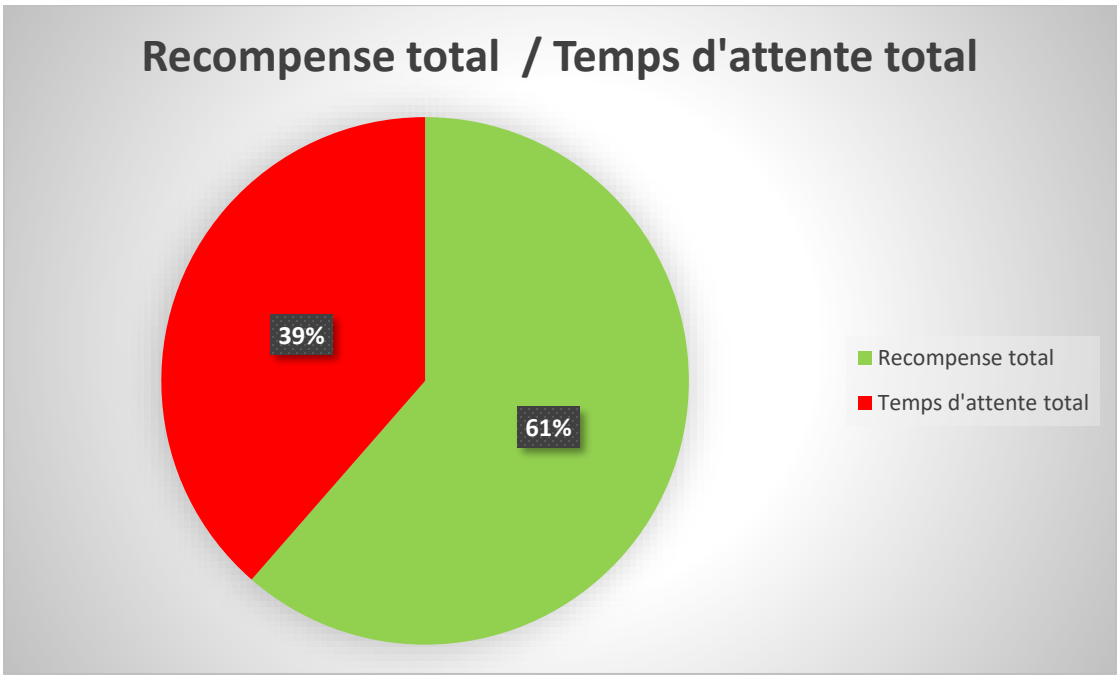


Figure 3.18 : *Récompense totale vs Temps d'attente total.*

Les résultats de la simulation présentés dans le tableau 3.1 ainsi qu'illustrés dans les figures 3.17 et 3.18 démontre la performance de notre approche. En effet, la récompense totale du système pour une simulation de 8 :00 à 21 :00 est de 8145 minutes. Cette récompense reflète le gain en temps (temps profitable) qui est plus grand que le temps d'attente totale (5115 minutes).

3.6. Expérimentation

Pour fournir une compréhension plus complète de l'efficacité de la méthode proposée, nous avons évalué notre système sur six environnements correspondant à différentes paramétrisations.

Tous les scénarios commencent à 08:00 et se terminent à 11:00. Les emplacements initiaux des offres et des demandes sont choisis au hasard dans l'ensemble de données.

Nous présentons dans le tableau 3.2, les paramètres des différents scénarios testés.

Tableau 3.2 : *Les paramètres des scénarios testés.*

	Δt	Nombre d'offres	Nombre de demandes	Temps d'attente
Scénario 1	4 min	5	Random [1,5]	Random [15,30]
Scénario 2	4 min	5	5	Random [15,30]
Scénario 3	1 min	Random [1,5]	Random [1,5]	Random [15,30]
Scénario 4	4 min	Random [1,5]	Random [1,5]	15 min
Scénario 5	4 min	Random [1,5]	Random [1,5]	30 min
Scénario 6	1 min	1	1	Random [15,30]

Les résultats de l'exécution de ces scénarios sont présentés dans les tableaux 3.3, 3.4, 3.5, 3.6, 3.7 et 3.8. Les visualisations de ces résultats sont données aux figures 3.19, 3.20, 3.21, 3.22, 3.23, 3.24.

Tableau 3.3 : *les résultats d'exécution du scenario 1.*

HEURE	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 11:30
N° CONDUCTEURS	75	150	225	230
N° REQUÊTES EFFECTUÉES	31	78	125	141
N° DEMANDES ACCEPTÉES	27	63	103	114
TAUX	0,87	0,81	0,82	0,81
RÉCOMPENSE TOTALE (TEMPS PROFITABLE)	4305 min			
TEMPS D'ATTENTE TOTAL	785 min			

Tableau 3.4 : *les résultats d'exécution du scenario 2.*

HEURE	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 11:30
N° CONDUCTEURS	75	150	225	230
N° REQUÊTES EFFECTUÉES	56	129	199	230
N° DEMANDES ACCEPTÉES	45	98	151	171
TAUX	0,8	0,76	0,76	0,74
RÉCOMPENSE TOTALE (TEMPS PROFITABLE)	3756 min			
TEMPS D'ATTENTE TOTAL	1579 min			

Tableau 3.5 : *les résultats d'exécution du scenario 3.*

HEURE	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 11:30
N° CONDUCTEURS	168	356	534	536
N° REQUÊTES EFFECTUÉES	142	330	491	536
N° DEMANDES ACCEPTÉES	101	230	346	362
TAUX	0,71	0,7	0,7	0,68
RÉCOMPENSE TOTALE (TEMPS PROFITABLE)	6990 min			
TEMPS D'ATTENTE TOTAL	4730 min			

Tableau 3.6 : *les résultats d'exécution du scenario 4.*

HEURE	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 11:30
N° CONDUCTEURS	43	85	123	128
N° REQUÊTES EFFECTUÉES	35	80	120	143
N° DEMANDES ACCEPTÉES	19	45	70	83
TAUX	0,54	0,56	0,58	0,58
RÉCOMPENSE TOTALE (TEMPS PROFITABLE)	1121 min			
TEMPS D'ATTENTE TOTAL	1024 min			

Tableau 3.7 : *les résultats d'exécution du scenario 5.*

HEURE	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 11:30
N° CONDUCTEURS	43	88	138	143
N° REQUÊTES EFFECTUÉES	24	65	106	128
N° DEMANDES ACCEPTÉES	21	49	76	85
TAUX	0,88	0,75	0,72	0,66
RÉCOMPENSE TOTALE (TEMPS PROFITABLE)	2417 min			
TEMPS D'ATTENTE TOTAL	1423 min			

Tableau 3.8 : *les résultats d'exécution du scenario 6.*

HEURE	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 11:30
N° CONDUCTEURS	60	120	180	181
N° REQUÊTES EFFECTUÉES	45	102	159	181
N° DEMANDES ACCEPTÉES	41	91	143	165
TAUX	0,91	0,89	0,9	0,91
RÉCOMPENSE TOTALE (TEMPS PROFITABLE)	3134 min			
TEMPS D'ATTENTE TOTAL	2296 min			

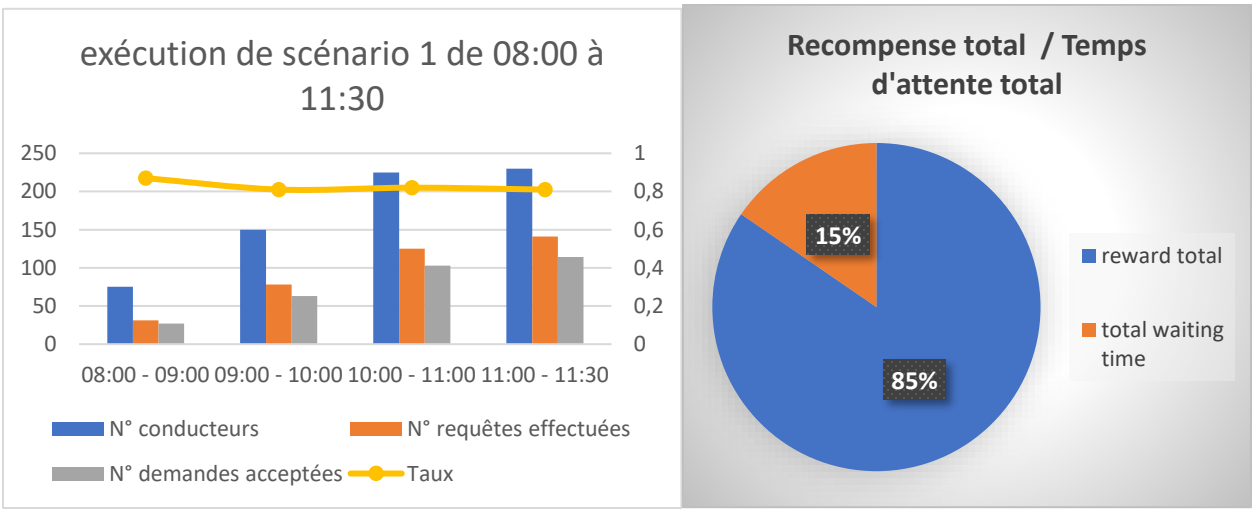


Figure 3.19 : *Résultats du scénario 1.*

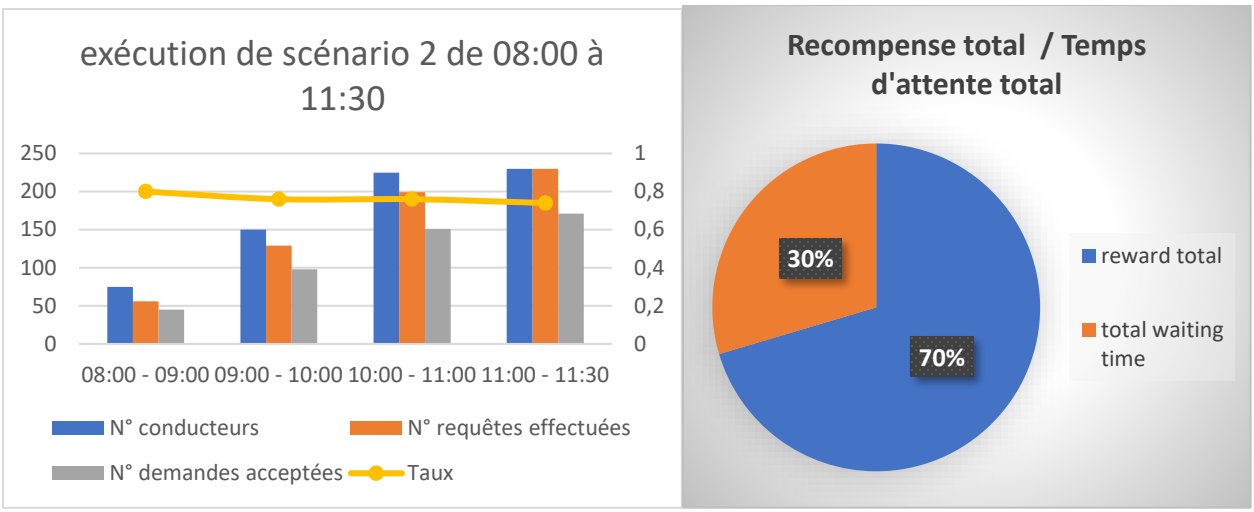


Figure 3.20 : *Résultat du scénario 2.*

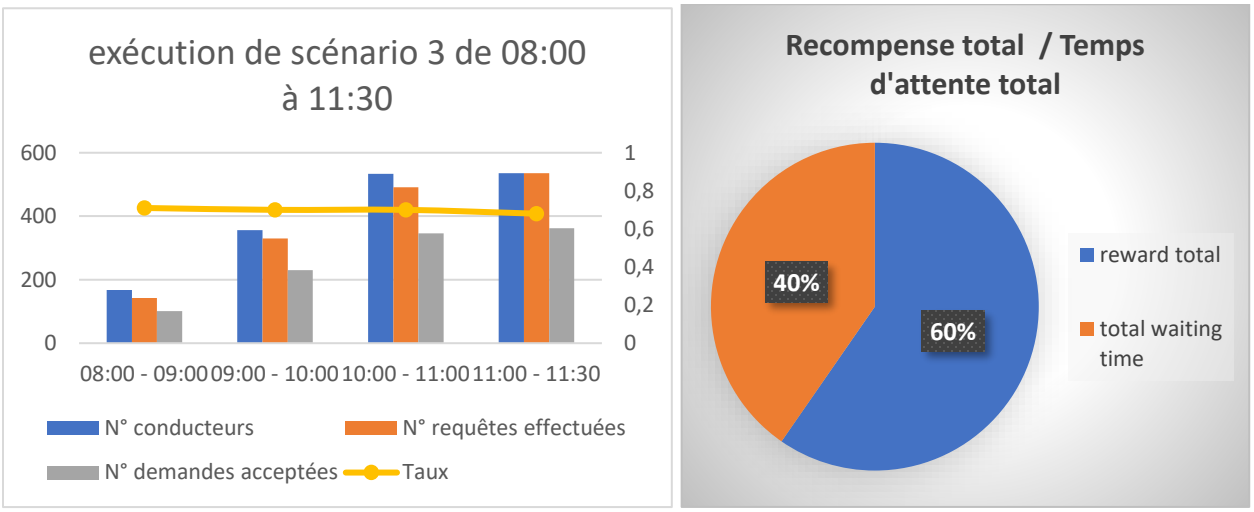


Figure 3.21 : *Résultat du scénario 3.*

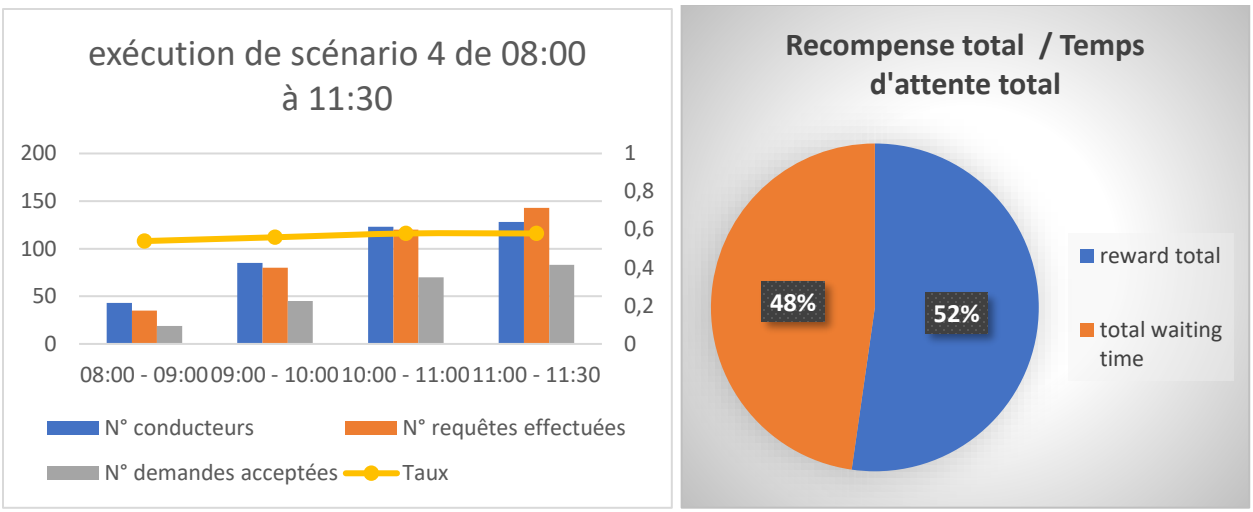


Figure 3.22 : *Résultat du scénario 4.*

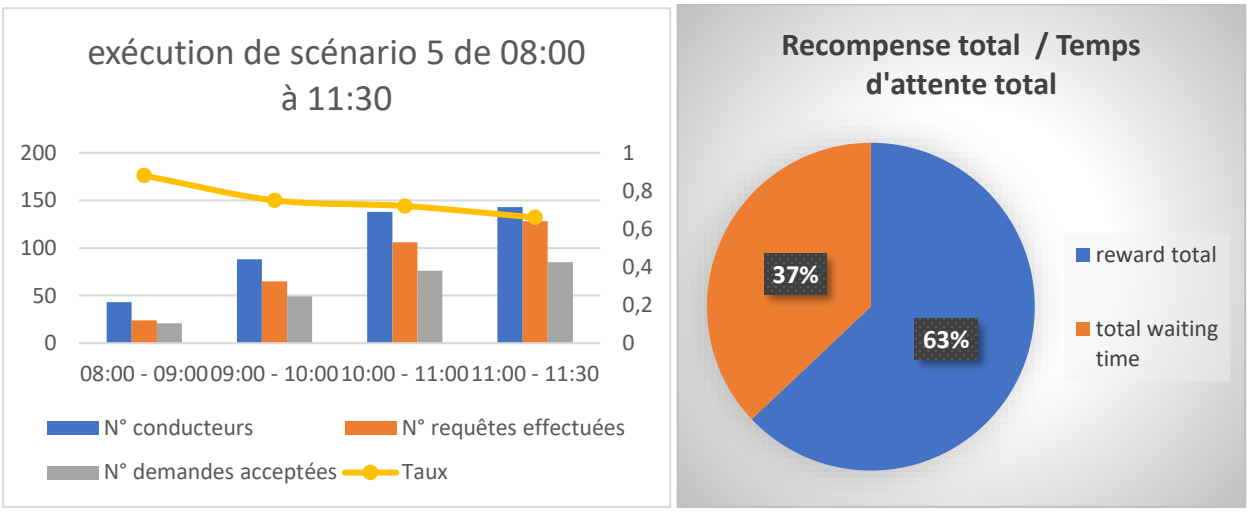


Figure 3.23 : *Résultat du scénario 5.*

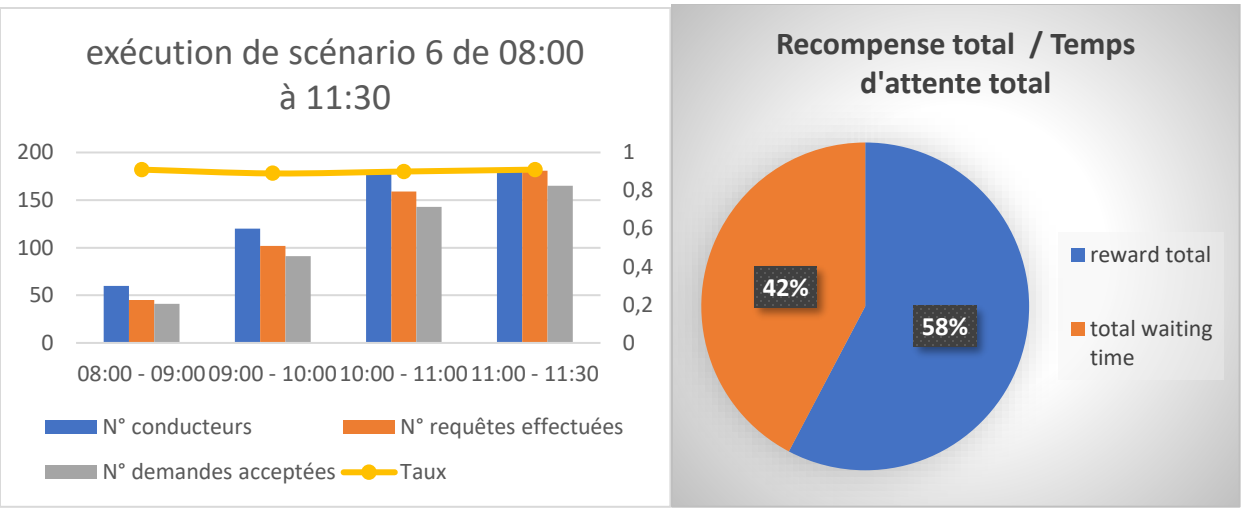


Figure 3.24 *: Résultat du scénario 6.*

Sur le scénario 1 où nous avons fixé l'intervalle d'arrivée des conducteurs et des passagers à 4 minutes, le nombre d'offre à 5 et le nombre de demandes était un nombre aléatoire entre 1 et 5 avec temps d'attente entre 15 et 30 minutes (aléatoire), le temps d'attente total était 785 minutes et le taux moyen d'acceptation de demande était à 0,82.

Dans le deuxième scénario, nous avons fixé le nombre de demandes et d'offres à 5, le temps d'attente a augmenté (1579 minutes) et le taux moyen d'acceptation était 0,76.

Dans les scénarios 3 et 6, nous avons serré l'intervalle d'arrivées de demandes et d'offres (1 minutes) avec une randomisation sur le nombre d'offre et de demande pour le scénario 3 et une demande et une offre pour le scénario 6. Le taux moyen d'acceptation du scénario 3 = 0,69 face à 0,90 pour le scénario 6 où à chaque minute le système reçoit une offre et une demande.

Dans les scénarios 4 et 5, nous avons joué sur le temps d'attente où on la réduit à 15 minutes pour le scénario 4 et on l'augmenté à 30 minutes pour le scénario 5. Les deux scénarios ont présenté une réduction dans le taux moyen d'acceptation (0,56 et 0,75) mais qui reste toujours acceptable.

Nous avons prouvé à travers ces scénarios la stabilité et surtout la robustesse de notre approche face à différents changements de paramètres. Ces résultats viennent aussi confirmer et justifier notre choix du temps d'attente comme paramètre tranchant dans la décision.

3.7. Conclusion

Dans ce chapitre, nous avons présenté les détails d'implémentation de notre approche et de notre simulateur. Nous avons exposé les différents résultats obtenus à chaque phase de réalisation ainsi que les expérimentations réalisées afin de tester les performances de notre système. Ces résultats sont prometteurs et viennent appuyer nos choix et notre modélisation.

Conclusion générale

Le covoiturage dynamique se caractérise par une grande souplesse d'utilisation et moins d'interdépendance que le covoiturage statique. Il s'agit de pouvoir trouver un trajet partagé dans un délai rapide (moins d'une demi-heure par exemple) pour un trajet donné et en fonction de la position des véhicules des conducteurs potentiels.

C'est un processus automatisé dans lequel un fournisseur de services met en relation des conducteurs et des passagers ayant des itinéraires et des horaires similaires pour partager un trajet à court préavis dans un véhicule personnel. Les systèmes de covoiturage sont naturellement dynamiques, leur complexité repose principalement sur l'appariement d'individus soumis à des contraintes spatio-temporelles, qui doivent être précisées de part et d'autre – c'est-à-dire conducteurs et passagers – avant que le trajet souhaité ne soit établi et exécuté.

L'étude approfondie de ce domaine nous a amené à constater que l'apprentissage par renforcement présente un bon candidat pour résoudre ce problème de décision.

Dans ce travail, nous avons proposé une modélisation des composants du problème d'appariement dynamique à contraintes par un processus décisionnel de Markov qui représente le fondement théorique de l'apprentissage par renforcement. La modélisation proposée est axée temps et l'objectif à optimiser est la minimisation du temps d'attente. La minimisation de ce paramètre a pour effet de satisfaire les passagers et ainsi maximiser le nombre de demandes achevées.

Cette proposition était validée sur la base d'un simulateur développé afin de pouvoir assurer l'aspect dynamique du système. Les résultats obtenus à travers les différentes expérimentations réalisées ont révélé l'efficacité de notre système et surtout sa robustesse et sa capacité à répondre à différentes requêtes sous différentes contraintes. Ces résultats sont venus prouver l'hypothèse que nous avons supposée par rapport au *temps* qui est un paramètre tranchant dans la décision.

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

Nous visons d'explorer encore les approches d'apprentissage par renforcement afin de faire apprendre à l'agent de choisir le meilleur passager à servir parmi un ensemble de passagers

potentiels. De plus, nous sommes intéressés par l'ajout d'une contrainte (plusieurs conducteurs - plusieurs passagers) à notre approche, ce qui ajoute des objectifs supplémentaires à optimiser. Nous visons aussi à déployer la méthode proposée dans l'application mobile que nous avons développé auparavant (PFE Licence) afin de l'évaluer dans le monde réel.

Références bibliographiques

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.

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.

Alisoltani, N., Leclercq, L., & Zargayouna, M. (2019, September). Real-time ride-sharing systems performance considering network congestion. In *hEART 2019, 8th Symposium of the European Association for Research in Transportation* (p. 6p).

Armant, V., & Brown, K. N. (2014, November). Minimizing the driving distance in ride sharing systems. In 2014 IEEE 26th International Conference on Tools with Artificial Intelligence (pp. 568-575). IEEE.

Armant, V., & Brown, K. N. (2020). Fast optimised ridesharing: Objectives, reformulations and driver flexibility. Expert Systems with Applications, 141, 112914.

Baldacci, R., Maniezzo, V., & Mingozzi, A. (2004). An exact method for the car pooling problem based on lagrangean column generation. Operations research, 52(3), 422-439.

Ben Cheikh, S., Hammadi, S., & Tahon, C. (2015). Agent-based evolutionary cooperative approach for dynamic multi-hop ridematching problem. IFAC-PapersOnLine, 48(3), 887-892.

Ben Cheikh, S., Tahon, C., & Hammadi, S. (2017). An evolutionary approach to solve the dynamic multihop ridematching problem. Simulation, 93(1), 3-19.

Ben Cheikh, S., & Hammadi, S. (2016). Multi-Hop Ridematching optimization problem: Intelligent chromosome agent-driven approach. Expert Systems with Applications, 62,161-176.

Calvo, R. W., de Luigi, F., Haastrup, P., & Maniezzo, V. (2004). A distributed geographic information system for the daily carpooling problem. Computers & Operations Research, 31(13), 2263-2278.

Cheikh, S. B. (2016). Optimisation avancée au service du covoiturage dynamique (Doctoral dissertation, Ecole centrale de Lille.

Cheikh-Graiet, S. B., Dotoli, M., & Hammadi, S. (2020). A Tabu Search based metaheuristic for dynamic carpooling optimization. Computers & Industrial Engineering, 140, 106217.

Chen, H., Jiao, Y., Qin, Z., Tang, X., Li, H., An, B., ... & Ye, J. (2019, November). Inbede: integrating contextual bandit with td learning for joint pricing and dispatch of ride-hailing platforms. In 2019 IEEE International Conference on Data Mining (ICDM) (pp. 61-70). IEEE.

Cici, B., Markopoulou, A., & Laoutaris, N. (2015, November). Designing an on-line ride-sharing system. In Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems (pp. 1-4).

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

Filcek, G., Hojda, M., & Żak, J. (2017). A heuristic algorithm for solving a Multiple Criteria Carpooling Optimization (MCCO) problem. Transportation Research Procedia, 27, 656-663.

Fu, X., Zhang, C., Lu, H., & Xu, J. (2019). Efficient matching of offers and requests in social-aware ridesharing. GeoInformatica, 23(4), 559-589.

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.

Gagnon, M. (2016). Algorithme de jumelage multimodal pour le covoiturage (Doctoral dissertation, Ecole Polytechnique, Montréal (Canada).

Gu, Q. P., Liang, J. L., & Zhang, G. (2018). Algorithmic analysis for ridesharing of personal vehicles. Theoretical Computer Science, 749, 36-46.

Haliem, M., Aggarwal, V., & Bhargava, B. (2020, November). AdaPool: An adaptive model-free ride-sharing approach for dispatching using deep reinforcement learning. In Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (pp. 304-305).

Herbawi, W., & Weber, M. (2012, June). The ridematching problem with time windows in dynamic ridesharing: A model and a genetic algorithm. In 2012 IEEE Congress on Evolutionary Computation (pp. 1-8). IEEE.

Herbawi, W. (2013). Solving the ridematching problem in dynamic ridesharing (Doctoral dissertation, Universität Ulm).

Hu, M., & Zhou, Y. (2022). Dynamic type matching. Manufacturing & Service Operations Management, 24(1), 125-142.

Huang, Y., Jin, R., Bastani, F., & Wang, X. S. (2013). Large scale real-time ridesharing with service guarantee on road networks. arXiv preprint arXiv:1302.6666.

Huang, S. C., Jiau, M. K., & Lin, C. H. (2014). A genetic-algorithm-based approach to solve carpool service problems in cloud computing. IEEE Transactions on intelligent transportation systems, 16(1), 352-364.

Huang, S. C., Jiau, M. K., & Liu, Y. P. (2018). An ant path-oriented carpooling allocation approach to optimize the carpool service problem with time windows. IEEE Systems Journal, 13(1), 994-1005.

Jeribi, K. (2012). Conception et réalisation d'un système de gestion de véhicules partagés: de la multimodalité vers la co-modalité (Doctoral dissertation, Ecole centrale de Lille).

Jin, J., Zhou, M., Zhang, W., Li, M., Guo, Z., Qin, Z., ... & Ye, J. (2019, November). Coride: joint order dispatching and fleet management for multi-scale ride-hailing platforms. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (pp. 1983-1992).

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

Jintao, K. E., 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.

Jung, J., Jayakrishnan, R., & Park, J. Y. (2016). Dynamic shared-taxi dispatch algorithm with hybrid-simulated annealing. Computer-Aided Civil and Infrastructure Engineering, 31(4), 275-291.

Liu, M., Luo, Z., & Lim, A. (2015). A branch-and-cut algorithm for a realistic dial-a-ride problem. Transportation Research Part B : Methodological, 81, 267-288.

Lu, W., Liu, L., Wang, F., Zhou, X., & Hu, G. (2020). Two-phase optimization model for ride-sharing with transfers in short-notice evacuations. Transportation research part C: emerging technologies, 114, 272-296.

Ma, C., He, R., & Zhang, W. (2018). Path optimization of taxi carpooling. PLoS One, 13(8), e0203221.

Manna, C., & Prestwich, S. (2014, November). Online stochastic planning for taxi and ridesharing. In 2014 IEEE 26th international conference on tools with artificial intelligence (pp. 906-913). IEEE.

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.

Masmoudi, M. A., Hosny, M., Demir, E., & Pesch, E. (2020). Hybrid adaptive large neighborhood search algorithm for the mixed fleet heterogeneous dial-a-ride problem. Journal of Heuristics, 26(1), 83-118.

Masson, R., Lehuédé, F., & Péton, O. (2014). The dial-a-ride problem with transfers. Computers & Operations Research, 41, 12-23.

Molenbruch, Y., Braekers, K., & Caris, A. (2017). Typology and literature review for dial-a-ride problems. Annals of Operations Research, 259(1), 295-325.

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.

Özkan, E., & Ward, A. R. (2020). Dynamic matching for real-time ride sharing. Stochastic Systems, 10(1), 29-70.

Prieto, M., Baltas, G., & Stan, V. (2017). Car sharing adoption intention in urban areas: What are the key sociodemographic drivers?. Transportation Research Part A: Policy and Practice, 101, 218-227.

Puterman, M. L. (2014). Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons.

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, September). Reinforcement learning for ridesharing: A survey. In 2021 IEEE International Intelligent Transportation Systems Conference (ITSC) (pp. 2447-2454). IEEE.

Ritzinger, U., Puchinger, J., & Hartl, R. F. (2016). A survey on dynamic and stochastic vehicle routing problems. International Journal of Production Research, 54(1), 215-231.

Santos, D. O., & Xavier, E. C. (2013, June). Dynamic taxi and ridesharing: A framework and heuristics for the optimization problem. In Twenty-Third International Joint Conference on Artificial Intelligence.

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.

Sarriera, J. M., Álvarez, G. E., Blynn, K., Alesbury, A., Scully, T., & Zhao, J. (2017). To share or not to share: Investigating the social aspects of dynamic ridesharing. Transportation Research Record, 2605(1), 109-117.

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

Shen, B., Huang, Y., & Zhao, Y. (2016). Dynamic ridesharing. Sigspatial Special, 7(3), 3-10.

Silwal, S., Gani, M. O., & Raychoudhury, V. (2019, June). A survey of taxi ride sharing system architectures. In 2019 IEEE International Conference on Smart Computing (SMARTCOMP) (pp. 144-149). IEEE.

Singh, A., Al-Abbasi, A., & Aggarwal, V. (2019, December). A reinforcement learning based algorithm for multi-hop ride-sharing: Model-free approach. In Neural Information Processing Systems (Neurips) Workshop.

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.

Song, J., Cho, Y. J., Kang, M. H., & Hwang, K. Y. (2020). An application of reinforced learning-based dynamic pricing for improvement of ridesharing platform service in seoul. Electronics, 9(11), 1818.

Stach, C., & Brodt, A. (2011, June). vHike-a dynamic ride-sharing service for smartphones. In 2011 IEEE 12th International Conference on Mobile Data Management (Vol. 1, pp. 333-336). IEEE.

Stiglic, M., Agatz, N., Savelsbergh, M., & Gradisar, M. (2016). Making dynamic ride-sharing work: The impact of driver and rider flexibility. Transportation Research Part E: Logistics and Transportation Review, 91, 190-207.

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction, 2nd Edition. MIT press.

Swinnen, Gérard. (2012). Apprendre à programmer avec Python 3.

Tang, X., Qin, Z., Zhang, F., Wang, Z., Xu, Z., Ma, Y., ... & Ye, J. (2019, July). A deep value-network based approach for multi-driver order dispatching. In Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining (pp. 1780-1790).

Turan, B., Pedarsani, R., & Alizadeh, M. (2020). Dynamic pricing and fleet management for electric autonomous mobility on demand systems. Transportation Research Part C: Emerging Technologies, 121, 102829.

Wang, X., Agatz, N., & Erera, A. (2018). Stable matching for dynamic ride-sharing systems. Transportation Science, 52(4), 850-867.

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

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

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

Yan, C., Zhu, H., Korolko, N., & Woodard, D. (2020). Dynamic pricing and matching in ride-hailing platforms. Naval Research Logistics (NRL), 67(8), 705-724.

Yang, H., Qin, X., Ke, J., & Ye, J. (2020). Optimizing matching time interval and matching radius in on-demand ride-sourcing markets. Transportation Research Part B: Methodological, 131, 84-105.

Zhou, M., Jin, J., Zhang, W., Qin, Z., Jiao, Y., Wang, C., ... & Ye, J. (2019, November). Multi-agent reinforcement learning for order-dispatching via order-vehicle distribution matching. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (pp. 2645-2653).

Webographie

[1] «python,» [En ligne]. Available: https://python.doctor. [Accès le 12 05 2022].

[2] «anaconda,» [En ligne]. Available: https://docs.anaconda.com/anaconda/navigator/. [Accès le 12 05 2022].

[3] «jupyter,» [En ligne]. Available: https://jupyter.readthedocs.io/en/latest/projects/architecture/content-architecture.html. [Accès le 12 05 2022].

[4] «visualstudio,» [En ligne]. Available: https://code.visualstudio.com/docs. [Accès le 15 05 2022].

[5] «spyder,» [En ligne]. Available: https://docs.spyder-ide.org/current/index.html. [Accès le 15 05 2022].

[6] «Pandas,» [En ligne]. Available: https ://en.wikipedia.org/wiki/Pandas . [Accès le 12 05 2022].

[7] «Numpy,» [En ligne]. Available: https ://en.wikipedia.org/wiki/Numpy . [Accès le 12 05 2022].

[8] «kivymd,» [En ligne]. Available: https://kivymd.readthedocs.io/en/latest/. [Accès le 12 05 2022].

[9] «gym,» [En ligne]. Available: https://gym.openai.com/docs/. [Accès le 12 05 2022].

[10] «stable-baselines3,» [En ligne]. Available: https://stable-baselines3.readthedocs.io/en/master/index.html. [Accès le 12 05 2022].

[11] «tensorboard,» [En ligne]. Available: https://www.tensorflow.org/tensorboard. [Accès le 12 05 2022].

[12] «directions,» [En ligne]. Available: https://developers.google.com/maps/documentation/directions/overview. [Accès le 12 05 2022].

[13] «Figma,» [En ligne]. Available: https://en.wikipedia.org/wiki/Figma_(software). [Accès le 30 05 2020].

[14] «kaggle,» 2017. [En ligne]. Available: https://www.kaggle.com/datasets/fivethirtyeight/uber-pickups-in-new-york-city. [Accès le 15 05 2022].

[15] «medium,». [En ligne]. Available: https://medium.com/aureliantactics/understanding-ppo-plots-in-tensorboard-cbc3199b9ba2. [Accès le 15 05 2022].

[16] «keras,». [En ligne]. Available: https://keras.io/examples/rl/ppo_cartpole/. [Accès le 01 06 2022].

[17] «stable-baselines3,». [En ligne]. Available: https://stable-baselines3.readthedocs.io/en/master/modules/ppo.html. [Accès le 01 06 2022].

[18] «medium,». [En ligne]. Available https://medium.com/intro-to-artificial-intelligence/proximal-policy-optimization-ppo-a-policy-based-reinforcement-learning-algorithm-3cf126a7562d [Accès le 15 05 2022].