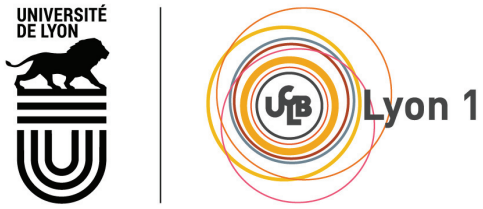
Régulation de trafic urbain multimodal : une modélisation multi-agents

Matthis Gaciarz



N° d'ordre NNT : 2016LYSE1281

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de
l'Université Claude Bernard Lyon 1

École Doctorale ED 512
École Doctorale Informatique et Mathématiques

Spécialité de doctorat : informatique

Soutenue publiquement le 05/12/2016, par :
Matthis Gaciarz

Régulation de trafic urbain multimodal : une modélisation multi-agents

---

Devant le jury composé de :

René Mandiau, Professeur des Universités, UVHC, Valenciennes	Rapporteur
Leila Merghem, Maître de Conférence, UTT, Troyes	Rapporteure
Saïd Mammar, Professeur des Universités, UEVE, Evry	Examinateur
Samir Aknine, Professeur des Universités, UCBL, Lyon	Directeur de thèse
Neila Bhouri, Chargée de Recherche, IFSTTAR, Marne la Vallée	Co-directrice de thèse

Matthis Gaciarz
Régulation de trafic urbain multimodal : une modélisation multi-agents
Informatique, 5 décembre 2016
Rapporteurs : René Mandiau et Leila Merghem
Examinateur : Saïd Mammar
Directeurs de thèse : Samir Aknine et Neila Bhouri

LIRIS-CNRS
Laboratoire d'InfoRmatique en Image et Systèmes d'information
UMR 5205 CNRS
Universtité Claude Bernard Lyon 1
Bâtiment Nautibus
Campus de la Doua
25 avenue Pierre de Coubertin
69622 Villeurbanne Cedex, France

IFSTTAR - GRETTIA
Génie des Réseaux de Transports Terrestres et Informatique Avancée
14-20 Boulevard Newton
Cité Descartes, Champs sur Marne
F-77447 Marne la Vallée Cedex 2, France

Cette thèse a été financée par une Allocation Doctorale de Recherche de **La Région Rhône-Alpes** *dans le cadre du dispositif Communautés de Recherche Académique ARC 7 : Innovations, Mobilités, Territoires et Dynamiques Urbaines*

Résumé

Depuis plusieurs décennies, la congestion urbaine est à l'origine de différentes nuisances et dégrade la qualité de vie des habitants des villes. Plusieurs méthodes sont utilisées pour diminuer la congestion urbaine, notamment la régulation du trafic et la valorisation des transports en commun. Depuis les années 1990, l'utilisation d'outils issus de l'intelligence artificielle, et en particulier des méthodes distribuées et des systèmes multi-agents, a permis de concevoir de nouvelles méthodes de régulation du trafic. En effet, ces méthodes permettent de prendre en compte la complexité et la richesse des problèmes liés au trafic par la distribution. Parallèlement, l'amélioration des capacités de communication des véhicules et des infrastructures et l'arrivée de voitures autonomes permettent d'envisager de nouvelles approches en matière de régulation, notamment dans le cadre des systèmes coopératifs dans lequel les véhicules et les infrastructures collaborent.

Le travail de recherche proposé dans le cadre de cette thèse est structuré en deux volets. Nous proposons d'abord une méthode de régulation du trafic à une intersection s'appuyant sur la négociation automatique. Notre méthode se fonde sur un système d'argumentation décrivant l'état du trafic et les préférences de chaque véhicule, appuyé par des méthodes de raisonnement pour les véhicules et les infrastructures. Notre méthode de régulation permet ainsi de coordonner la traversée de l'intersection par les véhicules avec plus de précision que ce que permettent les feux tricolores, en prenant notamment en compte la complexité de l'environnement ainsi que la trajectoire individuelle de chaque véhicule. Par ailleurs, notre méthode permet l'application de politiques de régulation particulières à une échelle plus large.

Dans le deuxième volet de cette thèse, nous proposons une méthode de coordination des bus avec les autres véhicules. En effet, dans les systèmes de régulation actuels, les bus bénéficient souvent d'une priorité à l'intersection, mais ils subissent les dégradations de la qualité du trafic général ce qui doit être nécessairement pris en compte puisque cela nuit à leur efficacité et à leur attractivité. La méthode que nous proposons permet à un bus de se coordonner de manière anticipative avec les prochaines intersections qu'il prévoit de traverser, afin de mettre en place une politique commune de régulation qui permet au bus d'atteindre plus facilement son prochain arrêt en subissant le minimum de congestions potentielles. La mise en place

de cette politique commune repose sur la méthode de régulation à l'intersection décrite précédemment.

Abstract

Since several decades, urban congestion causes various problems and deteriorate the quality of life of citizens who live in cities. Several methods are used to reduce urban congestion, notably traffic regulation and promotion of public transportation. Since the 1990's, the usage of tools from artificial intelligence, particularly distributed methods and multi-agent systems, allowed to design new methods for traffic regulation. Indeed, these methods ease to take into account the complexity of traffic-related problems with distribution. Moreover, the improvement of the communication abilities of the vehicles and the coming of autonomous vehicles allow to consider new approaches for regulation, in which the vehicles and infrastructures collaborate.

The research work presented in this work is twofold. First we propose a method for traffic regulation at an intersection based on automatic negotiation. Our method is based on an argumentation system describing the state of the traffic and the preferences of each vehicle, relying on reasonning methods for vehicles and infrastructures. Our regulation method enables to coordinate the crossing of the intersection with more accuracy than traffic lights, taking into account the complexity of the environment and individual trajectory of each vehicle. Moreover, our method allows the application of specific regulation policies on a larger scale.

In the second part of this thesis, we propose a coordination method for buses with the other vehicles. Indeed, in contemporary regulation systems the bus often benefit from priority on an intersection, but they suffer from traffic deterioration and it has to be taken into account because this damages their efficiency. Our method allows a bus to coordinate in an anticipatory way with the next intersections on its trajectory, in order to define a common regulation policy allowing the bus to reach its next stop without suffering from potential congestions. Setting up this policy is based on the regulation method for the intersection described previously.

Remerciements

Je tiens tout d'abord à exprimer ma reconnaissance à Samir Aknine, directeur de cette thèse, pour m'avoir conseillé et pendant ces quatre années, pour sa grande disponibilité et ses nombreuses relectures, ainsi que pour sa bonne humeur et son soutien.

Je tiens également à remercier Neila Bhouri, co-directrice de cette thèse, pour ses précieux conseils et pour m'avoir aidé à acquérir une meilleure culture du milieu des transports, ainsi que pour ses encouragements.

Je souhaite exprimer ma gratitude aux membres du jury pour avoir accepté de consacrer du temps à mon travail de recherche : René Mandiau et Leila Merghem-Boulahia, qui ont pris la peine d'être rapporteurs de cette thèse, ainsi que Saïd Mammar, examinateur de cette thèse et qui m'a fait l'honneur de présider le jury. Les remarques qu'ils m'ont faites, les questions qu'ils m'ont posées et les conversations que j'ai pu avoir avec eux m'ont permis de préciser mon travail et d'approfondir ma réflexion.

Je remercie les membres de l'équipe SMA du LIRIS, permanents et doctorants, pour tous les échanges qui ont eu lieu au sein de l'équipe, que ce soit lors des différents séminaires et réunions ou lors de conversations plus informelles, toujours agréables et constructifs.

Je suis reconnaissant à toutes les personnes avec qui j'ai collaboré dans le cadre d'enseignements, car ils ont contribué à rendre cette expérience aussi agréable et enrichissante.

Je remercie également ceux qui ont pris part aux travaux liés à cette thèse, notamment Huan Vu à qui je souhaite bon courage pour sa thèse, ainsi qu'aux différents stagiaires ayant collaboré avec l'équipe, notamment Antoine Richard, Paul Talvat, Bruno Dumas et Damien Mornieux.

Je souhaite également remercier tous les collègues et camarades du LIRIS pour tous ces moments de convivialité si agréables et importants, autour d'un café, d'un repas,

d’une table de jeu... Je ne me risquerai pas à lister les noms par crainte d’oublier quelqu’un, mais je compte sur ces personnes pour se reconnaître et les remercie toutes vivement.

Je remercie enfin ma famille ainsi que mes amis, tous les copains du Nord, de Picardie, de Paris, de Belgique, de Lyon, de Savoie, et bien sur de Valence. Merci à tous pour votre soutien et vos encouragements.

Table des matières

Introduction

Contexte et motivation

Depuis plusieurs décennies, la congestion urbaine est à l'origine de différentes nuisances pour les habitants des villes. Plusieurs réponses sont apportées à ce problème, notamment par la régulation du trafic et par le développement des transports en commun, ce qui permet d'en atténuer les effets négatifs. Les évolutions des technologies durant les dernières décennies ont permis d'améliorer le niveau d'équipement des véhicules, ce qui permet de proposer de nouvelles réponses aux problèmes liés à la question du trafic urbain.

Les véhicules sont maintenant capables de communiquer et de se coordonner, et les approches récentes tirent parti de ces avancées pour résoudre des problèmes de recherche qui peuvent être liés à différentes exigences comme la sécurité, le confort, l'efficacité énergétique ou celle qui nous intéresse plus particulièrement dans le cadre de cette thèse, à savoir la régulation du trafic. L'exploitation de ces capacités a notamment lieu dans le cadre des systèmes coopératifs, dans lesquels les véhicules, infrastructures et conducteurs coopèrent, et dans lequel cette thèse s'inscrit. Différentes méthodes s'appuient sur des techniques d'intelligence artificielle pour proposer de nouvelles réponses aux problèmes de trafic.

Dans cette thèse, nous proposons une méthode de régulation pour le trafic, notamment pour les transports en commun, s'appuyant sur des méthodes d'intelligence artificielle distribuée appelées systèmes multi-agents.

Objectifs et contributions

Notre approche a pour but de favoriser les transports en commun dans le trafic, et en particulier les bus. Or, ceux-ci sont soumis au trafic général au même titre que les

véhicules particuliers présents sur le réseau et traiter le problème de la régulation pour les bus implique de traiter la régulation du trafic général.

Nous articulons donc notre approche sur l'utilisation de deux méthodes complémentaires. La première consiste à déterminer et appliquer localement, c'est-à-dire à l'échelle d'une intersection, une politique de régulation prenant aussi bien en compte les particularités des différents véhicules que des critères apparaissant à plus large échelle, comme les besoins d'un bus ayant des horaires à respecter. La seconde méthode consiste à déterminer ces besoins pour un bus ayant à franchir une série d'intersections avant une certaine date, en coordonnant ces intersections et ce bus.

Nous présentons dans un premier temps la méthode pour la régulation à l'échelle de l'intersection. Celle-ci tire parti des capacités de communication des véhicules. Dans cette méthode, les véhicules décident d'une date d'engagement dans l'intersection pour chacun avec un niveau de précision élevé, en réalisant une négociation automatique. Cette négociation s'appuie sur des arguments portant sur l'état du trafic à différentes échelles, permettant de prendre en compte différents critères pour cette prise de décision locale, comme les contraintes individuelles des véhicules, l'état général du trafic, la localisation d'éventuelles congestions, ou encore la circulation des bus.

Nous présentons dans un second temps une méthode permettant la coordination d'un bus avec plusieurs intersections, permettant à celui-ci d'anticiper sa progression à l'échelle de plusieurs intersections, en fonction de ses objectifs et de son retard éventuel. Cette anticipation de la progression du bus à l'échelle de plusieurs intersections s'appuie sur le précédent mécanisme, qui permet de mettre en oeuvre à l'échelle de chaque intersection la politique décidée pour celle-ci lors de la coordination du bus avec les intersections.

Organisation de la thèse

Le présent document est organisé de la manière suivante.

Le chapitre 1 présente un état de l'art des méthodes de régulation de trafic. Après avoir expliqué les notions de base de régulation du trafic, nous y présentons les principales méthodes existantes pour la régulation, pour le trafic général comme pour les transports en commun, qu'elles soient réellement mises en oeuvre ou non, et dont certaines s'appuient sur les systèmes multi-agents.

Le chapitre 2 présente notre méthode de régulation à l'échelle de l'intersection. Celle-ci s'inscrit dans le cadre des systèmes coopératifs, dans lequel on suppose une communication et une coopération des différents véhicules. Ceux-ci échangent des informations et décident collectivement de la politique de régulation à appliquer, attribuant de cette manière une date de passage dans l'intersection à chaque véhicule. Cette méthode repose sur une négociation entre les véhicules.

Le chapitre 3 présente notre méthode de coordination du bus avec les intersections. Le but de cette méthode est de générer un ensemble de politiques pour les intersections permettant à un bus d'atteindre son prochain arrêt à la date prévue tout en pénalisant un minimum le trafic pour les autres véhicules présents dans la zone. La méthode utilisée s'inspire des techniques dites "branch and bound".

Le chapitre 4 présente les implémentations réalisées pour les méthodes présentées dans les chapitres 2 et 3, et donne les résultats obtenus dans les expérimentations menées.

Le chapitre 5 fait la synthèse sur les méthodes développées, les résultats obtenus et les perspectives de recherches liées à ces travaux.

1

État de l'art des systèmes de régulation du trafic

La congestion urbaine est un problème de plus en plus présent dans de nombreuses villes à travers le monde. Ce phénomène a des effets sur les déplacements quotidiens des citadins et a de nombreux impacts négatifs en termes de qualité de vie : durée des déplacements allongée, pollution atmosphérique, pollution visuelle et sonore, coût financier, etc.

Ce problème peut être vu comme une surutilisation d'une ressource rare et peu coûteuse pour l'utilisateur : la route. Un axe de réponse possible pour réduire cette surutilisation est de fournir plus d'infrastructures routières, mais ce type de solutions est coûteux, parfois impossible en termes d'espace, et ne tient pas compte des nuisances provoquées par l'extension du réseau routier. Un autre axe consiste à réduire le nombre de véhicules sur le réseau. Parmi les solutions couramment appliquées, on peut citer la réduction de l'utilisation du réseau routier par des politiques coercitives, avec des zones piétonnes ou des zones à péage dans certains centres-villes, ou incitatives, par exemple par l'encouragement de l'utilisation des transports en commun. En effet, certains types de transports en commun, comme les métros, ne sollicitent pas du tout le réseau routier, et les bus permettent de transporter beaucoup plus d'usagers avec des ressources routières égales. Enfin, un dernier axe consiste à réguler le trafic sur le réseau afin d'améliorer son efficacité et de diminuer le temps passé par les véhicules sur le réseau. Nous avons choisi dans cette thèse de combiner les deux dernières approches : d'une part, améliorer la qualité de la régulation du trafic afin de diminuer le temps passé par les véhicules dans le réseau et d'autre part, encourager l'utilisation des transports en commun, notamment les bus, en améliorant leur temps de parcours et leur régularité.

Dans la suite de ce chapitre, nous donnons en premier les notions de base de la régulation du trafic urbain et puis l'état de l'art des systèmes de régulation du trafic des véhicules particuliers et des transports en commun (1.1). Dans cet état de l'art, nous commençons par passer en revue les systèmes classiques connus pour la régulation du trafic monomodal. Après avoir présenté le fonctionnement général d'un plan de feu (1.1.1), nous classons ces systèmes suivant leur manière de réguler les carrefours, distinguant ainsi les systèmes cycliques (1.1.2) et acycliques (1.1.3). Dans la section 1.2 nous faisons un état de l'art des systèmes fondés sur une

modélisation multi-agents. Nous distinguons dans ces systèmes ceux qui régulent des carrefours isolés (1.2.1) de ceux qui font de la coordination sur une zone ou tout un réseau (1.2.3). La section 1.3 est consacrée à la régulation des transports en commun. Dans sa première partie nous présentons les différents modes de régulation des bus (1.3.1), dans la deuxième nous explicitons les différentes méthodes pour donner la priorité aux transports en commun dans les réseaux routiers (1.3.2) et enfin, nous faisons un état de l'art des systèmes de régulation qui tiennent compte des transports en commun (1.3.3).

1.1 Régulation de trafic urbain

La régulation du trafic urbain prend principalement deux formes.

La première est l'affectation de trafic, consistant à orienter les utilisateurs afin de répartir la charge de véhicules sur le réseau de manière à l'équilibrer. Cette affectation peut consister à guider les usagers ou à utiliser des moyens de régulation comme les "sens interdits", des cycles de feux plus longs, etc. Le guidage peut prendre différentes formes selon la stratégie utilisée, que l'information soit communiquée à l'utilisateur avant que celui-ci ne prenne la route ou tout au long de son trajet [54].

La seconde est le contrôle des véhicules aux intersections, le plus souvent réalisé par des feux de signalisation. Des systèmes de contrôle à feux fixes ou adaptatifs sont utilisés dans différentes villes à travers le monde.

1.1.1 Fonctionnement d'un plan de feu

Une intersection est un lieu où se croisent deux routes ou plus. On appelle "section" la portion de route située entre deux intersections. Chaque section peut être composée d'une ou plusieurs voies dans chaque sens. On peut découper une intersection en plusieurs zones (voir Figure 1.1) :

- la zone de conflit, qui correspond à la zone où plusieurs routes se superposent effectivement et où les conflits entre les véhicules sont les plus susceptibles d'arriver.

- la zone de stockage, en amont de la zone de conflit, contient les véhicules susceptibles de s'engager dans la zone de conflit. La limite entre cette zone et la zone de conflit est appelée "ligne de feu".

- la zone de sortie, en aval de la zone de conflit, par laquelle les véhicules sont évacués.

Les feux de signalisation permettent d'attribuer aux différents usagers de la route, piétons ou véhicules, le droit de s'engager dans la zone de conflit de l'intersection ou sur un passage piéton. Pour les véhicules, ceci est représenté par un code couleur :

- le vert autorise les véhicules à s'engager dans la zone de conflit.
- le jaune est un état transitoire annonçant le passage du vert au rouge, et les véhicules doivent s'arrêter dans la mesure du possible.
- le rouge interdit de s'engager dans la zone de conflit.

Le groupe de véhicules concernés par le signal dépend de la configuration de l'intersection, et le signal peut concerner une ou plusieurs voies d'une même section. Parfois le signal concerne les véhicules empruntant un courant particulier de l'intersection, c'est-à-dire les véhicules allant d'un certain groupe de voies de la zone de stockage à un certain groupe de voies de la zone de sortie.

Les systèmes de feux classiques fonctionnent de la manière suivante : le temps est découpé en une succession de cycles, qui sont eux-mêmes une succession de phases. Une phase est la période durant laquelle un ou plusieurs courants sont admis dans l'intersection. Ces courants doivent être cohérents entre eux, c'est-à-dire ne pas se chevaucher (on dit alors qu'ils sont compatibles), ou être suffisamment peu utilisés pour que le risque de conflits soit bas. À la fin d'une phase le signal passe au jaune, puis au rouge. Après quelques secondes de rouge intégral durant lesquelles aucune phase n'est active, afin de permettre l'évacuation de la zone de conflit, une autre phase commence. Un cycle est donc une succession de phases durant lesquelles tous les courants admis dans l'intersection ont été servis au moins une fois.

Ce type de fonctionnement classique est mis en oeuvre par le système TRANSYT (TRAffic Network Study Tool) [56], l'un des plus anciens systèmes proposés pour la gestion des feux. Celui-ci nécessite la connaissance de nombreux paramètres sur le réseau sur lequel il est déployé : géométrie du réseau, nombre de véhicules mesuré ou attendu, etc. Ce système réalise une optimisation des plans des feux afin de produire un paramétrage optimal de ceux-ci. Toutefois cette optimisation est réalisée hors-ligne, et ce système est donc incapable de s'adapter aux variations effectives du trafic ayant lieu en temps réel.

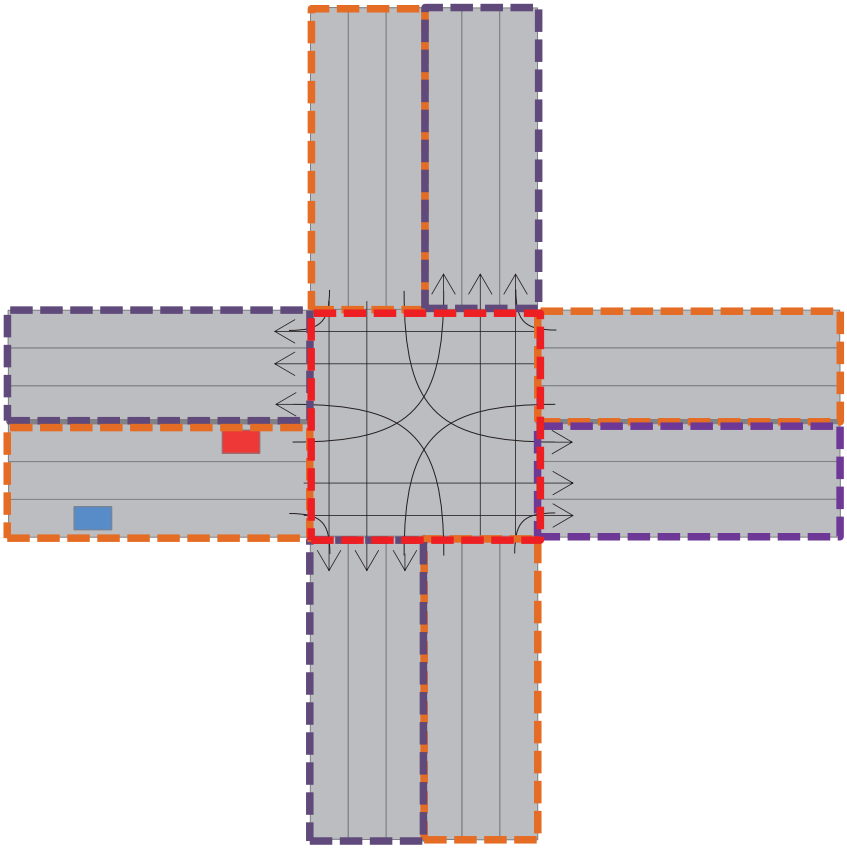


Figure 1.1: Une intersection entre deux routes ayant 6 voies chacune. On distingue la zone de conflit en rouge, la zone de stockage en orange et la zone de sortie en violet. Les flèches représentent les différents courants de véhicules possibles. Ainsi le véhicule rouge ou un véhicule sur la même voie ne peut que tourner à gauche, empruntant le courant Ouest-Nord. Le véhicule bleu ou un véhicule sur la même voie peut emprunter deux courants différents, le courant Ouest-Est ou le courant Ouest-Sud.

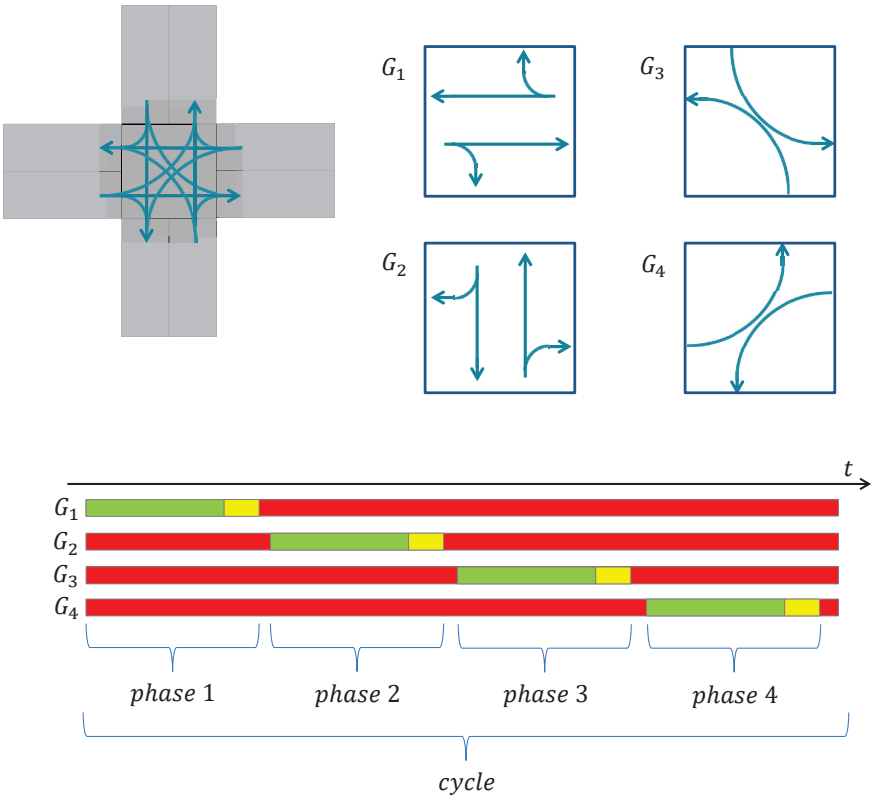


Figure 1.2: Une intersection entre deux routes ayant 2 voies chacune. 12 courants existent pour traverser cette intersection. On peut décomposer ces courants en 4 groupes de courants compatibles G_1, G_2, G_3 et G_4 (d'autres découpages sont possibles). Les périodes durant lesquelles le vert et le jaune sont affectés à chacun de ces groupes de courants sont les différentes phases formant le cycle.

Au contraire, certains systèmes, dits "adaptatifs", mesurent en temps réel certaines données sur le trafic et exercent une stratégie qui s'adapte en temps réel aux conditions effectives du trafic.

De plus certains systèmes de transport, qu'ils concernent la question de la régulation ou non, sont dits "coopératifs". Un système de transport est dit coopératif lorsqu'il atteint un "État de coopération [de ses] acteurs [...] (exploitants, infrastructures, véhicules, leurs conducteurs et autres usagers de la route) dans le but d'offrir le plus sûr, le plus fiable et le plus confortable des déplacements" [61]. Un système coopératif nécessite donc des capacités de communication et de réaction de la part des acteurs, notamment des véhicules. Ceci est permis par le niveau d'équipement croissant des véhicules et des conducteurs, et on appelle "véhicule connecté" un véhicule équipé pour la communication. Ceci inclut les véhicules autonomes, c'est-à-dire sans chauffeur humain, dont la présence sur les routes pourrait devenir commune dans un avenir proche. Un système coopératif peut avoir différents objectifs, comme la sécurité des déplacements ou la qualité du trafic.

1.1.2 Systèmes adaptatifs cycliques

Plusieurs systèmes adaptatifs restent proches de ce fonctionnement classique des feux de signalisation par cycles. Toutefois, ceux-ci modifient les plans de feu dynamiquement dans le but de s'adapter aux conditions de circulation sur le réseau.

Le système de ce type le plus répandu est le système SCOOT (Split Cycle and Offset Optimisation Technique) [42], développé au Royaume-Uni dans les années 1980. Des capteurs (boucles électromagnétiques) situés sur chaque tronçon permettent de mesurer le flux de véhicules, de détecter les remontées de files d'attente dues aux congestions, et la présence de véhicules éventuellement arrêtés pour une autre raison. Toutes ces informations sont utilisées et traitées par un modèle d'écoulement du trafic pour affiner les durées des phases des feux pour chacune des intersections de la zone supervisée par le système, afin de minimiser la somme pondérée du nombre d'arrêts des véhicules et de la longueur des files d'attente devant les intersections. Cette stratégie d'optimisation est mise en oeuvre par de légères variations apportées à la durée des phases (4 secondes au maximum) et/ou à la durée des cycles (8 secondes au maximum) des feux de la zone concernée et des décalages ponctuels de commutation des feux.

SCATS (Sydney Co-ordinated Adaptative Traffic System) [60] a été développé en Australie dans les années 1980. Tout comme SCOOT, ce système récupère des informations par des boucles électromagnétiques, cependant celles-ci sont situées au niveau de la ligne de feu. En effet SCOOT s'appuie sur un modèle d'écoulement

du trafic pour anticiper la progression alors que SCATS fonctionne sur les données directement perçues. Cette stratégie s'appuie sur plusieurs bibliothèques : pour les durées de cycles, les décalages et les durées de vert. Un algorithme permet de sélectionner les éléments les plus appropriés dans les différentes bibliothèques pour construire dynamiquement un plan de feu. Différents critères d'optimisation sont utilisés en fonction du contexte (nuit, fluidité, période de pointe, congestion). La mise en oeuvre de cette stratégie prend les mêmes formes que la stratégie de SCOOT, à savoir modification des durées de cycles, de phases et des décalages. SCATS réalise une optimisation sur un sous-système allant de l'intersection isolée à un ensemble de 10 intersections.

TUC (Traffic-responsive Urban Control) [20] est un système développé dans les années 1990-2000 en réponse au manque d'efficacité de SCOOT et SCATS face à des conditions de trafic changeant rapidement, en particulier en situation de congestion, et à la complexité algorithmique des systèmes acycliques décrits ci-après (OPAC, PRODYN, RHODES...) qui est un obstacle à leur déploiement à l'échelle d'un réseau urbain entier. La stratégie principale s'appuie sur une théorie du contrôle automatique (Linear-Quadratic-Regulator) pour modifier dynamiquement la durée relative des phases du cycle. Dans ce système, la récolte des données dynamiques peut s'appuyer sur des boucles électromagnétiques comme pour SCOOT ou SCATS, ou sur un système de détection vidéo.

1.1.3 Systèmes adaptatifs acycliques

À l'inverse des systèmes adaptatifs présentés ci-dessus, certains systèmes s'éloignent du fonctionnement par cycles des feux de signalisation, en faisant disparaitre cette notion. Le plus souvent celle-ci est remplacée par le choix de l'instant de commutation, c'est-à-dire le passage à la phase suivante.

L'un des plus anciens systèmes de ce type est PRODYN (PROgrammation DYNamique) [41] développé dans les années 1980. Dans ce système, 2 à 3 boucles électromagnétiques sont disposées sur chaque tronçon, et l'état supposé du trafic sur chaque voie est estimé à l'aide d'un modèle d'écoulement simple permettant d'anticiper la progression des véhicules sur la voie. Ce système réalise une optimisation sur l'intersection "isolée", toutefois certaines versions de ce système permettent une communication entre les intersections voisines afin d'anticiper les flux entrants. La stratégie utilisée par ce système consiste à analyser à chaque pas de temps (de 5 secondes) si commuter l'état du feu (i.e. changer de phase) est la décision optimale, c'est-à-dire si elle minimise le temps d'attente des véhicules devant l'intersection pour les 75 prochaines secondes d'après le modèle d'écoulement utilisé.

OPAC-RT (Real-Time Optimization Policies for Adaptative Control) [37], également développé dans les années 1980, partage un certain nombre de points communs avec PRODYN : celui-ci est également basé sur la commutation, et sa stratégie s'appuie également sur la minimisation du temps d'attente des véhicules à l'aide d'un modèle d'écoulement. Il existe plusieurs différences avec PRODYN : la position des capteurs (un seul en début de tronçon), le modèle d'écoulement utilisé, ainsi que la durée de l'horizon de temps consideré.

UTOPIA (Urban Traffic OPtimisation by Intergrated Automation) [51] fut développé dans les années 1980, un de ses objectifs principaux étant la mise en place de priorités pour les transports en commun. Des capteurs sont situés sur chaque section près des lignes de feu. Ce système fonctionne sur deux niveaux. Au niveau régional, un modèle d'écoulement macroscopique est utilisé pour prédire la progression des véhicules et une optimisation est réalisée afin de minimiser le temps de trajet des véhicules sur le réseau. Des plans de feu de référence ainsi que d'autres paramètres issus de cette optimisation sont transmis au niveau local, c'est-à-dire au niveau des intersections. À partir des recommandations du niveau régional, le niveau local, c'est-à-dire chaque intersection, réalise localement une optimisation de la somme pondérée du temps perdu par les véhicules, de la longueur des files d'attente et du nombre d'arrêts des véhicules. Les deux niveaux de régulation considèrent des échelles de temps différentes : le niveau régional et le niveau local ont respectivement un horizon de temps de 30 minutes et de 120 secondes.

1.1.4 Autres systèmes adaptatifs

Les systèmes présentés dans les deux sections précédentes s'appuient sur des approches de référence et ont tous fait l'objet d'une mise en oeuvre réelle dans une ou plusieurs villes. Plusieurs autres systèmes ayant des approches différentes existent mais n'ont pas fait l'objet d'une implantation, excepté une expérimentation sur site pour certains d'entre eux.

CRONOS (ContROl of Networks by Optimization of Switchovers)[12] est un système différent qui s'appuie sur une approche heuristique du problème. Il permet une optimisation qui peut être décentralisée, où le cas de chaque intersection est traité localement et individuellement, ou plus centralisée, où une optimisation globale est réalisée sur une zone allant jusqu'à 6-8 intersections. Une de ses particularités est de récupérer des données sur le trafic non pas par des boucles électromagnétiques mais par des caméras. Enfin, CRONOS ne dispose pas de phases prédéterminées mais s'appuie sur des contraintes de sécurité permettant d'interdire des états de feu susceptibles de créer des conflits.

RHODES (Real-Time Hierarchical Optimized Distributed Effective System) [52] fonctionne sur 3 niveaux différents : le premier prend en compte les caractéristiques du trafic qui évoluent lentement (modification du réseau, évolution des itinéraires les plus fréquentés...), le deuxième niveau s'appuie sur le premier pour estimer le nombre de véhicules par heure sur chaque section et d'en déduire une durée de vert approximative, ce qui permet au dernier niveau de produire un contrôle adapté à la demande à l'échelle d'une intersection.

MOVA (Microprocessor Optimised Vehicle Actuation) [64] s'appuie sur un type d'information original, les distances inter-véhiculaires, pour déterminer l'état de saturation du trafic. Ainsi, les critères pris en compte dans la stratégie de régulation varient en fonction de cet état.

La stratégie de MOTION (Method for the Optimization of Traffic signals In On-line controlled Networks) [11] fonctionne de la manière suivante : chaque intersection détermine une durée minimale de cycle en fonction des flux de véhicules estimés, puis une durée de cycle commune à toutes les intersections est déterminée, et une coordination a lieu sous la forme d'un décalage de phases afin de créer des ondes vertes.

La stratégie de CARS (Control Audoadaptativo para Redes Semaforizades) [7] s'approche beaucoup de celle de SCOOT, à ceci près qu'elle s'appuie sur un horizon de temps glissant dont la durée est ajustée en permanence pour correspondre à la durée du cycle du carrefour.

1.1.5 Discussion sur les approches classiques

Un certain nombre des systèmes présentés ci-dessus, que ce soit TRANSYT ou les systèmes adaptatifs, sont mis en oeuvre dans de nombreuses villes à travers le monde. Ceci peut bien sûr s'expliquer historiquement, cependant ces systèmes possèdent un certain nombre de propriétés intéressantes :

- perception limitée de l'environnement : ces systèmes s'appuient sur les données récoltées par des capteurs simples, le plus souvent des boucles électromagnétiques, présentes entre une et trois fois par tronçon. Si l'information utilisée par ces systèmes est limitée, cela ne nécessite pas un équipement complexe et coûteux.

- simplicité computationnelle : les calculs réalisés par ces systèmes sont le plus souvent réalisés de manière individuelle par les intersections et sont relativement simples, ce qui facilite leur application en temps réel.

- décentralisation et simplicité communicationnelle : certains systèmes comme UTOPIA sont structurés en plusieurs niveaux et ont recours à une centralisation régionale, d'autres comme PRODYN permettent une communication entre des intersections voisines. Bien que ces communications aient un coût, celles-ci sont relativement rares, ce qui rend ces systèmes moins sensibles à un certain nombre de pannes et ne nécessitent pas d'infrastructures de communication complexes.

Ces propriétés permettent d'écarter un certain nombre d'hypothèses, notamment sur le niveau d'équipement de l'environnement (infrastructures et véhicules). Cependant les technologies contemporaines, notamment celles embarquées dans les véhicules, rendent ces hypothèses de moins en moins coûteuses. De plus ces systèmes adaptatifs font d'autres hypothèses, qui peuvent être plus ou moins simplificatrices.

Ces systèmes, en particulier les systèmes cycliques, limitent les actions à un certain nombre de possibilités, le plus souvent issus de la régulation traditionnelle : modification de durées de phases, de cycles et déphasages entre intersections voisines. D'autres actions de régulation peuvent cependant être envisagées pour permettre une régulation plus dynamique. De plus les véhicules connectés permettent d'envisager de nouvelles possibilités d'action qui peuvent être utilisées pour la régulation, par exemple communiquer sur la stratégie de contrôle d'une intersection afin que le véhicule adapte son profil d'accélération.

Certains de ces systèmes s'appuient sur des modèles d'écoulement afin de réaliser des prédictions de l'évolution du trafic. S'il semble difficile de s'affranchir de tout modèle de prédiction dans une démarche anticipative, la simplicité des perceptions de ces systèmes ne permet pas un ajustement dynamique des prédictions réalisées. Cette critique peut être renforcée par le fait que les véhicules connectés, de par leurs capacités de communication entre eux et de coordination, présentent des dynamiques d'écoulement différentes des véhicules classiques. La diffusion de ce type de véhicules pourrait rendre les prédictions réalisées par ce type de systèmes de moins en moins pertinentes, et ainsi diminuer l'efficacité de la régulation réalisée.

1.2 Agents dans la régulation de trafic

Différents problèmes liés à la régulation du trafic urbain ont été identifiés dans la communauté informatique, notamment par la communauté multi-agents [8]. Nous décrivons ici plusieurs de ces travaux, dans leur diversité, qu'ils portent sur la régulation ou la coordination entre les véhicules, à l'échelle d'une intersection ou d'une zone plus large. Dans le premier cas, la coordination et la régulation à une in-

tersection peuvent être considérées comme un problème indépendant de la question de la régulation globale sur un réseau, et on parle d'intersection "isolée" (le reste du réseau n'étant pas pris en considération). Dans le second cas, une coordination a lieu entre les intersections afin de réaliser une régulation plus cohérente à l'échelle du réseau.

1.2.1 Régulation sur une intersection isolée

Pour une intersection isolée, différents problèmes de coordination entre les véhicules peuvent être envisagés et différentes approches sont possibles pour chacun de ces problèmes. Une grande partie de ces difficultés concerne l'affectation de trafic en temps réel et la manière dont le droit de passage, c'est-à-dire l'autorisation pour un véhicule de s'engager dans l'intersection, est attribué à chaque véhicule. Certaines de ces approches nécessitent un agent régulateur qui applique seul une politique de régulation, d'autres impliquent une coordination inter-véhiculaire.

Une première approche est abordée dans [66]. Dans ce travail, chaque véhicule communique les informations dont il dispose aux autres afin de permettre aux véhicules de se coordonner avec eux. Les différents véhicules ont des trajectoires qui se coupent sur des points appelés "points de conflit". Afin d'allouer une date de passage à chaque véhicule, le comportement des agents s'appuie sur un plan de collaboration. En l'absence de toute coordination, les véhicules rencontrent des conflits. En s'appuyant sur une méthode simple d'évaluation des conflits, les véhicules repoussent leurs dates respectives d'entrée dans l'intersection, une à une, jusqu'à éviter tout conflit. Le plan de collaboration proposé par cette méthode permet aux véhicules de changer l'ordre dans lequel leurs dates de passage sont repoussées. Toutefois les auteurs ne fournissent pas les détails du mécanisme d'interaction, et n'indiquent pas ses propriétés ni les garanties qu'il apporte.

[4] s'appuie sur la notion de justice (fairness) pour la régulation de trafic, en proposant une politique de contrôle pour les intersections s'appuyant sur l'historique des véhicules. Cette politique réduit la variance du temps total passé par les véhicules à attendre devant les intersections à feux durant leur trajet. Chaque intersection a un contrôleur capable de produire différents plans de feu. Un plan de feu est une combinaison de durées de vert, jaune et rouge pour chaque approche de l'intersection, et évite tout conflit. Le contrôleur utilise différentes fonctions de score basées sur l'efficacité et la justice pour évaluer chaque plan de régulation possible. L'efficacité et la justice de chaque plan sont évaluées, pour différentes tailles de grilles et différentes charges de trafic.

Intersections autonomes

Certains travaux portant sur l'intersection isolée concernent la coordination des véhicules, et d'autres concernent la régulation effectuée par l'intersection. L'AIM (Autonomous Intersection Management, gestion des intersections autonomes) cherche à coordonner les véhicules autonomes sur l'intersection. Cette coordination passe par l'accord du droit de passage aux véhicules, et l'AIM réalise donc une régulation de l'intersection à l'aide de différentes informations portant sur les véhicules. Les travaux suivants concernent l'AIM.

Dans [23], K. Dresner and P. Stone proposent un mécanisme d'attribution du droit de passage pour les véhicules autonomes s'appuyant sur la réservation. Celui-ci s'appuie sur une politique appelée FCFS (First Come First Served, premier arrivé premier servi), accordant le droit de passage à chaque véhicule en faisant la demande, à la première date possible. Ce mécanisme permet de prendre en compte les conducteurs humains [24] en utilisant une politique classique d'intersection à feu pour les conducteurs humains, et en fournissant des dates de passage aux véhicules autonomes en utilisant la politique FCFS. Bien que ce mécanisme s'accommode de la présence de conducteurs humains, ses principaux bénéfices sont dus à l'utilisation de FCFS pour les véhicules autonomes.

[38] permet une coordination entre les véhicules à l'approche d'une intersection en construisant un graphe de priorité (orienté) déterminant l'ordre d'admission des véhicules. L'article propose une caractérisation d'un graphe de priorité réalisable. Les auteurs évoquent comme possible suite à ces travaux l'utilisation d'heuristiques permettant de construire de tels graphes de manière optimale.

[65] s'appuie sur la notion de courants ("streams") représentant les trajectoires possibles des véhicules à l'intersection. Par exemple, l'ensemble des véhicules arrivant à l'intersection depuis une voie située au Sud de l'intersection pour se rendre sur une voie située à l'est de celle-ci forment un courant. Dans cette approche, des groupes de courants sont constitués de manière à ce qu'aucun des courants formant un groupe ne se coupent. On parle de courants compatibles entre eux. Les courants d'un groupe peuvent avoir simultanément le feu au vert. Le problème de l'attribution du droit de passage est alors représenté comme un problème d'ordonnancement de tâches. À partir des groupes de courants compatibles, des groupes de véhicules sont constitués. Ces groupes sont représentés comme des groupes de tâches, et grâce à cette représentation, l'évacuation totale de l'intersection est minimisée en utilisant une méthode de résolution exacte (séparation et évaluation, programmation dynamique).

[58] et [59] proposent des mécanismes évaluant l'importance d'un gain de temps pour chaque conducteur. Par exemple, une minute a *a priori* plus de valeur pour un conducteur en retard à un entretien d'embauche que pour un conducteur rentrant chez lui après sa journée de travail. Dans ces mécanismes, chaque véhicule a un budget lui permettant d'acheter ou de vendre des créneaux temporels. Dans [58], un mécanisme d'enchère appelé ITSA (Initial Time Slot Auction) est proposé : lorsqu'un véhicule s'approche d'une intersection, il a la possibilité d'enchérir avec une partie de son budget pour tenter d'obtenir le premier créneau de passage disponible, lui assurant une traversée rapide de l'intersection. Dans [59], un autre mécanisme est proposé : TSE (Time Slot Exchange). Avec TSE, les véhicules peuvent échanger leurs créneaux respectifs contre du crédit. Un conducteur pressé pourra dépenser ce qu'il a économisé pour gagner du temps, les autres gagneront ainsi du crédit. Un agent courtier gère ces échanges en fonction des demandes de chaque conducteur.

[62] propose également une approche de l'AIM inspirée par les marchés. Lorsqu'ils choisissent leurs itinéraires, les conducteurs optent souvent pour le plus rapide après avoir estimé le temps de trajet pour chacun des itinéraires potentiels. Dans ce modèle, les conducteurs doivent acheter une réservation auprès d'agents gestionnaires afin de pouvoir franchir les intersections. Ce système de réservation fournit des incitations aux conducteurs afin de les amener à explorer des itinéraires alternatifs. Dans ce mécanisme, chaque gestionnaire d'intersection doit déterminer ses frais de réservation afin de maximiser son profit. Avec quelques véhicules, le gestionnaire d'intersection obtient un profit peu élevé, mais avec un grand nombre de véhicules il perd en profit à cause de la congestion. Il doit donc ajuster ses frais de réservation afin d'attirer un nombre moyen de véhicules. Les gestionnaires d'intersection utilisent un Q-learning afin d'ajuster ces frais de réservation.

[22] propose un modèle anticipatif de coordination pour les systèmes multi-agents. Dans ce modèle, les agents calculent les conséquences de leurs actions en utilisant des réseaux de contraintes permettant de prédire et d'éviter l'apparition d'états indésirables. Cet algorithme est appliqué à un problème de régulation de trafic à l'échelle de l'intersection. Les véhicules sont des agents capables d'avancer ou de s'arrêter et doivent traverser une intersection. À l'échelle d'une intersection, des situations d'interblocage (deadlocks) peuvent avoir lieu, notamment à cause des véhicules tournant à gauche (dans un pays où les véhicules roulent à droite). Ces situations sont considérées comme des états indésirables par l'algorithme, qui cherche à les éviter.

[53] propose une approche pour construire des politiques de régulation multi-agents en utilisant des méthodes d'apprentissage automatique non supervisé. Sur une intersection isolée, les véhicules peuvent se déplacer et doivent éviter les collisions. Une autorité de trafic rassemble les informations et réalise un raisonnement cas par

cas, s'appuyant sur les expériences passées pour déterminer la solution à appliquer pour résoudre la situation actuelle. Ensuite un gestionnaire de normes transforme les solutions trouvées en normes pour les agents véhicules, qui appliquent ces règles en utilisant un moteur de règles. Une réduction du nombre de normes nécessaires à la réalisation des buts du système (éviter les conflits) est également opérée pour que le nombre de règles à vérifier par les véhicules reste raisonnable.

1.2.2 Problèmes de coordination inter-véhiculaire à l'échelle de l'intersection

Les problèmes de régulation peuvent être vus comme des problèmes de coordination entre les véhicules, plus ou moins centralisés, directs et autoritaires en fonction de la méthode utilisée. Cependant ils ne sont pas les seuls problèmes de coordination entre les véhicules existant à l'échelle de l'intersection. Dans les approches suivantes, les véhicules utilisent la communication avec les autres véhicules pour réaliser une coordination dans le cadre d'une politique de régulation déjà choisie.

Dans [17], la coordination entre les véhicules est utilisée pour la simulation du trafic. En représentant le problème de l'intersection comme un jeu à deux joueurs où les joueurs sont les véhicules, puis comme un jeu à n joueurs, cette méthode permet de simuler un comportement réaliste quand une règle de priorité existe déjà. Les coups possibles pour le joueur sont "déplacement" et "arrêt", et la matrice de gain est construite par les joueurs, leur permettant de choisir le comportement le plus pertinent. Néanmoins la complexité de cette méthode est élevée et il est difficile de l'appliquer au-delà de quelques véhicules.

Un problème de coordination commun au sujet d'une intersection est celui de l'évitement des collisions (CA, Collision Avoidance), par exemple dans [16], [39]. Dans les approches que nous présentons ici, la CA consiste à ajuster la vitesse des véhicules autonomes approchant une intersection afin d'éviter toute collision, et la solution à ce problème passe souvent par l'utilisation d'équations mécaniques pour trouver la vitesse appropriée pour chaque véhicule. La CA ne concerne donc pas la politique de régulation bien que l'ajustement de la vitesse des véhicules puisse affecter leur ordre d'admission dans l'intersection. Cependant cet aspect n'est pas abordé en détail. Par exemple [16] affirme que l'approche présentée concernant l'ordre d'admission des véhicules peut être vue comme une simple règle de priorité ("Decision order : [...] our approach can be seen as a priority rule").
[47] présente le problème de CA comme un problème d'optimisation sous contraintes non linéaires. Dans cette méthode, les véhicules adaptent leurs vitesses sous différentes contraintes (accélération et décélération maximales, vitesses minimale et maximale, distance intervéhiculaire minimale) afin de minimiser la durée de che-

vauchement des trajectoires des véhicules au sein de la zone de conflit. Le calcul est réalisé de manière centralisée par un agent de contrôle de l'intersection.

1.2.3 Coordination de la régulation pour plusieurs intersections

Les travaux suivants s'appuient sur une coordination à l'échelle de plusieurs intersections. Une coordination à plus large échelle permet une meilleure efficacité de la régulation à l'échelle du réseau, par exemple par la formation d'ondes vertes. Une onde verte est un phénomène consistant à coordonner les feux tricolores de plusieurs intersections de manière à ce qu'un groupe de véhicules rencontre une succession de feux au vert, leur permettant une circulation fluide, et diminuant les pertes d'efficacité dues aux arrêts-redémarrages des véhicules.

[29] propose un modèle multi-agents hiérarchique pour la régulation de trafic. Dans ce modèle, des agents de trafic locaux (LTA, Local Traffic Agents) réalisent une politique de régulation à l'échelle de l'intersection en utilisant les données perçues. À une large échelle, un agent (ITA, Information Traffic Agent) mémorise des informations sur l'état de chaque intersection. À une échelle intermédiaire, des agents coordinateurs (CTA, Coordinator Traffic Agents) surveillent les intersections d'une zone pour fournir aux LTAs des informations sur l'état de leurs voisins, en particulier en cas de congestions, permettant aux LTAs d'ajuster leurs comportements en prenant en compte des informations et des objectifs à plus large échelle.

[45] s'appuie sur des groupes de courants compatibles (signal groups), qui sont représentés par des agents. À l'échelle de l'intersection, chaque groupe de courants négocie avec les autres pour obtenir le feu vert, ou une extension de vert, en fonction de la taille des files d'attente des véhicules pour chaque groupe de courants. La logique floue est utilisée pour déterminer si les files doivent être considérées comme longues ou courtes, de manière non booléenne. À l'échelle du réseau, les intersections sont capables d'échanger leurs données de trafic et de contrôle. Cela permet aux groupes de courants de prendre en compte les décisions de contrôle des intersections voisines afin d'éventuellement obtenir des extensions de vert permettant la formation d'ondes vertes.

[15] représente le contrôle du trafic comme un jeu stochastique. Le trafic a différents états possibles, et les différentes politiques de trafic sont les actions pouvant être réalisées par l'agent contrôlant l'intersection. Un Q-learning distribué est réalisé afin d'apprendre à appliquer la meilleure politique de trafic pour chaque état.

[2] propose un système multi-agents holonique pour le contrôle des feux à une intersection. Un système holonique est un système multi-niveaux dans lequel chaque "holon" est constitué de holons de niveau inférieur, ou d'agents atomiques du niveau minimal. Dans ce modèle, un agent atomique est un contrôleur de feux pour une intersection et réalise une régulation locale s'appuyant sur un Q-learning. Des holons de niveaux plus élevés représentent des zones du réseau, et leur rôle est de restreindre l'espace d'action de leurs sous-holons en leur donnant des actions abstraites à réaliser. Les super-holons et sous-holons réalisent un Q-learning commun et chaque niveau met à jour sa propre politique.

[40] montre qu'une optimisation à l'échelle de plusieurs intersections peut avoir lieu dans le contexte de l'AIM. À une échelle individuelle, la communication de l'itinéraire permet à chaque agent gestionnaire d'intersection de produire une estimation de la date d'admission des véhicules dans l'intersection. Ensuite, cette date de passage est communiquée aux véhicules, qui peuvent alors changer leur itinéraire à partir de ces estimations plus réalistes. À plus large échelle, ce travail traite du paradoxe de Braess, selon lequel l'ouverture de nouvelles options d'itinéraires pour les véhicules peut réduire l'efficacité globale du système. En effet, permettre aux véhicules de réaliser un choix dynamique d'itinéraires avec des buts individuels est susceptible de mener à un équilibre de Nash, sous-optimal, provoquant le paradoxe de Braess. En utilisant une inversion dynamique des voies, la topologie du réseau est dynamiquement modifiée et évite ainsi le paradoxe de Braess.

Certaines méthodes s'appuient sur le comportement des conducteurs, d'autres sur le contrôle des intersections à feux. [9] traite de la co-adaptation des véhicules et des contrôleurs de trafic. Plusieurs expériences sont menées, dans lesquelles seuls les conducteurs adaptent leurs comportements, seuls les contrôleurs adaptent leurs comportements, ou les deux à la fois. Le travail conclut que la co-adaptation mène à une amélioration du trafic, en particulier dans les situations observables à plus large échelle qui impliquent des centaines de véhicules.

[50] présente une approche pour la gestion des congestions dans les CSIN (Complex Self-Interested Networks) en utilisant la négociation entre les agents. Le réseau est représenté par un graphe et subdivisé en sous-graphes appelés "mondes". Cette division du graphe s'appuie sur certaines des propriétés classiques des graphes. Le problème principal est ainsi subdivisé en sous-problèmes, plus faciles à résoudre, et les agents négocient la position des "portes" entre chaque monde, permettant aux agents d'aller d'un monde à un autre. Un scénario de gestion du trafic est par ailleurs présenté succinctement, illustrant comment cette approche pourrait être appliquée à la gestion du trafic.

1.2.4 Discussion sur les différentes approches

Les méthodes présentées ci-dessus ne font pas des hypothèses similaires à celles des systèmes de contrôle classiques et supposent des environnements différents.

Mesures

La première différence est la perception des données et de l'environnement. Les systèmes classiques s'appuient essentiellement sur des capteurs classiques mesurant les variables macroscopiques du trafic (débit, concentration, vitesse...), cependant d'autres types de données peuvent être utiles. Les méthodes présentées ci-dessus n'abordent en général pas la question de la récolte des informations sur l'environnement. Cependant un certain nombre d'entre elles s'appuient sur une perception précise de la position des véhicules sur le réseau. Cette hypothèse est raisonnable dans un environnement où les véhicules sont hautement équipés et ont des capacités de perception et de communication importantes. Un environnement dense en véhicules connectés peut raisonnablement faire ce type d'hypothèses car les véhicules connectés disposent de ces deux capacités (perception et communication).

Variables de contrôle

Les variables de contrôle les plus utilisées sont les paramètres classiques des feux de signalisation à savoir les longueurs des phases et des cycles ainsi que les déphasages. Cependant d'autres approches sont présentées, comme par exemple dans [9] qui propose d'influencer le comportement des conducteurs dans leur choix d'itinéraire, ou [47] et [23] où l'on s'affranchit des feux de signalisation classiques pour envisager individuellement le droit de passage de chaque véhicule dans l'intersection.

Objectifs des différentes approches

Une grande partie des méthodes présentées, qu'elles soient issues des systèmes classiques ou plus innovantes, réalisent une forme d'optimisation. Cependant la fonction objectif à minimiser ou à maximiser est différente en fonction des approches et surtout du type de problème traité. Les fonctions objectif les plus utilisées classiquement concernent, directement ou indirectement, le retard des véhicules ou la minimisation de la longueur des files d'attente devant l'intersection. Certaines minimisent le retard moyen des véhicules par rapport à une prédiction, d'autres minimisent le temps

d'attente des véhicules aux intersections. [4] introduit un autre paramètre, l'équité entre les véhicules, en minimisant la variance du temps d'attente des véhicules devant les intersections. À l'inverse [58] n'évalue pas de la même manière le temps perdu par les différents véhicules et cherche à minimiser la somme, pondérée par cette évaluation, du temps d'attente des véhicules à l'intersection. Enfin, certaines approches présentées ci-dessus concernent des problèmes de coordination différents des problèmes d'optimisation du trafic, comme [47] qui cherche à produire une coordination des véhicules pour la traversée de l'intersection et réalise pour cela une minimisation de la durée de chevauchement des trajectoires des véhicules traversant l'intersection.

Centralisation et décentralisation

Les systèmes de contrôle classiques ne communiquent pas avec les véhicules sur le réseau, et un certain nombre d'entre eux exercent une régulation totalement locale où chaque intersection applique sa propre politique indépendamment des autres. Dans certains systèmes, les intersections sont coordonnées les unes avec les autres. Cette coordination peut être réalisée de manière distribuée par les intersections qui échangent des informations avec les intersections voisines à la manière de PRODYN, où chaque intersection donne à ses voisines une prédiction de ses flux sortants ; ou la décision de coordination peut être prise de manière centralisée par un superviseur régional, comme le fait SCATS. La coordination peut prendre différentes formes et peut être plus ou moins explicite. Par exemple, dans la stratégie PRODYN les intersections échangent des informations sur les flux de véhicules susceptibles de se rendre d'une intersection à l'autre sans plus de consignes, alors que dans [45] les intersections se coordonnent explicitement afin de créer des ondes vertes.

La décentralisation permet de découper le problème de l'optimisation du trafic sur le réseau en sous-problèmes d'optimisation locale, plus simples à résoudre et permettant de bénéficier de certains avantages. On peut par exemple citer, de manière non exhaustive, la parallélisation de certains calculs, l'économie de certaines communications en ne transmettant pas toutes les données à une entité centrale, une conception plus modulaire, ou une meilleure résilience à certaines pannes [26]. L'optimisation des sous-problèmes ne garantit pas l'optimisation du problème global, toutefois le problème de l'optimisation globale aurait une complexité bien trop importante pour être résolu de manière exacte. Le compromis trouvé par certaines approches consiste en un découpage du problème prévoyant une optimisation par intersection ou par petite zone, supervisée à une échelle supérieure afin de guider la résolution locale. Cela permet de prendre en compte des dynamiques liées aux différentes échelles

(microscopique et meso/macroscopique) en limitant la complexité computationnelle et en conservant certaines des propriétés de la distribution.

Problème d'optimisation

Certaines approches posent un problème lié à la régulation du trafic comme un problème d'optimisation et apportent une résolution exacte du problème, c'est-à-dire que la solution trouvée est mathématiquement la meilleure solution possible à ce problème. Ces approches peuvent sembler meilleures que les approches heuristiques, qui ne produisent pas nécessairement la meilleure solution à un problème, mais une solution qu'on pourra qualifier de "suffisamment bonne" en fonction de l'objectif. Toutefois ces approches heuristiques permettent souvent, par les approximations qu'elles réalisent, de traiter des problèmes plus complexes en une durée acceptable. Cela soulève la question de la modélisation du problème : une modélisation d'un problème, qu'elle soit mathématique ou informatique, contient des choix de conception qui l'éloignent du problème réel. Il est donc possible que des résultats sous-optimaux pour un problème bien modélisé s'avèrent meilleurs que des résultats optimaux sur un problème moins bien modélisé, ce qui rend souvent difficile la comparaison de différentes méthodes.

Certaines méthodes adoptent des approches gloutonnes, c'est-à-dire qu'elles réalisent une succession de choix optimaux localement mais pas nécessairement globalement et la qualité de l'optimisation réalisée est assez discutable. Par exemple [23] utilise, pour le problème du passage des véhicules, une méthode consistant à donner à chaque véhicule souhaitant traverser l'intersection le premier créneau disponible. Au contraire, apporter une coordination en retardant certains véhicules permettrait de faire passer les véhicules par groupes, et ainsi probablement d'augmenter le débit de l'intersection.

Nous avons vu que certaines méthodes résolvent des problèmes d'optimisation locaux, à l'échelle d'une intersection ou d'une petite zone du réseau. Ces approches sont en un certain sens gloutonnes car le découpage et la résolution d'un problème de régulation zone par zone ne garantit pas un résultat optimal à l'échelle globale.

1.3 Prise en compte des transports en commun dans la régulation du trafic

Le réseau routier est multi-modal, c'est-à-dire qu'il existe plusieurs modes de transport pour faire un même trajet. Cette multi-modalité a pour conséquence que les bus sont souvent soumis aux mêmes conditions de trafic que les autres véhicules, ce qui diminue leur efficacité sur les réseaux sujets à la congestion. Dans le cadre de cette thèse, nous abordons la question de la régulation des bus au sein d'un trafic urbain multimodal, et nous nous plaçons pour cela dans le cadre d'un système coopératif. Nous présentons ici le fonctionnement général des bus ainsi que différents travaux portant sur leur régulation, incluant différentes formes de priorité.

1.3.1 Fonctionnement général des bus

Du point de vue de l'exploitant du réseau de transport collectif, la régulation est un processus d'adéquation en temps réel du tableau de marche aux conditions d'exploitation [10][30]. Le tableau de marche contient l'ensemble des courses horodatées que les bus doivent réaliser. Toutefois, le maintient de ce tableau n'est pas possible dans certaines conditions de trafic et la régulation doit s'adapter à la situation. Pour cela, il existe trois logiques couramment utilisées :

- logique d'enlèvement de la charge : l'objectif est de charger un maximum de voyageurs afin de leur éviter une durée d'attente trop importante aux arrêts. Cette logique est utile lorsqu'une grande charge de voyageurs est attendue à certains points précis du réseau, par exemple près des stades à la fin d'un événement sportif important.

- logique de régularité : l'objectif est de minimiser le temps d'attente moyen des voyageurs en assurant que tous les arrêts soit desservis avec régularité. Cette logique est utile en période creuse comme dans les périodes où la charge de passagers est élevée sur une grande partie du réseau.

- logique de ponctualité : l'objectif est de respecter les horaires prévus par le tableau de marche. Cette logique est utile pour les premiers et derniers départs de la journée, dans les situations d'intermodalité, par exemple pour une ligne desservant un aéroport, ainsi que dans les situations où la fréquence de passage des transports en commun est peu élevée.

Ces différentes logiques permettant de traiter différents types de problèmes, il est possible qu'une ligne utilise successivement différentes logiques en fonction de l'heure et des lieux desservis.

1.3.2 Règles de priorité aux bus

Il existe plusieurs systèmes de priorité permettant d'avantager les bus sur un réseau. On distingue notamment deux types de priorité aux transports en commun, appelés "priorité passive" et "priorité active".

La priorité passive ne s'appuie pas sur la détection du passage des bus mais sur leur présence supposée et leur fréquence, c'est-à-dire sur le fait qu'un axe soit parcouru ou non par une ou plusieurs lignes de bus. Il s'agit, principalement, d'adapter les modalités de régulation utilisées habituellement, notamment hors-ligne, afin de prendre en compte certaines particularités des bus : ajuster le déphasage entre des intersections voisines pour que leur coordination soit plus adaptée à la vitesse des bus qu'à celle des voitures, concevoir des plans de feu réalisant une optimisation sur le nombre estimé de passagers plutôt que sur le nombre estimé de véhicules, produire des cycles de feu contenant plusieurs phases permettant de faire passer les courants utilisés par les bus, etc.

La priorité active s'appuie sur des règles se basant sur la détection de bus. Une règle minimaliste de priorité active pourrait consister à déclencher un changement de phase dès qu'un bus est détecté comme approchant de l'intersection. Les règles sont souvent plus fines, et utilisent cinq moyens d'action principaux :
- l'extension de vert pour le courant du bus
- la troncature de rouge pour le courant du bus
- l'insertion d'une phase dans le cycle de feu permettant le passage du courant du bus
- la rotation de phases permettant de réordonner les phases du cycle
- le déclenchement (ou l'insertion) de phase qui insère immédiatement une nouvelle phase au cycle

Ces cinq moyens d'action sont utilisés ensemble ou séparément par différentes stratégies, qui les exploitent ou les combinent en ajoutant certains critères : sélection de l'action la plus appropriée en fonction de la date estimée de l'arrivée du bus [6], refus de donner la priorité deux fois de suite [64], compensation du temps perdu lorsque la priorité est donnée à un autre courant [55], etc.

Un certain nombre de ces travaux portant sur les priorités aux bus ont pour conséquence une amélioration du trafic pour ces bus, néanmoins cette amélioration se fait au détriment du trafic général et suscite des retards pour les véhicules particuliers.

Bien que certains systèmes tentent de prendre en compte le trafic général en rendant la priorité conditionnelle plutôt qu'absolue, même ceux-ci dégradent la qualité du trafic général.

1.3.3 Systèmes de régulation de trafic incluant la priorité aux transports en commun

Plusieurs travaux ont intégrés des mécanismes de priorité aux systèmes de régulation du trafic général.

Un certain nombre d'approches intègrent des règles de priorité telles que celles décrites dans la section 1.3.2 à des systèmes adaptatifs de régulation du trafic général comme ceux décrits en 1.1. Par exemple, [28] étudie l'intégration de règles de priorités à SCOOT. Ces règles consistent en des extensions de vert et des troncatures de rouge sous certaines conditions, qui diffèrent selon les stratégies envisagées. D'autres systèmes, tels que SCATS, SPPORT ou TUC, utilisent également ce type de priorité [10].

Certains systèmes intègrent des variables directement liées à la circulation des transports en commun dans leur stratégie d'optimisation. Par exemple dans [52], le système RHODES présenté en 1.1.4 a intégré un module appelé BUSBAND dans lequel le bus est traité comme un véhicule particulier comme les autres, à ceci près que celui-ci dispose d'un poids différent dans la stratégie d'optimisation adoptée par RHODES. Selon les versions de RHODES/BUSBAND, le poids du bus peut-être fixé à une valeur élevée ou ajusté dynamiquement en fonction du nombre de passagers du bus et du retard de celui-ci.

Certains travaux sont allés plus loin avec des approches de régulation pour le trafic général prenant en compte la problématique des bus dès le départ. Cependant les formes prises par ces approches ressemblent à celles utilisées par les approches décrites précédemment. Par exemple le système DARVIN [25] réalise une minimisation de la somme du retard des véhicules, pondérée en fonction de leur importance. Il s'appuie pour cela sur une optimisation à partir d'algorithmes génétiques. Cependant une originalité majeure de ce système est qu'il détermine, pour chaque phase, la date de début et la date de fin de cette phase.

Une stratégie de régulation bimodale, NeTPrior (Network Transit Priority) a été développée [5]. Elle consiste en une « commande linéaire quadratique » qui a pour objectif d'éliminer la congestion dans les arcs où circulent les véhicules de transport en commun à l'instant de leurs présences sur les arcs, ce qui la distingue des systèmes de régulation donnant une priorité passive aux véhicules de transport en commun.

Ces différents types de systèmes prennent en compte la priorité aux transports en commun dans le cadre d'une régulation adaptative du trafic général. Pourtant bien que dans ces deux types de systèmes le temps de trajet des véhicules des tronçons adjacents est meilleur qu'avec un plan de feu fixe, il reste plus élevé qu'avec un système sans priorité. De plus toutes ces approches ne prennent que peu en compte les logiques de régulation auxquelles sont successivement soumis les bus (enlèvement de la charge, régularité, ponctualité).

[43], [44] ont présenté une stratégie de régulation multi-objectifs qui vise d'une part, la régulation du trafic général et d'autre part, la régularité des bus. Cette stratégie est basée sur un modèle événementiel du trafic bi-modal et utilise une optimisation par essaims particulaires. L'objectif de régulation consistait à minimiser d'une part, le nombre de véhicules dans le réseau et d'autre part, l'écart quadratique entre les positions réelles des véhicules de transport en commun et des positions de référence à atteindre.

[5] se sont également intéressé à ce problème. Ce travail cherche à améliorer le trafic global sur le réseau ainsi que la régularité entre les véhicules de transport en commun mais en utilisant cette fois-ci une modélisation multi-agents. Quatre types d'agents sont utilisés, deux au niveau microscopique et deux au niveau macroscopique. Au niveau microscopique, un agent « phase » calcule la durée de vert nécessaire pour évacuer les véhicules sur les arcs du carrefour pour chaque cycle de feu et un agent « bus » permettant d'assurer l'évolution des véhicules de transport en commun dans le réseau. Au niveau macroscopique, un agent « ligne bus » veille à la régularité entre les véhicules de transport en commun et un agent « intersection » permet de gérer le conflit entre les différentes phases d'un même carrefour et la priorité des bus. Cette priorité est obtenue par un mécanisme de communication et de négociation entre les agents phase et les agents intersection d'une part, et entre les agents bus et les agents intersection d'autre part.

Ces systèmes sont généralement centralisés et n'exploitent pas tous les apports des nouveaux moyens de communication entres véhicules d'une part et véhicules et infrastructures de l'autre part.

1.4 Conclusion

Nous avons vu dans cette section que le problème de la régulation du trafic était un problème important par ses enjeux, et difficile puisqu'il implique plusieurs variables et des phénomènes complexes.
Nous avons également vu que les méthodes issues de l'intelligence artificielle, et en

particulier des systèmes multi-agents, offrent de nouvelles possibilités quant aux formes qu'est susceptible de prendre la régulation et aux méthodes de calcul utilisables. Ceci est renforcé par l'évolution du niveau d'équipement des environnements routiers urbain, en particulier des véhicules qui embarquent de plus en plus de dispositifs permettant le calcul et/ou la communication.
Enfin, nous avons vu que la gestion du trafic pour les transports en commun est également une question difficile qui apporte ses propres problématiques à celle du trafic général, et que les approches obtenant de meilleurs résultats abordent ces deux aspects, la gestion du trafic général et la gestion des bus, simultanément.

Des avancées dans le domaine peuvent être obtenues en incluant les apports des nouveaux moyens de communication dans les véhicules et sur les infrastructures.

2

Négociation du droit de passage à l'intersection

Le problème de la congestion est un problème difficile qu'on ne peut pas résoudre uniquement par le renforcement des infrastructures existantes, qui serait très coûteux. Différentes méthodes de contrôle du trafic ont été développées durant les dernières décennies afin d'optimiser l'utilisation des réseaux existants. L'intersection étant une zone de conflit entre les véhicules à l'origine d'importants ralentissements, une grande part des systèmes de contrôle du trafic urbain se focalise donc sur la régulation au niveau de l'intersection, optimisant le droit de passage des véhicules aux intersections à feux.

Durant les années 1990 et 2000, l'intelligence artificielle a permis d'envisager de nouvelles méthodes pour la modélisation et la régulation du trafic, particulièrement à l'aide des méthodes multi-agents qui permettent de résoudre différents problèmes de manière décentralisée et/ou distribuée [8]. Les technologies de communication actuelles permettent la conception de méthodes s'appuyant sur une communication en temps réel afin d'obtenir des informations précises. Chaque véhicule sur un réseau dispose d'un contexte différent, et l'information constituant ce contexte est potentiellement utile pour réaliser une régulation plus efficace : le retard accumulé par le véhicule depuis le début de son trajet, sa position actuelle, ses intentions à court et long termes, etc.

Grâce à cette importante quantité d'informations, de nouvelles stratégies permettent une régulation du trafic à l'échelle de l'intersection isolée [23]. Certaines stratégies réalisent un contrôle à l'échelle du réseau [57] et d'autres se concentrent sur la coordination de plusieurs intersections successives en constituant ce qu'on appelle des “ondes vertes" [19]. Les ondes vertes permettent de réduire le nombre d'arrêts-redémarrages des véhicules, à l'origine de pertes de temps importantes. L'efficacité des ondes vertes dans la régulation classique souligne l'importance de la conception de mécanismes permettant la coordination à l'échelle de plusieurs intersections.

Dans [23], K. Dresner et P. Stone proposent un mécanisme d'allocation du droit de passage fondé sur la réservation pour les véhicules autonomes. Celui-ci s'appuie sur une politique appelée FCFS (First Come First Served, premier arrivé premier servi), accordant le droit de passage à chaque véhicule en faisant la demande,

aussi tôt que possible. Le mécanisme présenté prend en compte les conducteurs humains en utilisant une politique classique à base de feux tricolores, et les véhicules autonomes à qui il accorde des réservations durant la phase de rouge en utilisant FCFS [24]. Bien que ce mécanisme supporte la présence de conducteurs humains, les améliorations de la régulation réalisée sont dues à l'utilisation de FCFS pour les véhicules autonomes.

Dans ce chapitre, nous proposons un mécanisme d'attribution du droit de passage à l'échelle de l'intersection, qui met en oeuvre deux aspects complémentaires. Premièrement, nous prenons en compte le contexte de trafic afin de prendre des décisions sur la base d'informations plus précises : le contexte global (informations à l'échelle du réseau) et le contexte individuel des véhicules (historique, informations sur l'état courant, intentions) sont des informations utiles qui peuvent être utilisées pour produire une politique de régulation juste et efficace. Deuxièmement, afin de favoriser la coopération des véhicules, ceux-ci prennent les décisions de régulation par eux-mêmes ce qui permet de prendre en compte efficacement la grande quantité et la grande variété des informations disponibles en termes de temps de calcul. Pour atteindre ces objectifs, nous proposons une méthode de régulation s'appuyant sur un mécanisme de négociation automatique réalisé par des agents intelligents représentant les intérêts des véhicules. Notre mécanisme doit amener les véhicules à atteindre une décision collective dans laquelle chaque véhicule peut mettre en avant ses propres contraintes et connaissances individuelles, suggérer des solutions et prendre part à la décision en temps réel. Un tel mécanisme temps réel d'attribution du droit de passage permet de prendre en compte efficacement les véhicules autonomes et les conducteurs humains disposant d'un véhicule équipé pour ce type de communication.

Une partie fondamentale de nos travaux consiste à conceptualiser les interactions multilatérales en termes d'intérêts individuels et collectifs. Dans ce but, nous proposons dans ce chapitre un nouveau cadre de négociation pour une régulation de trafic orientée agents et abordons les difficultés liées au traitement d'un trafic en flux continu. Dans de telles négociations, les véhicules construisent différentes propositions d'attribution du droit de passage que nous appelons “configurations". Ces configurations sont présentées aux autres véhicules présents dans le voisinage de l'intersection, qui peuvent apporter des arguments sur les avantages et les inconvénients de chaque configuration. Avec l'aide de l'intersection, qui contribue à la coordination des interactions, les véhicules décident collectivement de la configuration à adopter. Le mécanisme présenté dans ce chapitre a fait l'objet de plusieurs publications : [31], [32], [33], [34] et [35].

La suite de ce chapitre est organisée ainsi : La section 2.1 présente le modèle d'intersection pour lequel nous avons opté, et le problème d'attribution du droit de passage

tel qu'il se présente physiquement à l'intersection. La section 2.2 détaille la méthode utilisée par les agents pour construire des propositions de configurations en modélisant le problème sous la forme d'un CSP (problème de satisfaction de contraintes). La section 2.3 présente le mécanisme de négociation permettant aux véhicules de prendre une décision collective à partir de leurs propositions individuelles. Cette section introduit le problème posé par la continuité du trafic et détaille la manière dont ce problème est géré par les agents.

2.1 Description de l'intersection et modélisation du problème

Le problème que nous présentons dans ce chapitre consiste à attribuer une date d'admission à chaque véhicule approchant d'une intersection. Cette date est définie comme un créneau durant lequel le véhicule a le droit de s'engager dans la zone de conflit de l'intersection afin de la traverser. Une configuration doit être capable de permettre un trafic efficace et de respecter les différentes contraintes physiques et de sécurité, tout en prenant en compte le contexte du trajet de chaque véhicule ainsi que le contexte global de trafic. Un modèle multi-agents est défini dans lequel tous les véhicules et les intersections sont modélisés par des agents.

La représentation physique du réseau est faite par un modèle s'appuyant sur un automate cellulaire. Ce type de modèle est largement utilisé dans la littérature parce qu'il permet de conserver les principales propriétés d'un réseau tout en étant relativement simple à utiliser [13] [49]. Bien qu'il existe des outils plus avancés tels qu'ARCHISIM [21] ou VISSIM [27] pour effectuer une telle modélisation, nous avons opté pour un modèle de ce type car il conserve les principales propriétés d'un réseau tout en restant simple, ce qui permet un traitement facile de l'état de l'environnement.

Comme expliqué dans le chapitre précédent, l'intersection est entre autres composée de plusieurs voies d'entrée, appelées “approches", et d'une zone centrale appelée “zone de conflit". Dans ce chapitre, on appelle “trajectoire" le chemin du véhicule à travers l'intersection. Chaque approche ou trajectoire est une succession de cellules (cf. Figure 2.1). Une cellule en dehors de la zone de conflit appartient à exactement une voie. Une cellule à l'intérieur de la zone de conflit peut appartenir à une ou plusieurs trajectoires. Dans ce second cas, cette cellule est appelée “point de conflit".

Les règles de déplacement des véhicules sont les suivantes :
(1) Si un véhicule est dans la cellule avant d'une approche, ce véhicule avance et

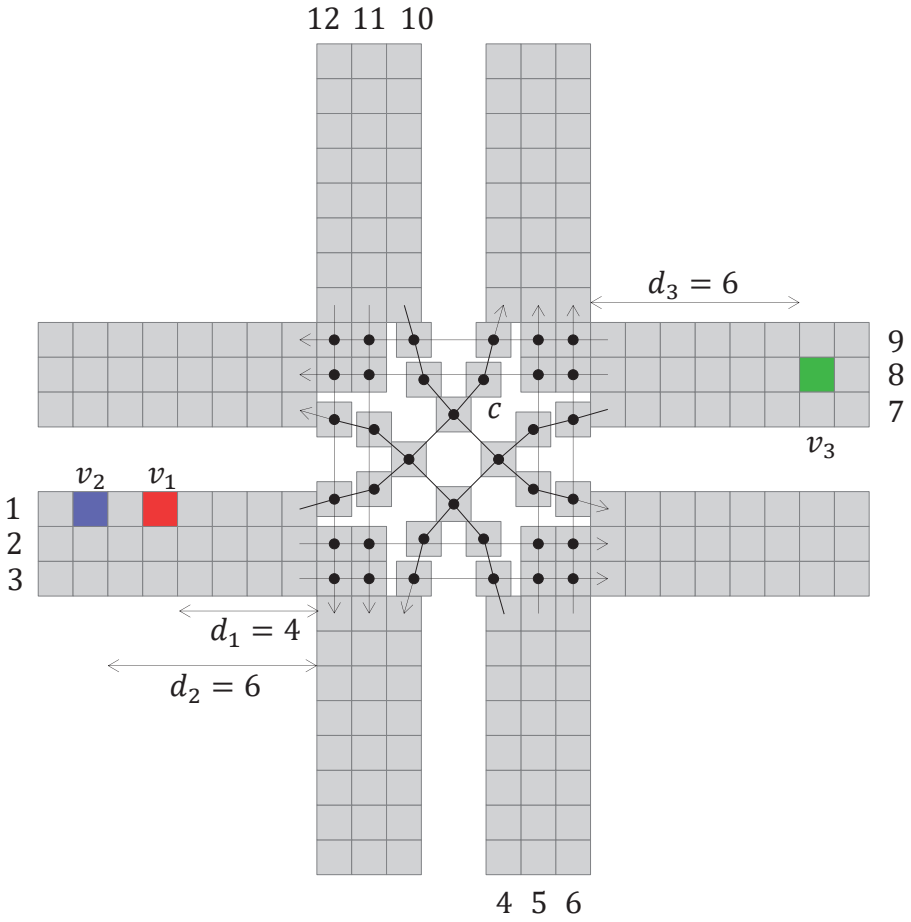


Figure 2.1: Intersection avec 12 approches et 12 voies de sortie, composées de cellules. Les approches sont numérotées de 1 à 12. La zone de conflit est traversée par différentes trajectoires, également composées de cellules. Les cellules appartenant à différentes trajectoires (toutes les cellules de la zone de conflit dans cet exemple) sont des points de conflit. Les cellules colorées sont occupées par un véhicule. Par exemple le rectangle annoté v_1 sur l'approche 1 est un véhicule en provenance de l'Ouest, sur le point de traverser l'intersection en direction du Nord.

s'engage dans la zone de conflit si et seulement s'il a le droit de passage.
(2) Si un véhicule est sur une approche, il avance si et seulement si la cellule de destination est vide ou devient vide durant ce pas de temps.
(3) Si un véhicule est dans la zone de conflit, il avance nécessairement. Notre méthode doit donc garantir pour chaque véhicule que celui-ci ne rencontrera aucun autre véhicule sur les cellules de sa trajectoire.
Lorsqu'un véhicule avance, il avance nécessairement d'une cellule par pas de temps.

La décision est distribuée : chaque agent est capable de raisonner et de communiquer avec l'intersection et les autres véhicules. L'agent intersection prend part à la coordination des interactions des véhicules. Lorsqu'on cherche à proposer un mécanisme permettant aux véhicules de prendre une décision de manière distribuée, on peut choisir entre deux approches :
(1) Les agents construisent individuellement des configurations complètes puis débattent et décident collectivement sur ces configurations.
(2) Les agents construisent des solutions partielles basées sur leurs contraintes individuelles, puis fusionnent ces solutions partielles.

Les différentes dates d'admission constituant une configuration étant fortement interdépendantes du fait des différentes contraintes physiques et de sécurité, fusionner des solutions partielles serait une tâche complexe nécessitant de nombreuses interactions répétées entre les agents, avec de nombreux échanges de messages, ce qui ralentirait le processus de décision. Pour cette raison, nous avons décidé d'opter pour une approche dans laquelle les véhicules construisent individuellement des configurations complètes puis décident collectivement du choix final avec un processus de négociation. Avant d'aborder l'étape de négociation, les agents doivent modéliser le problème d'attribution du droit de passage afin de construire les configurations qui feront l'objet des négociations.

Notre mécanisme doit satisfaire les propriétés suivantes :

- **Distribution et autonomie :** Les véhicules sont considérés comme des agents qui ont leurs propres connaissances, buts et capacités de calcul.

- **Réalisme des capteurs :** Les connaissances de chaque véhicule doivent correspondre au type et à la quantité d'informations qu'ils peuvent obtenir de manière réaliste par leurs capteurs et leurs capacités de communication.

- **Faible coût calculatoire et communicationnel :** Le mécanisme doit être appliqué en temps réel. Les véhicules doivent donc prendre leurs décisions

en temps réel, ce qui suppose que les calculs et les communications sont suffisamment courts.

- **Prévention des interblocages :** Sur un réseau routier, les situations d'interblocage sont les situations dans lesquelles plusieurs véhicules se gênent mutuellement empêchant tout déplacement. Ces situations pénalisent très fortement le trafic et doivent donc être évitées.

- **Unicité de l'accord :** Si les agents ne partagent pas le même accord sur la configuration choisie par l'intersection, les véhicules peuvent s'engager dans la zone de conflit sans qu'aucun contrôle de sécurité ne soit fait, ce qui est susceptible de créer des collisions.

- **Configuration "anytime" :** Même si un agent subit des ralentissements computationnels ou des retards de communication, le véhicule correspondant à cet agent continue sa progression physique. Lorsqu'un processus de décision en cours n'a pas le temps d'être terminé, tous les agents doivent connaitre la configuration de l'intersection afin de savoir quel comportement le véhicule doit appliquer.

2.2 Modélisation du problème d'attribution du droit de passage et construction des configurations

Afin de construire des configurations, nous modélisons le problème d'attribution du droit de passage comme un CSP (Constraint Satisfaction Problem, problème de satisfaction de contraintes [46]). Le CSP convient à notre problème puisqu'il permet de modéliser facilement les contraintes structurelles (contraintes de sécurité et contraintes physiques).

Définition Soit V l'ensemble des véhicules approchant une intersection, et t_{cur} la date courante. Une configuration est un ensemble $c = \{t_1, ..., t_k\}$ où chaque t_i est la date d'admission au sein de la zone de conflit accordée à $v_i \in V$.

Une configuration possible pour le scénario présenté Figure 2.1 est la suivante :

Pour chaque $v_i \in V$, app_i est l'approche sur laquelle se situe v_i, d_i est la distance (en nombre de cellules) entre v_i et la zone de conflit, $traj_i$ est la trajectoire de v_i au sein de la zone de conflit. T est l'ensemble des trajectoires au sein de la zone de conflit. $pos(cell_1, traj)$ est la distance, en nombre de cellules, entre la cellule $cell_1 \in traj$ et

V	c
v_1	5
v_2	11
v_3	10

Table 2.1: Exemple de configuration

le début de la zone de conflit sur la trajectoire $traj$ (la première cellule de la zone de conflit a la position 0). sp est la vitesse des véhicules en cellules par pas de temps. Dans notre modèle, $sp = 1$ cellule/pas de temps.

Nous identifions 3 types de contraintes structurelles pour les véhicules, basées sur les règles suivantes :

R1. Règle de distance Un véhicule doit traverser la distance le séparant de la zone de conflit avant de s'y engager. Nous avons : $\forall v_i \in V, t_i > t_{cur} + \frac{d_i}{sp}$

R2. Règle d'antériorité Un véhicule ne peut s'engager dans la zone de conflit qu'après les véhicules qui le précèdent sur sa voie (cette règle pourrait être supprimée dans le cadre d'un modèle plus complexe qui prendrait en compte les dépassements). Nous avons :
$\forall v_i, v_j \in V^2, app_i = app_j, d_i < d_j \Rightarrow t_i < t_j$

R3. Règle de conflit Deux véhicules ne peuvent pas être simultanément dans la même cellule. Si ces véhicules appartiennent à la même voie ou la même trajectoire, les règles de déplacement préviennent cette situation. Cependant, si la cellule est un point de conflit alors il nous faut modéliser cette règle dans le cas de véhicules appartenant à différentes trajectoires partageant au moins une cellule. Dans une première version, nous avons :
$\forall v_i, v_j \in V^2, \forall cell_1 \in traj_i,$
$cell_1 \in traj_j \Rightarrow (t_i + \frac{pos(cell_1, traj_i)}{sp}) \neq (t_k + \frac{pos(cell_1, traj_j)}{sp})$.
Cette règle peut être renforcée pour des raisons de sécurité. En effet, ajouter un délai de sécurité t_{safe} entre le passage d'un véhicule sur une cellule $cell_1$ et le passage d'un autre véhicule ayant une trajectoire différente dans cette même cellule permet de diminuer les risques de collision (t_{safe} étant fixé de manière experte). La règle de conflit complète est donc la suivante :
$\forall v_i, v_j \in V^2, \forall cell_1 \in traj_i,$
$cell_1 \in traj_j \Rightarrow \left|(t_i + \frac{pos(cell_1, traj_i)}{sp}) - (t_k + \frac{pos(cell_1, traj_j)}{sp})\right| > t_{safe}$

Définition Une configuration c est valide si et seulement si c respecte les trois règles R1, R2 et R3, et :
$\forall v_i \in V, \exists t_i \in c$, où chaque t_i est la date d'admission de v_i.

Le scénario représenté sur la figure 2.1 illustre ces trois types de contraintes structurelles. Nous considérons trois véhicules v_1, v_2, v_3 approchant de l'intersection à $t_{cur} = 0$. Afin d'assurer une sécurité appropriée, R3 est appliquée avec un délai de sécurité $t_{safe} = 2$. R1 génère une contrainte par véhicule : v_1, v_2 et v_3 dans ce scénario. R2 génère une contrainte par paire de véhicules partageant la même approche : (v_1, v_2) dans ce scénario. R3 génère une contrainte par paire de véhicules ayant des trajectoires conflictuelles : (v_1, v_3) et (v_2, v_3) dans ce scénario. Nous avons donc 6 contraintes :
- **R1** $(ct_1)\ t_1 > 4$; $(ct_2)\ t_2 > 6$; $(ct_3)\ t_3 > 6$
- **R2** $(ct_4)\ t_2 > t_1$
- **R3** $(ct_5)\ |(t_1 + 4) - (t_3 + 2)| > 2$; $(ct_6)\ |(t_2 + 4) - (t_3 + 2)| > 2$

À partir de cette modélisation sous la forme d'un CSP, un agent utilise un solveur afin de trouver des dates d'admission compatibles (i.e., qui respectent les contraintes ci-dessus) pour un ensemble $V^{neg} \subseteq V$ de véhicules approchant de l'intersection. Pour toute configuration c, $\forall v_i \in V^{neg}, \exists d_i \in c$ tel que d_i respecte les contraintes structurelles ci-dessus. Plusieurs configurations possibles peuvent exister pour une situation donnée, et il peut arriver que les agents produisent des configurations sous-optimales.

Un véhicule a initialement des perceptions limitées, cependant il est capable de connaitre en temps réel la position des véhicules approchant de l'intersection. Ces travaux prenant place dans le cadre des approches coopératives des systèmes de transport intelligents ([1],[18]), chaque véhicule a un comportement coopératif avec l'intersection et communique sa trajectoire lorsqu'il s'approche de celle-ci.

Grâce à ses capacités de calcul et aux informations disponibles, un véhicule peut utiliser un solveur pour produire des configurations. Les solveurs CSP actuels permettent d'intégrer une fonction objectif à minimiser ou maximiser, il est donc possible d'orienter la recherche de configurations par le solveur. De plus, un agent peut introduire des contraintes additionnelles à son solveur en tant que lignes directrices. Si un agent estime que l'ajout d'une contrainte particulière est susceptible de séparer l'espace de recherche de manière pertinente, il peut l'appliquer comme heuristique de recherche. Cependant une telle contrainte n'est pas structurelle, elle n'est utilisée par l'agent que de manière heuristique et est donc susceptible d'être violée. Le choix de cette fonction objectif et l'éventuel ajout de contraintes additionnelles en tant que lignes directrices sont des possibilités dépendant de la stratégie de chaque agent véhicule.

Exemple Considérons un bus b_1 et un véhicule v approchant d'une intersection. v_1 et b ont des trajectoires conflictuelles. Plusieurs autres véhicules sont présents sur

toutes les approches de l'intersection, on peut donc être confronté à de nombreuses contraintes structurelles sur les configurations et l'espace de recherche peut être plus complexe à explorer. Les véhicules considèrent que le bus a une forme de priorité. v_1 estime qu'une bonne heuristique pour trouver des configurations pertinentes (d'après son utilité individuelle et/ou pour le bien-être social) consiste à permettre un passage rapide à b_1 (en dessous d'un seuil t_{quick}), et de rechercher des configurations acceptables dans cet espace de recherche réduit. v_1 guide donc sa recherche en ajoutant à son solveur la contrainte $t_{b_1} \leq t_{quick}$, où t_{b_1} est la date d'admission de b_1 et t_{quick} correspond à ce que v_1 considère être une date d'admission rapide pour le bus.

2.3 Modèle de négociation du droit de passage

Chaque véhicule construit une ou plusieurs configurations lui permettant de traverser l'intersection, cependant seule une configuration sera appliquée à une date donnée. Un processus de négociation a lieu pour la choisir. Le mécanisme que nous proposons s'appuie sur un modèle d'argumentation [3]. À travers ce processus de négociation, les agents cherchent à atteindre un accord collectif en faisant des concessions. Pour réaliser une négociation, l'agent véhicule s'appuie sur son propre état mental fondé sur ses connaissances, ses buts et ses préférences (cf Fig. 2.2). Cet état mental évolue durant la négociation. Les agents utilisent des arguments pour modifier les états mentaux des autres agents et pour faire évoluer leur propre état mental afin d'atteindre un meilleur compromis pour chacun.

Chaque agent a_i manipule les bases suivantes :

$\mathcal{K}_i$ est la base de connaissances de a_i sur son environnement. Cette base regroupe des croyances incertaines, donc chaque croyance $k_i^l \in \mathcal{K}_i$ a un niveau de certitude ρ_i^l.

$\mathcal{KO}_i$ est la base de connaissances de a_i sur les autres véhicules. Chaque $ko_i^j \in \mathcal{KO}_i$ représente une base contenant les connaissances estimées de a_j d'après a_i. Chacune de ces croyances $ko_i^{j,l} \in ko_i^j$ a un niveau de certitude $\delta_i^{j,l}$.

$\mathcal{G}_i$ est la base des buts de a_i. Ces buts ont différents niveaux de priorité, donc chaque but $g_i^l \in \mathcal{G}_i$ a un niveau de priorité λ_i^l.

$\mathcal{GO}_i$ est la base des buts que a_i suppose pour les autres véhicules. Chaque $go_i^j \in \mathcal{GO}_i$ est une base contenant les buts estimés de a_j, d'après a_i. Chacune de ces croyances $go_i^{j,l} \in go_i^j$ a un niveau de certitude $\gamma_i^{j,l}$.

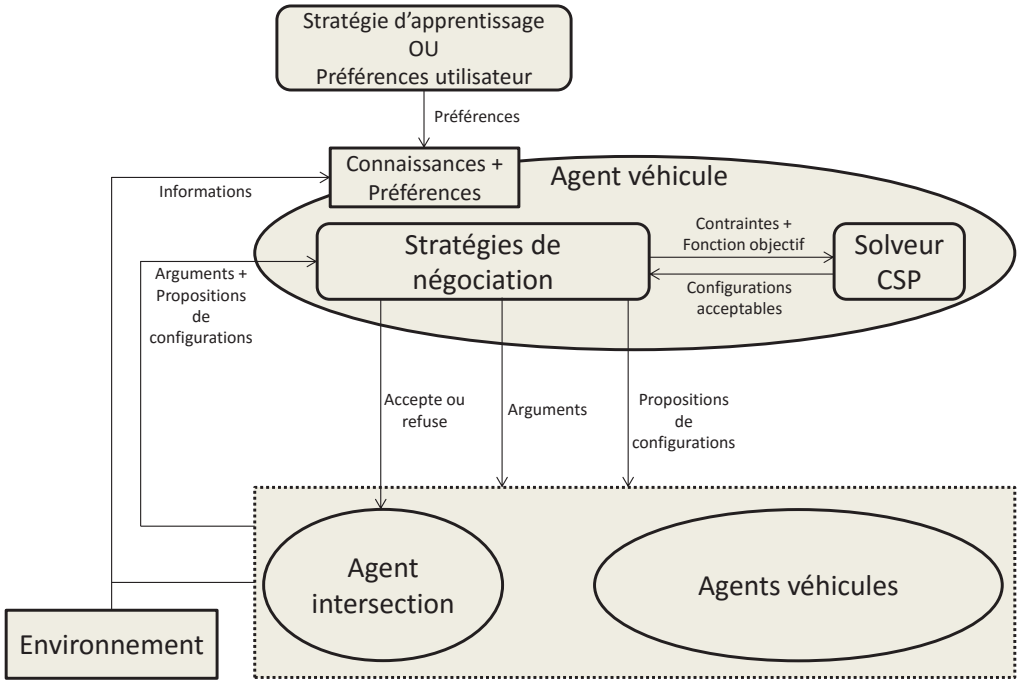


Figure 2.2: Architecture des agents véhicules et du système de négociation

L'initialisation et la mise-à-jour de telles bases dépendent en grande partie des différents types d'arguments reconnus par le système et des connaissances qui peuvent être introduites de manière experte dans le système ou qui peuvent faire par exemple l'objet d'un apprentissage automatique.

À titre d'exemple, un véhicule v_i approchant d'une intersection sait qu'il est suivi par plusieurs véhicules, dont un bus b. Il ajoute donc la connaissance $k_i^1 : voiechargee$ à sa base de connaissances avec un niveau de certitude ρ_i^1 dépendant de la fiabilité de ses capteurs. Il ajoute également la connaissance $k_i^2 : busenapproche$ avec un niveau de certitude ρ_i^2 déterminé de la même manière.

Par ailleurs, il attribue à chaque véhicule v_j de son entourage des connaissances k_j^1 et k_j^2 similaires à k_i^1 et k_i^2 avec un facteur décroissant en fonction de la distance $dist_i^j$:
$\delta_i^{j,l} = dist_i^j * \rho_i^l$.

v_i suppose que le bus a besoin d'une date de passage moyenne t_{av} compte tenu de la distance entre b et l'intersection et du niveau de congestion général du réseau (le bus est plus susceptible d'être en retard si le réseau est congestionné). On suppose, dans cet exemple, que v_i a une approche parfaitement coopérative et souhaite favoriser le bus si nécessaire, il initialise donc un but t_b, avec $t_b \leq t_{av}$. Cependant ce n'est que

supposition, le bus b peut être plus en retard ou plus en avance que ce que suppose v_i. Ce but aura donc un poids λ_i^1 bas.

Enfin, pour chaque véhicule v_j de son entourage, il attribue des buts déterminés de manière similaire, par exemple en s'appuyant sur le niveau de coopération $coop_i^j$ que v_i attribue à v_j :
$\gamma_i^{j,l} = coop_i^j * \lambda_i^l$

Ces bases sont mises à jour de différentes manières, en premier lieu du fait que l'environnement évolue avec le temps ce qui permet à v_i de mettre à jour $\mathcal{K}$. De plus, chaque échange d'arguments avec les autres agents au cours des différentes itérations de la négociation permet à v_i de mettre à jour ses bases, pour différentes raisons :

- Si v_i apprend que le bus a pour objectif de franchir l'intersection avant la date t_{goal}, il mettra à jour g_i^1 et chaque $g_i^{j,1}$ pour remplacer le but $t_b \leq t_{av}$ par $t_b \leq t_{goal}$.

- En recevant les arguments des autres véhicules, il lui est possible d'inférer leurs buts. Par exemple, si lors d'une itération v_i reçoit un argument en faveur du but $g_i^{j,1}$ de la part de v_j, il est vraisemblable que ce but soit bien un but réel de v_j, donc v_i revoit la valeur de $\gamma_i^{j,1}$ à la hausse.

- Si v_i reçoit un argument en faveur d'un but absent de g_i^j, celui-ci peut y être intégré avec un poids $\gamma_i^{j,l}$ relatif à la force de l'argument.

- De la même manière, les connaissances sur lesquels s'appuient les arguments des autres agents permet de mettre à jour $\mathcal{KO}$ en intégrant des connaissances absentes de k_i^j.

- Lorsqu'un argument est émis dans la négociation, les connaissances sur lesquelles s'appuie cet argument peuvent être prises en compte dans les bases de tous les agents. Premièrement, elles peuvent être intégrées à $\mathcal{K}$ avec un poids initial ρ_i^l bas qui augmentera avec la récurrence de l'apparition d'arguments s'appuyant sur cette connaissance dans la négociation. Deuxièmement, elles peuvent être ajoutées à $\mathcal{KO}$ et mises à jour de la même manière.

Cet exemple ne présente qu'un échantillon de ce que ces bases $\mathcal{K}, \mathcal{KO}, \mathcal{G}, \mathcal{GO}$ permettent de représenter, et les choix de conception réalisés par chaque implémentation nécessitent d'intégrer différents types de connaissances.

Comme nous le détaillons dans la prochaine section, chaque véhicule a par ailleurs un poids attribué par les intersections. De plus, deux sortes d'arguments peuvent être utilisés par les agents, les arguments favorables et les arguments défavorables.

Définition Un argument en faveur d'une décision de configuration d est un quadruplet $A =< Supp, Cons, d, w_A >$ où $Supp$ est le support de l'argument A, $Cons$ représente ses conséquences, w_A est le poids de l'argument (fixé par le véhicule v_i qui produit cet argument et a un poids w_i), tel que :
- $d \in \mathcal{D}$, $\mathcal{D}$ étant l'ensemble des décisions possibles
- $Supp \subseteq \mathcal{K}^*$ et $Cons \subseteq \mathcal{G}^*$
- $Supp \cup \{d\}$ est consistant
- $Supp \cup \{d\} \vdash Cons$
- $Supp$ est minimal et $Cons$ est maximal (pour l'inclusion) parmi les ensembles qui satisfont les conditions ci-dessus.
- $0 \leq w_A \leq w_i$

Exemple Considérons un bus b_1 proposant une configuration c_1 lui permettant de traverser l'intersection sans perdre de temps afin de rattraper un retard. Un véhicule v_1 précède ce bus sur la même voie. Accorder une date d'admission rapide à b_1 (en dessous d'un seuil t^b_{quick}) implique de donner une date d'admission rapide à v_1 (en dessous d'un seuil t^v_{quick}), et un des buts de v_1 est de traverser l'intersection aussi vite que possible. Ainsi :
$\mathcal{K}_{v_1} = \{crossesQuickly(b_1) \rightarrow crossesQuickly(v_1)\}$
$\mathcal{G}_{v_1} = \{crossesQuickly(v_1)\}$
v_1 peut tirer avantage de la configuration c_1, il produit donc l'argument suivant :
$< \{crossesQuickly(b_1), crossesQuickly(b_1) \rightarrow$
$crossesQuickly(v_1)\}, \{crossesQuickly(v_1)\}, c_1 >$.

Définition Un argument en défaveur d'une décision de configuration d est un 4-uplet $A =< Supp, Cons, d, w_A >$ où $Supp$ est le support de l'argument A, $Cons$ représente ses conséquences, w_A est le poids de l'argument (fixé par le véhicule v_i qui produit cet argument est a un poids w_i), tel que :
- $d \in \mathcal{D}$, $\mathcal{D}$ étant l'ensemble des décisions possibles
- $Supp \subseteq \mathcal{K}^*$ et $Cons \subseteq \mathcal{G}^*$
- $Supp \cup \{d\}$ est consistant
- $\forall g_i \in Cons, Supp \cup \{d\} \vdash \neg g_i$
- $Supp$ est minimal et $Cons$ est maximal (pour l'inclusion) parmi les ensembles qui satisfont les conditions précédentes
- $0 \leq w_A \leq w_i$

Exemple Considérons un bus b_2 proposant une configuration c_2 lui permettant de traverser l'intersection dès que possible afin de rattraper un retard. Un véhicule v_2 a une trajectoire conflictuelle avec b_2. Accorder une date d'admission rapide à b_2 (en dessous d'un seuil t^b_{quick}) implique de donner une date d'admission lente à v_2 (au-dessus d'un seuil t^v_{quick}). Un des buts de v_2 est de traverser l'intersection aussi vite que possible. Ainsi :
$\mathcal{K}_{v_2} = \{crossesQuickly(b_2) \rightarrow \neg crossesQuickly(v_2)\}$
$\mathcal{G}_{v_2} = \{crossesQuickly(v_2)\}$
La configuration c_2 peut désavantager v_2, il produit donc l'argument suivant :
$< \{crossesQuickly(b_2), crossesQuickly(b_2) \rightarrow$
$\neg crossesQuickly(v_2)\}, \{crossesQuickly(v_2)\}, c_2 >$.

Pour des raisons de sécurité, l'intersection n'a qu'une seule configuration courante à la fois. Les agents utilisent cette configuration comme point de départ pour la négociation. Dans cette négociation, le but d'un agent est de remplacer la configuration courante c_{cur} par une autre configuration c_{best} qui améliore son utilité individuelle. Pour mener cette négociation, les agents s'appuient sur un langage de communication spécifique pour interagir. L'ensemble des actes de communication disponibles est le suivant : $Acts = \{Offer, Argue, Accept, Refuse\}$.

Offer(c_{new}, c_{cur}) : avec cet acte, un agent propose qu'une configuration c_{new} remplace la configuration courante c_{cur}. Un agent ne peut soumettre une offre qu'une seule fois.

Argue$(c, arg(c))$: avec cet acte, un agent émet un argument en faveur ou en défaveur de c.

Accept(c_{new}, c_{cur}) : avec cet acte, un agent accepte qu'une configuration c_{new} remplace la configuration courante c_{cur}.

Refuse(c_{new}, c_{cur}) : avec cet acte, un agent refuse qu'une configuration c_{new} remplace la configuration courante c_{cur}.

c_{new} est acceptée si et seulement si :
$\frac{\sum_{v_i \in V(c_{new})} w_i}{\sum_{v_i \in V^{neg}} w_i} \geq th_{accept}$, où :
th_{accept} est un seuil d'acceptation ($th_{accept} > 0.5$).
$V(c_{new}) \subseteq V^{neg}$ est l'ensemble des véhicules acceptant que la configuration $c_{new} \in \mathcal{D}$ remplace c_{cur}. w_i est le poids donné par les intersections au véhicule v_i. Quand une configuration est acceptée, cette configuration devient la configuration courante de l'intersection.

Exemple Considérons deux véhicules v_1 et v_2 qui s'approchent d'une intersection. La configuration initiale de l'intersection est c_0. Avec son solveur, v_1 génère deux propositions c_1 et c_2. La configuration c_1 n'étant pas sa préférée, il ne la propose pas dans la négociation pour l'instant et la conserve si besoin pour plus tard. Il propose donc la configuration c_2. Pour cela, il utilise l'acte $Offer(c_2, c_0)$ qu'il transmet à v_2. Pour appuyer cette offre, il apporte un argument $arg_1(c_2)$ en faveur de c_2 avec l'acte $Argue(c_2, arg_1(c_2))$. v_2 apporte un argument $arg_2(c_2)$, cette fois en défaveur de c_2, avec l'acte $Argue(c_2, arg_2(c_2))$.

Supposons dans un premier temps que l'argument $arg_1(c_2)$ a un poids plus fort que $arg_2(c_2)$, et que les deux véhicules n'ont plus d'arguments sur cette proposition, v_1 accepte alors sa propre proposition avec $Accept(c_2, c_0)$ puisqu'elle lui est favorable et dispose d'arguments globalement positifs. Si v_2 accepte également, alors la proposition est acceptée, c_2 remplace c_0 en tant que configuration courante de l'intersection.

Supposons maintenant que l'argument $arg_2(c_2)$ a un poids plus fort que $arg_1(c_2)$, et que les deux véhicules n'ont plus d'arguments sur cette proposition, v_1 peut par exemple refuser cette proposition s'il souhaite avoir un comportement coopératif et ne pas voter pour une proposition ayant des arguments globalement négatifs, et que v_2 l'accepte. v_1 et v_2 utilisent respectivement les actes $Refuse(c_2, c_0)$ et $Accept(c_2, c_0)$. v_2 a un poids plus fort et la proposition est acceptée, c_2 remplace c_0 en tant que configuration courante de l'intersection.
Simultanément, v_1 a envoyé l'acte $Offer(c_1, c_0)$ car la proposition c_1 est devenue plus intéressante. Cependant la configuration courante n'est plus c_0 lorsque l'acte est reçu par v_2 et celui-ci est donc ignoré. Afin de soumettre cette proposition à nouveau, v_1 doit confirmer qu'il souhaite bien proposer que c_1 remplace c_2 en envoyant l'acte $Offer(c_1, c_2)$.

L'utilisation de ce mécanisme représente un gain d'autonomie pour les véhicules par rapport à une intersection à feux, puisque dans ce mécanisme chaque véhicule prend part au choix de la politique de régulation qui sera appliquée et peut faire valoir des arguments le concernant. De plus les véhicules gagnent également en autonomie par rapport à la majorité des mécanismes présentés en 1.2.1 et 1.2.2 où les véhicules n'ont aucune capacité décisionnelle et ne peuvent pas mettre en oeuvre de stratégie pour modifier leur admission au sein de l'intersection.

2.3.1 Rôle de l'agent intersection

Afin de réaliser une allocation du droit de passage qui maximise le bien-être social et encourage les comportements coopératifs, l'agent intersection prend part au processus de négociation.

Dans une négociation, chaque véhicule défend avant tout ses intérêts individuels, mais défend également d'autres intérêts qui peuvent mener vers une issue qui lui est favorable. Un véhicule peut également représenter l'intérêt de véhicules en dehors de V^{neg} (par exemple, des véhicules qui le suivent) ou des intérêts existant à plus large échelle (par exemple, évacuer certaines voies) si cela peut l'avantager. Cependant, il peut arriver que ces arguments ne concernent pas directement des véhicules de V^{neg}, et que ces derniers puissent les ignorer malgré leur contribution positive au bien-être social.

Pour éviter cet effet, l'agent intersection est en mesure de représenter ces intérêts externes. Comme les agents véhicules, l'agent intersection dispose de ses propres états mentaux et est capable de produire des arguments. Cependant, il ne peut pas accepter ou refuser des propositions. Le poids que l'agent intersection donne à chacun de ses arguments dépend de l'importance des intérêts externes représentés par celui-ci.

Un poids w_i est donné à chaque véhicule v_i par les agents intersections pour encourager les véhicules à avoir des comportements coopératifs. Selon le niveau de coopération de v_i dans son comportement de négociation, l'intersection augmente ou diminue w_i pour le reste du trajet de v_i. Le comportement d'un véhicule est complètement coopératif si ses acceptations et ses refus n'entrent en conflit avec aucun des arguments émis durant la négociation. Cependant certains arguments étant contradictoires, un comportement sera rarement complètement coopératif. Un véhicule qui refuse une proposition ayant de nombreux arguments forts en sa faveur (ou qui accepte une proposition ayant de nombreux arguments forts en sa défaveur) reçoit une importante pénalité sur le poids. À l'inverse, un véhicule acceptant une proposition ayant de nombreux arguments forts en sa faveur (ou refusant une proposition ayant de nombreux arguments forts en sa défaveur) reçoit une récompense sur le poids. Pour un véhicule, ces récompenses et pénalités sont significatives à moyen et à long termes car ils affectent durablement sa capacité à influencer les configurations de chaque intersection à venir.

L'intersection fixe les poids des récompenses et des pénalités en fonction du poids des véhicules. Un véhicule ayant déjà un poids élevé obtient une faible récompense en cas de comportement coopératif, mais une pénalité importante en cas de comportement

non-coopératif. Au contraire, un véhicule ayant un poids peu élevé obtiendrait une faible pénalité en cas de comportement non-coopératif mais une récompense importante en cas de comportement coopératif.

Soit $V^{min} \in V^{neg}$ l'ensemble des véhicules ayant émis des acceptations ou des refus s'opposant à une majorité d'arguments. Afin d'avoir plus d'influence sur le comportement des véhicules, l'intersection utilise des pénalités plus importantes quand le poids moyen des véhicules de V^{min} est supérieur au poids moyen des véhicules de V^{neg}, et utilise des récompenses plus importantes dans le cas contraire, comme décrit dans l'Algorithme 1. La mise-à-jour des poids des véhicules est réalisée par l'appel à la procédure UPDATE(V^{neg}, V^{neg}_{new}), où V^{neg}_{new} est l'ensemble des véhicules de V^{neg} n'ayant pas pris part à la négociation lors des itérations précédentes de celle-ci.

L'algorithme fonctionne de la manière suivante. Pour chaque véhicule ayant pris part à la négociation, on calcule son niveau de coopération $strengthCooperation$ comme la somme des poids des arguments portant sur les configurations au sujet desquelles ce véhicule à émis un $Accept$ ou un $Refuse$ durant la négociation, pondérée positivement si le véhicule va dans les sens des arguments majoritaires en poids et négativement sinon. $pastNegotiation(v_i)$ représente l'ensemble des coups réalisés par les véhicules durant les étapes passées de la négociation auxquelles v_i a pris part, et $author(move)$ représente le véhicule ayant réalisé le coup $move$. On compare ensuite le poids de ce véhicule au poids moyen des véhicules ayant pris part à la négociation, et on calcule un facteur $factor$ dépendant de l'écart entre le poids de ce véhicule et le poids moyen des véhicules. La valeur de $factor$ est basée sur la valeur de $strengthCooperation$ atténuée lorsque l'écart entre le poids du véhicule et le poids moyen est important, afin d'éviter une trop grande variabilité du poids des véhicules. Le poids du véhicule est ensuite mis à jour en étant multiplié par k^{factor}, la valeur de k étant supérieure dans le cas d'un véhicule ayant un poids inférieur à la moyenne que dans le cas contraire ($k_{lower} > k_{higher}$).

2.3.2 Scénario illustratif

Nous continuons le scénario décrit dans la section 2.2 (Figure 2.1). Chaque véhicule a construit les contraintes structurelles pour modéliser le problème et dispose d'un solveur pour construire des solutions. Trois solutions Pareto-optimales sont possibles : $c_1 = \{5, 7, 12\}$, $c_2 = \{5, 11, 10\}$, $c_3 = \{8, 9, 7\}$ (cf. Figure 2.4). Par exemple, la date d'admission de v_1 dans la configuration c_1 est $t_1^{c_1} = 5$. Sur un scénario très simple comme celui-ci, on peut raisonnablement supposer que le solveur de chaque véhicule est capable de produire ces 3 solutions dès sa première recherche, ainsi que d'autres solutions sous-optimales. En revanche lorsque le nombre de véhicules à l'approche

Algorithme 1 Politique d'attribution des poids

procedure UPDATE(V^{neg}, V^{neg}_{new})
  for each $v_i \in V^{neg} \setminus V^{neg}_{new}$ **do**
    $weight_i^{old} \leftarrow weight(v_i)$
    $strengthCooperation \leftarrow 0$
    for each $move \in pastNegotiation(v_i)$ **do**
      if $author(move) = v_i$ **then**
        if $move$ is $Accept$ **then**
          $c_j \leftarrow move.configuration$
          $sum =$SUMARGS$(c_j, pastNegotiation(v_i))$
          $strengthCooperation \leftarrow strengthCooperation + sum$
        end if
        if $move$ is $Refuse$ **then**
          $c_j \leftarrow move.configuration$
          $sum =$SUMARGS$(c_j, pastNegotiation(v_i))$
          $strengthCooperation \leftarrow strengthCooperation - sum$
        end if
      end if
    end for
    $average \leftarrow$AVERAGEWEIGHT$(pastNegotiation(v_i))$
    if $weight_i^{old} > average$ **then**
      $factor \leftarrow strengthCooperation * (\frac{average}{weight_i^{old}})^{coeff}$
      $weight_i \leftarrow weight_i^{old} * (k_{higher})^{factor}$
    else
      $factor \leftarrow strengthCooperation * (\frac{weight_i^{old}}{average})^{coeff}$
      $weight_i \leftarrow weight_i^{old} * (k_{lower})^{factor}$
    end if
  end for
end procedure

function AVERAGEWEIGHT($pastNegociation$)
  $vehicles \leftarrow \emptyset$
  for each $move \in pastNegotiation$ **do**
    $vehicles \leftarrow vehicles \cup author(move)$
  end for
  $sum \leftarrow 0$
  for each $v \in vehicles$ **do**
    $sum \leftarrow sum + weight(v)$
  end for
  return $\frac{sum}{|vehicles|}$
end function

de l'intersection est élevé, l'espace de recherche devient large et tous les véhicules ne trouvent pas nécessairement l'ensemble des solutions Pareto-optimales. Afin d'illustrer ce phénomène, nous supposons ici que tous les véhicules ne trouvent pas les 3 solutions Pareto-optimales dès leur première recherche. Nous supposons

function SUMARGS($c_j, pastNegociation$)
  $sum \leftarrow 0$
  for each $move \in pastNegotiation$ **do**
    if (($move$ is $Argue$) and ($move.configuration == c_j$)) **then**
      if ($move.argument$ is positive) **then**
        $sum \leftarrow sum + weight(move.argument)$
      else
        $sum \leftarrow sum - weight(move.argument)$
      end if
    end if
  end for
  return sum
end function

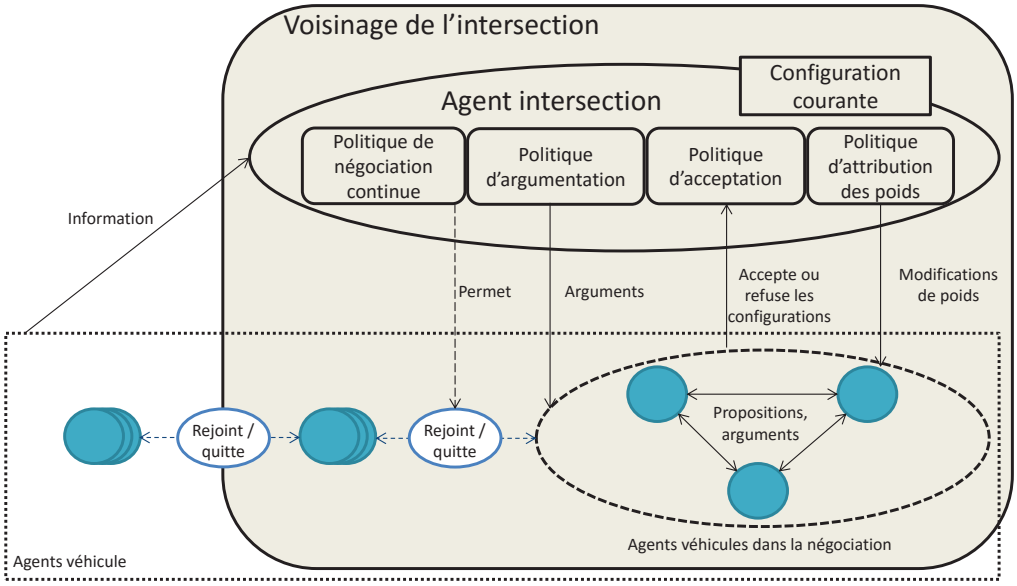


Figure 2.3: Architecture de l'agent intersection

que les résultats de cette première recherche de chaque solveur fournissent les configurations suivantes : (c_2, c_3) pour v_1, c_3 pour v_2, et (c_1, c_2) pour v_3.

Le contexte initial est le suivant : l'intersection a appliqué la politique FCFS pour construire une configuration par défaut, la configuration actuelle c_{cur} est donc $c_2 = \{5, 11, 10\}$. v_3 a un comportement coopératif depuis le début de son trajet et a donc maintenant un poids plus important que les deux autres véhicules : $w_1 = 10, w_2 = 10, w_3 = 25$. Nous supposons qu'un important groupe de véhicules gr_1 approche sur la voie de v_3, et le poids de chacun de ces véhicules est 10. Leur poids total est donc $w_{gr1} = 40$ (cf. Figure 2.5). Le seuil d'acceptation est $th_{accept} = 0.5$. Les états mentaux des agents sont fournis dans la Table 2.2.

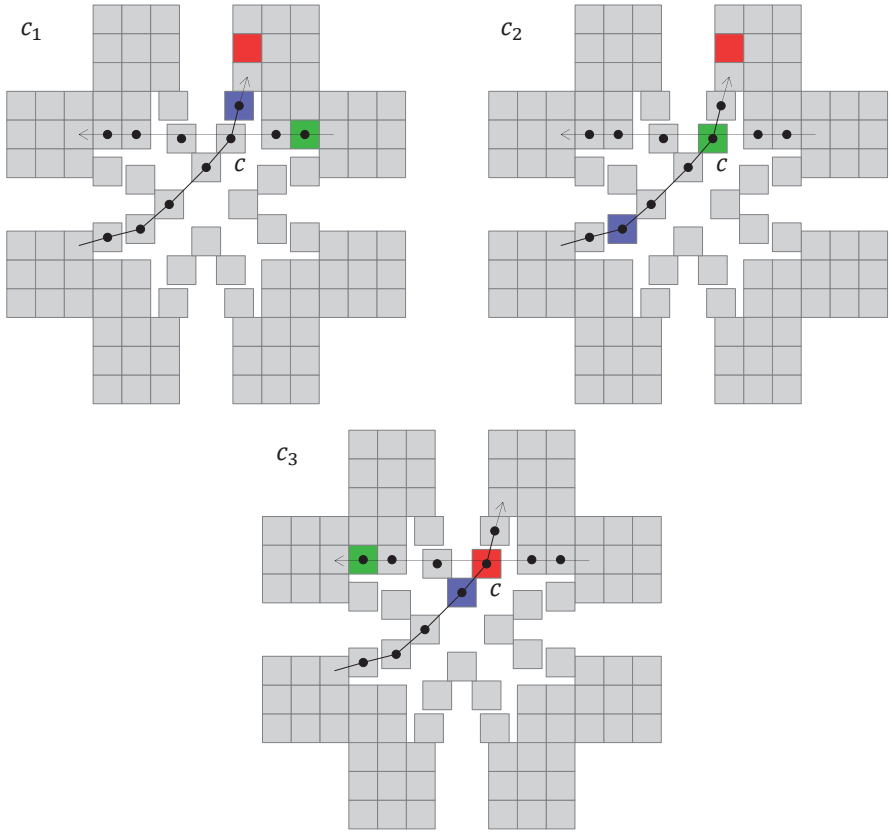


Figure 2.4: État de l'intersection à $t = 12$ en fonction de la configuration choisie

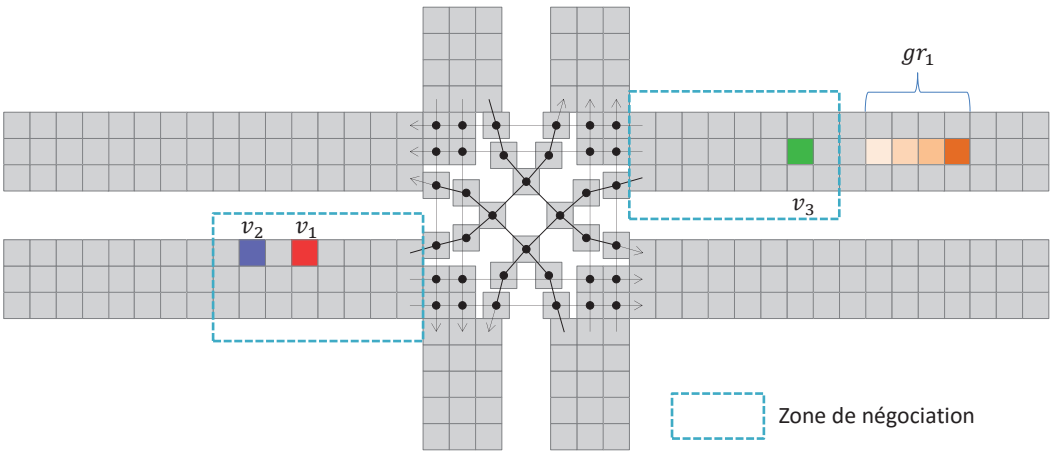


Figure 2.5: Un groupe de véhicule gr_1 approche de la zone de négociation.

v_i	$\mathcal{K}_i$	$\mathcal{G}_i$
v_1	$\mathcal{K}_1 = \{k_1^1 = (t_1^{new} < t_1^{cur} \rightarrow improve(v_1))(\rho_1^1 = 1)\}$	$\mathcal{G}_1 = \{g_1^1 = improve(v_1)(\lambda_1^1 = 0.4), g_1^3 = weight(v_1)(\lambda_1^3 = 0.6)\}$
v_2	$\mathcal{K}_2 = \{k_2^1 = (t_2^{new} < t_2^{cur} \rightarrow improve(v_2))(\rho_2^1 = 1), k_2^2 = (t_2^{new} - t_1^{new} \leq 3 \rightarrow group(v_2))(\rho_2^2 = 1)\}$	$\mathcal{G}_2 = \{g_2^1 = improve(v_2)(\lambda_2^1 = 0.3), g_2^2 = group(v_2)(\lambda_2^2 = 0.4), g_2^3 = weight(v_2)(\lambda_2^3 = 0.4)\}$
v_3	$\mathcal{K}_3 = \{k_3^1 = (t_3^{new} < t_3^{cur} \rightarrow improve(v_3))(\rho_3^1 = 1)\}$	$\mathcal{G}_3 = \{g_3^1 = improve(v_3)(\lambda_3^1 = 0.8), g_3^3 = weight(v_3)(\lambda_3^3 = 0.2)\}$

Table 2.2: États mentaux initiaux des agents

Étape	t_{cur}	v_1	v_2	v_3	it	c_{cur}	Coup(s)
	0	$c_2 \succ c_3$	c_3	$c_2 \succ c_1$	c_2	c_2	
0		-	$c_3 \succ c_2$	-	-	c_2	$v_2 : m_1$
1		-	-	$c_3 \succ c_2 \succ c_1$	$c_2 \sim c_3$	c_2	$v_1 : m_2$; $v_2 : m_3$; $v_3 : m_4$
2		-	-	-	$c_3 \succ c_2$	c_2	$it : m_5$
3		$c_3 \succ c_2$	-	-	-	c_2	$v_1, v_2, v_3 : m_6$
4	1	$c_1 \succ c_3 \succ c_2$	-	-	-	c_3	$v_1 : m_7$
5		-	$c_1 \succ c_3 \succ c_2$	-	$c_3 \succ c_2 \sim c_1$	c_3	$v_1 : m_8$; $v_2 : m_9, m_{10}$; $v_3 : m_{11}$; $it : m_{12}$
6		-	-	-	$c_1 \succ c_3 \succ c_2$	c_3	$it : m_{13}$
7		-	-	-	-	c_3	$v_1, v_2 : m_{14}$; $v_3 : m_{15}$

Table 2.3: Processus de négociation. Initialement, v_2 ne connaît que la configuration c_3. À l'étape 0, il prend connaissance de la configuration courante c_2. Il préfère alors c_3 à c_2, et applique donc le coup m_1.

Dans cet exemple, les agents ont trois types de buts.

(1) Avec le but $improve(v_i)$, l'agent cherche à améliorer la date d'admission de v_i, afin que ce véhicule traverse la prochaine intersection aussi tôt que possible.

(2) Avec $group(v_i)$, l'agent cherche à faire en sorte que v_i forme un groupe physique, appelé “peloton", avec d'autres véhicules sur la même voie, comme par exemple dans [48]. Les véhicules formant un peloton ont souvent des intérêts communs et peuvent avoir naturellement un comportement commun de négociation aux intersections. Un tel comportement leur donne un avantage de poids dans les négociations, et

peut apporter un avantage conséquent à long terme. Ce but représente le désir des véhicules de former des pelotons et de tirer parti des avantages potentiellement produits par ce phénomène.

(3) Avec $weight(v_i)$ l'agent cherche à maintenir le poids de v_i suffisamment haut pour que celui-ci soit influent lors des négociations aux prochaines intersections sur son trajet.

Pour des raisons de simplicité, on considère une évaluation booléenne de la réalisation de ces buts : ils peuvent être atteints ou non-atteints.

La négociation est décrite dans la Table 2.11. Cette table donne les préférences des agents v_1, v_2, v_3, et de l'agent intersection it. $c_x \succ c_y$ signifie que c_x est préféré à c_y. $c_x \sim c_y$ signifie que l'agent est indifférent entre c_x et c_y. ' - ' signifie que les préférences des agents n'ont pas changé depuis l'étape précédente. Durant chaque étape de la négociation, les agents produisent les coups décrits dans la Table 2.12.

v_2 peut améliorer sa date d'admission avec c_3 et la proposer dans la négociation (étape 0). c_3 améliore les dates d'admission de v_2 et v_3 et détériore la date de v_1. v_2 et v_3 construisent des arguments positifs sur c_3, et v_1 construit un argument négatif (étape 1). Les arguments positifs sont plus forts que l'argument négatif, donc l'intersection récompensera les véhicules qui voteront pour c_3, avec une récompense de poids égale à la force relative des arguments (étape 2). la récompense est suffisamment élevée pour changer les préférences de v_1, et v_1 accepte c_3. v_2 et v_3 sont favorables à c_3 et l'acceptent. Tous les véhicules ont accepté c_3, donc cette proposition remplace c_2 en tant que configuration courante de l'intersection (étape 3).

La négociation continue, et un pas de temps a eu lieu depuis le début de la négociation, durant lequel le solveur de v_1 a trouvé la configuration c_1. Puisque c_1 est maintenant la solution préférée de v_1, v_1 propose c_1 (étape 4). Les agents véhicules donnent leurs arguments pour c_1 (v_1 et v_2) ou contre c_1 (v_3). L'agent intersection estime que les véhicules de gr_1 peuvent tirer avantage de c_1, donc il donne un argument positif pour c_1 se basant sur cette information, avec un poids égal à w_{gr} (étape 5). Les arguments négatifs sont plus forts que les arguments positifs, donc l'intersection menace les véhicules qui ne voteraient pas pour c_1 de leur infliger une pénalité de poids égale au poids relatif des arguments (étape 6). La pénalité n'est pas assez importante pour changer les préférences de v_3, et v_3 refuse c_1. v_1 et v_2 étaient favorables à c_3, mais leurs poids cumulés ne sont pas suffisants pour changer la configuration et c_3 reste la configuration courante de l'intersection. v_3 est menacé

Nom du coup	Description du coup	Argument positif	Nom de l'argument
m_1	Offer(c_3, c_2)	/	/
m_2	Argue(c_3, Arg_1)	no	Arg_1
m_3	Argue(c_3, Arg_2)	yes	Arg_2
m_4	Argue(c_3, Arg_3)	yes	Arg_3
m_5	Argue($c_3, Reward_1$)	yes	$Reward_1$
m_6	Accept(c_3, c_2)	/	/
m_7	Offer(c_1, c_3)	/	/
m_8	Argue(c_1, Arg_4)	yes	Arg_4
m_9	Argue(c_1, Arg_5)	yes	Arg_5
m_{10}	Argue(c_1, Arg_6)	yes	Arg_6
m_{11}	Argue(c_1, Arg_7)	no	Arg_7
m_{12}	Argue(c_1, Arg_8)	yes	Arg_8
m_{13}	Argue($c_1, Threat_1$)	no	$Threat_1$
m_{14}	Accept(c_1, c_2)	/	/
m_{15}	Refuse(c_1, c_2)	/	/

Table 2.4: Coups utilisés durant la négociation

Nom	Argument
Arg_1	$< \{t_1^{c_3} \geq t_1^{cur}, t_1^{c_3} \geq t_1^{cur} \rightarrow \neg improve(v_1)\}, \{improve(v_1)\}, c_3, w_1 >$
Arg_2	$< \{t_2^{c_3} < t_2^{cur}, t_2^{c_3} < t_2^{cur} \rightarrow improve(v_2)\}, \{improve(v_2)\}, c_3, w_2 >$
Arg_3	$< \{t_3^{c_3} < t_3^{cur}, t_3^{c_3} < t_3^{cur} \rightarrow improve(v_3)\}, \{improve(v_3)\}, c_3, w_3 >$
$Reward_1$	$< \{weight(any)\}, \{weight(any)\}, c_3, w_2 + w_3 - w_1 >$
Arg_4	$< \{t_1^{c_1} < t_1^{cur}, t_1^{c_1} < t_1^{cur} \rightarrow improve(v_1)\}, \{improve(v_1)\}, c_1, w_1 >$
Arg_5	$< \{t_2^{c_1} < t_2^{cur}, t_2^{c_1} < t_2^{cur} \rightarrow improve(v_2)\}, \{improve(v_2)\}, c_1, w_2 >$
Arg_6	$< \{t_2^{c_1} - t_1^{c_1} \leq 3, t_2^{c_1} - t_1^{c_1} \leq 3 \rightarrow group(v_2)\}, \{group(v_2)\}, c_1, w_2 >$
Arg_7	$< \{t_3^{c_1} \geq t_3^{cur}, t_3^{c_1} \geq t_3^{cur} \rightarrow \neg improve(v_3)\}, \{improve(v_3)\}, c_1, w_3 >$
Arg_8	$< \{t_3 > 10, t_3 > 10 \rightarrow group(gr_1)\}, \{group(gr_1)\}, c_1, w_{gr1} >$
$Threat_1$	$< \{\neg weight(any)\}, \{weight(any)\}, c_1, w_1 + w_2 + w_{gr1} - w_3 >$

Table 2.5: Arguments utilisés durant la négociation

et s'il ne change pas son refus en une acceptation avant de traverser l'intersection, son poids sera réduit (étape 7)

2.3.3 Contraintes non-structurelles

Comme expliqué précédemment, un véhicule peut choisir de guider la recherche de solutions par le solveur en utilisant des contraintes additionnelles, différentes des contraintes structurelles générées par les 3 règles décrites précédemment (règles de distance, d'antériorité et de conflit).

Soit le scénario suivant. Un bus b_1 et plusieurs véhicules approchent d'une intersection, et les itérations précédentes de la négociation ont donné des poids forts à des arguments de la forme :
$Arg =< Supp, t_{b_1} \leq t^1_{quick}, d, w_x >$
où t^1_{quick} est une date de passage fixe.

Les agents mettent donc à jour leurs $\mathcal{GO}$ respectifs pour prendre en compte le fait que le passage du bus b_1 avant la date de passage t_{quick} fait partie des buts supposés des autres véhicules. Lors de l'itération suivante, si le poids cumulé des véhicules pour lesquels ce but est mentionné est élevé, il peut être intéressant de limiter l'espace de recherche des configurations afin de ne considérer que les solutions satisfaisant ce but. Ainsi lorsqu'un véhicule constate un but de poids important chez les autres véhicules, il peut utiliser ce but comme une contrainte à ajouter au solveur lors de sa recherche de solutions.

Considérons maintenant un scénario similaire, avec les différences suivantes : un autre bus b_2 est présent sur une autre voie, et sa trajectoire est conflictuelle avec celle de b_1. Les arguments de la forme suivante possèdent également un poids important :
$Arg =< Supp, t_{b_2} \leq t^2_{quick}, d, w_x >$
où t^2_{quick} est une date de passage fixe.

De la même manière que pour b_1, les agents mettent à jour leurs $\mathcal{GO}$. Il peut également leur être intéressant de formaliser l'importance de cet argument sous la forme d'une contrainte afin de guider la recherche des solutions. Cependant, il peut aussi arriver que les deux contraintes $t_{b_1} \leq t^1_{quick}$ et $t_{b_2} \leq t^2_{quick}$ ne puissent pas être satisfaites en même temps. Compte tenu du caractère facultatif de ces contraintes non-structurelles, on admet que celles-ci puissent être violées si aucune solution n'est trouvée en les respectant.

2.3.4 Propriétés du mécanisme

Notre mécanisme possède les propriétés suivantes :

Proposition 1 : Soit V^{neg} l'ensemble des agents approchant de l'intersection. Le nombre de propositions de configurations possibles est fini.

Preuve : Pour prouver cette proposition, considérons la pire configuration possible c_{worst} et démontrons par construction que l'ensemble des dates d'admission de c_{worst} pour chaque véhicule est borné. Une par une, nous ajoutons les dates d'admission des véhicules de V^{neg} à la configuration c_{worst}. Premièrement, nous remarquons que chaque véhicule $v_i \in V^{neg}$ peut produire une date est_i correspondant à son objectif d'arrivée à sa destination finale. Il existe donc pour chaque véhicule une date au-delà de laquelle ce véhicule n'a pas intérêt à ne pas s'engager dans l'intersection s'il le peut. Soit est_{max} la valeur maximale de est_i $\forall v_i \in V^{neg}$. Dans le pire des cas, le premier véhicule passe à $t_{first} = est_{max}$. À cette étape, c_{worst} contient exactement une date d'admission. Nous continuons à construire c_{worst}. Après t_{first}, chaque véhicule a intérêt à franchir l'intersection aussi vite que possible. Soit v_k le véhicule ayant la pire date d'admission dans c_{worst} et v'_k un véhicule qui n'a pas encore de date d'admission dans c_{worst}. Soit d_{max} la valeur maximale de d_i $\forall v_i \in V^{neg}$, et $traj_{max}$ la plus longue trajectoire au sein de la zone de conflit. Dans le pire des cas, v'_k doit attendre que v_k traverse l'intersection et laisser s'écouler le délai de sécurité t_{safe}, et/ou v'_k doit atteindre la zone de conflit s'il en est éloigné. Nous avons alors : $t'_k \leq t_k + \frac{d_{max}+traj_{max}+t_{safe}}{sp}$, où sp est la vitesse des véhicules. La date d'admission du premier véhicule est au pire est_{max}, nous avons alors :
$\forall v_i \in V^{neg}, t_i \leq est_{max} + |V^{neg}| * \frac{d_{max}+traj_{max}+t_{safe}}{sp}$
$\mathcal{D}$ est l'ensemble des configurations possibles et chaque configuration de $\mathcal{D}$ est un ensemble de dates d'admission respectant cette propriété. Donc :
$|\mathcal{D}| \leq (est_{max} + |V^{neg}| * \frac{d_{max}+traj_{max}+t_{safe}}{sp})^{|V^{neg}|}$

Proposition 2 : Le complexité communicationnelle du protocole est $\mathcal{O}(mn^2p^2)$.

Preuve : Considérant un ensemble de n agents, p configurations possibles et m arguments possibles pour chaque agent, nous pouvons borner le nombre de coups dans la négociation. Chaque agent peut fournir m arguments : mn messages $Argue$. Chaque agent peut refuser k propositions, avec $k < p$: pn messages $Refuse$. Chaque agent peut faire chaque coup $Offer$ une fois, et chaque proposition possède deux configurations comme paramètres : np^2 messages $Offer$. Le nombre de coups $Accept$ n'est pas limité. En effet, après qu'une proposition de configuration c_x ait été acceptée, une autre configuration c_y peut être acceptée et la remplacer. Puis de nouveaux arguments pour c_x peuvent être proposés et c_x peut être acceptée

à nouveau. Puis de nouveaux arguments pour c_y peuvent être proposés, c_y peut alors être acceptée de nouveau, etc. Une fois qu'une offre est acceptée, les messages $Accept$ correspondants sont retirés de la négociation, et la même offre peut être acceptée plus tard dans la négociation. Cependant les agents n'ont aucun intérêt à accepter à nouveau une proposition qui a été choisie puis remplacée, à moins que leur évaluation de l'utilité des propositions n'ait changé. L'utilité de la configuration pour un agent peut changer à chaque fois qu'un message $Argue$ est envoyé. Tant que les fonctions d'utilité des agents ne changent pas, chaque agent ne peut accepter chaque proposition qu'une fois. Puisqu'il y a au plus np^2 propositions et que l'utilité des agents peut changer à chaque message $Argue$ (au plus mn messages échangés), alors au plus mn^2p^2 messages $Accept$ peuvent être envoyés. Le nombre maximum de messages est donc $mn + pn + np^2 + mn^2p^2$. La complexité du protocole est : $\mathcal{O}(mn^2p^2)$.

2.3.5 Mécanisme de continuité de la négociation

Les véhicules ont la capacité de communiquer et de choisir collectivement la configuration de l'intersection. Cependant le flux de véhicules est continu, donc le mécanisme doit gérer cet aspect dynamique en définissant quels agents prennent part à chaque étape de la négociation, pour quels véhicules la configuration doit fournir une date d'admission, ainsi que les conditions sous lesquelles la configuration peut être révisée une fois choisie.

Afin de gérer les risques de conflits, l'intersection a une configuration courante c_{cur} à chaque instant. En fonction de la politique de continuité qui s'applique, le mécanisme de négociation peut autoriser les véhicules à changer cette configuration. Cependant, le mécanisme doit tenir compte des mesures de sécurité avant d'autoriser ce changement. Changer la configuration au dernier moment présente des risques importants à cause de la lenteur des réactions des conducteurs. Afin d'éviter ces situations critiques, nous définissons un seuil de sécurité th_{safe}. La date d'admission d'un véhicule ne peut donc pas être révisée (retirée ou accordée) à trop court terme.

Définition Soit t_i^{cur} la date d'admission du véhicule v_i dans la configuration courante et t_i^{next} sa date d'admission dans une configuration c. c est une proposition éligible si et seulement si c est valide et : $\forall v_i \in V^{neg}, (t_i^{cur} = t_i^{next}) \vee ((t_i^{cur} \geq t_{cur} + th_{safe}) \wedge (t_i^{next} \geq t_{cur} + th_{safe}))$

Dans ce qui suit, nous proposons plusieurs politiques pour gérer le problème de la continuité. Premièrement, nous distinguons deux zones sur les approches de l'intersection : la zone interne, où les véhicules approchent de la zone de conflit de

manière imminente, et la zone externe, où les véhicules devraient atteindre la zone de conflit à plus long terme (cf. Figure 2.6). La taille de chaque zone dépend de l'intersection. À chaque pas de temps t_i, l'ensemble V_i des véhicules à l'approche est divisé en deux sous-ensembles : V_i^{inn} les véhicules de la zone interne et V_i^{ext} les véhicules de la zone externe. $V_i = V_i^{inn} \cup V_i^{ext}, V_i^{inn} \cap V_i^{ext} = \emptyset$

Soit $\mathcal{T}$ la période consacrée à la négociation. Soit Δ^{ref} le seuil qui est le nombre maximum d'actes $Refuse$ que les agents peuvent réaliser et δ_i^{ref} le nombre de $Refuse$ qu'un agent v_i a envoyé durant $\mathcal{T}$. Si $\delta_i^{ref} = \Delta^{ref}$, v_i ne peut plus faire aucun coup $Offer$ ou $Refuse$. Soit Δ^{arg} le seuil qui représente le nombre maximum de messages $Argue$ qu'un agent peut envoyer et δ_i^{arg} le nombre de $Argue$ qu'un agent v_i a envoyé durant $\mathcal{T}$. Si $\delta_i^{arg} = \Delta^{ref}$, v_i ne peut faire aucun coup $Offer(c_x, c_y)$ durant $\mathcal{T}$. Un agent ne peut soumettre une offre qu'une seule fois durant la négociation. Une fois qu'un agent a fait le coup $Offer(c_x, c_y)$ durant $\mathcal{T}$, il ne peut plus le refaire durant toute la négociation. Nous avons ainsi l'ensemble de règles suivant :

- **NR1 :** $\forall v_i \in V^{neg}$, le coup $Offer(c_x, c_y)$ peut être réalisé à tout moment $t \in \mathcal{T}$ de la négociation par un agent v_i si ce coup n'a pas encore été fait par v_i durant $\mathcal{T}$ et si $\delta_i^{ref} < \Delta^{ref}$.

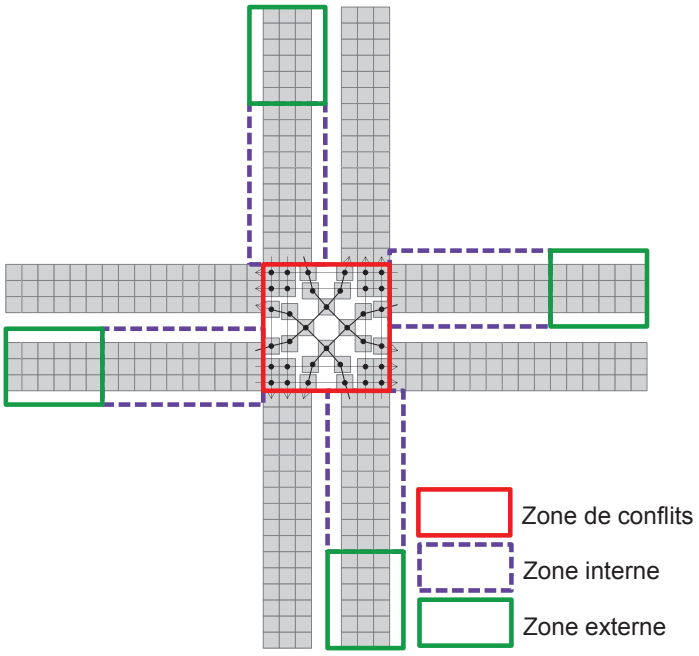


Figure 2.6: Deux zones sur les approches de l'intersection, la zone interne et la zone externe.

- **NR2 :** $\forall v_i \in V^{neg}$, le coup $Accept(c_x, c_y)$ peut être réalisé à tout moment $t \in \mathcal{T}$ de la négociation par l'agent v_i si le coup $Offer(c_x, c_y)$ a été fait à $t_0 \in \mathcal{T}, t_0 < t$.

- **NR3 :** $\forall v_i \in V^{neg}$, le coup $Refuse(c_x, c_y)$ peut être réalisé à tout moment $t \in \mathcal{T}$ de la négociation par l'agent v_i seulement si $\delta_i^{arg} < \Delta^{arg}$ et si le coup $Offer(c_x, c_y)$ a été fait à $t_0 \in \mathcal{T}, t_0 < t$.

- **NR4 :** $\forall v_i \in V^{neg}$, le coup $Argue(c_x, arg(c_x))$ peut être réalisé à tout moment $t \in \mathcal{T}$ de la négociation par l'agent v_i seulement si $\delta_i^{arg} < \Delta^{arg}$ et si le coup $Offer(c_x, c_y)$ a été fait à $t_0 \in \mathcal{T}, t_0 < t$, pour n'importe quel $c_y \in \mathcal{D}$.

Nous proposons ici deux types de politiques. La politique IP (Iterated Policy) fonctionne par ajouts successifs de groupes de véhicules qui négocient entre eux leurs dates de passages respectives sans remettre en cause ce qui a été précédemment décidé. La politique CP (Continuous Policy) fonctionne au contraire de manière totalement continue, et tous les véhicules proches de l'intersection sont susceptibles de renégocier l'intégralité de la configuration à tout moment, dans les limites permises par les contraintes de sécurité.

Iterated Policy (IP)

Lorsque cette politique est appliquée, les agents véhicules rejoignent la négociation par vagues, et prennent des décisions itérées qui ne peuvent plus être révisées une fois qu'une nouvelle vague a commencé sa négociation.

À un instant donné t_{i-1}, V^{inn} est vide. Lors du pas de temps suivant t_i, les véhicules se sont déplacés, V^{inn} et V^{ext} changent. L'ensemble des véhicules participant à la négociation V_i^{neg} devient égal à V_i^{inn}. Alors les véhicules de V_i^{neg} décident collectivement d'une configuration portant sur tous les véhicules de V_i^{neg}. Un processus de négociation a lieu, avec une durée limitée d_{neg} en s'appuyant sur l'ensemble de règles ci-dessus. $\mathcal{T} = [t_0^{neg}, t_0^{neg} + d_{neg}]$, où t_0^{neg} est la date du début de la négociation. Cette durée limitée amène les agents à soumettre rapidement des propositions raisonnables pour tous les véhicules. À la fin de l'étape de négociation, une configuration c_i est choisie, attribuant une date d'admission à chaque véhicule de V_i^{neg}

À chaque instant t_{i+k}, une nouvelle itération commence. Si $t_{i+k} < t_0^{neg} + d_{neg}$, alors V_{neg} reste inchangé. Sinon, $t_0^{neg} \leftarrow t_{i+k}$ et $V_{i+k}^{neg} = V_{i+k}^{inn} \setminus V_i^{neg}$. Les véhicules de V_{i+k}^{neg} commencent alors une nouvelle négociation, mais les véhicules ayant déjà pris part à une des itérations précédentes de la négociation ne prennent pas part

à celle-ci. Par ailleurs, les agents de V_{i+k}^{neg} ne sont pas autorisés à réviser c_i, ils négocient uniquement les dates d'admission des véhicules de V_{i+k}^{neg} puisque les autres véhicules de V_i^{inn} ont déjà des dates d'admission définies dans c_i ou dans une des configurations précédentes. Une nouvelle configuration c_{i+k} est choisie, similaire à c_i excepté qu'elle comprend des dates d'admission pour les véhicules de V_{i+k}^{neg}. $c_i \setminus c_i^{out} \subseteq c_{i+k}$ où c_i^{out} est l'ensemble des véhicules admis dans la zone de conflit : $\forall t_j \in c_i, t_j < t_i \Leftrightarrow t_j \in c_i^{out}$.

La politique continue à itérer et à produire de nouvelles dates d'admission pour les nouveaux véhicules de la zone interne sans réviser celles des véhicules déjà présents dans celle-ci.

Une politique étendue EIP (Extended Iterated Policy) a été, par ailleurs, définie à partir d'IP. Cette politique est similaire à IP, excepté que lorsqu'une itération se termine, une nouvelle itération ne démarre pas avant d'avoir un nombre minimum de véhicules prêts à négocier.

Si $V_i^{neg} = V_i^{inn} \setminus V_{i-1}^{neg}$ ne contient pas au moins min_v véhicules, il peut être plus intéressant d'attendre avant de démarrer une nouvelle itération de la négociation. Dans ce cas, une intersection donne une date d'admission temporaire aux véhicules de V_i^{neg} en utilisant la politique FCFS (First Come First Served) sur la configuration courante. Ce fonctionnement est décrit dans l'Algorithme 2.

Étendre une configuration avec FCFS consiste à accorder à chaque nouveau véhicule la première date d'admission disponible, sans changer les dates d'admission pour les véhicules déjà admis. Ces véhicules prennent part à la prochaine itération de la négociation lorsqu'elle commence et peuvent réviser la date d'admission temporaire qui leur a été attribuée avec FCFS. Dans ce cas, $V_{i+1}^{neg} = V_{i+1}^{inn} \setminus V_{i-1}^{neg}$.

Exemple illustratif de la politique IP

Nous continuons ici l'exemple commencé en 2.3.2 en appliquant la politique de continuité IP.

Afin de mieux illustrer le fonctionnement de la politique IP, nous modifions légèrement le scénario en ajoutant un véhicule v_8 avec un poids de $w_8 = 15$ sur la même voie que v_1 et v_2 (cf. Figure 2.7). Le début du processus de négociation donné en section 2.3.2, qui représente la première itération de la négociation puisque nous utilisons une durée de 2 pas de temps par itération, reste inchangé. Nous décidons

Algorithme 2 EIP Policy

$V_{old} \leftarrow \emptyset$
$c_{prec} \leftarrow \emptyset$
$V_{neg} \leftarrow V^{inn}$
while $(true)$ **do**
  $t_0^{neg} \leftarrow t_{cur}$
  while $(t_{cur} < t_0^{neg} + d_{neg})$ **do**
    $c_{new} \leftarrow negotiate(V_{neg}, c_{cur})$
    if $c_{prec} \subset c_{new}$ **then**
      $c_{cur} \leftarrow c_{new}$
    end if
  end while
  $c_{prec} \leftarrow c_{cur}$
  $V_{old} \leftarrow V_{old} \cup V_{neg}$
  $V_{neg} \leftarrow V^{inn} \setminus V_{old}$
  while $|V_{neg}| < min_v$ **do**
    $c_{cur} \leftarrow FCFS(c_{prec}, V_{neg})$
    $wait()$
    $V_{neg} \leftarrow V^{inn} \setminus V_{old}$
  end while
end while

d'utiliser la version EIP de cette politique, et un minimum de 3 véhicules pour démarrer une nouvelle itération.

À la fin de $t = 1$, la première itération de la négociation est terminée. c_3 est choisie et une pénalité de poids est attribuée à v_3. À $t = 2$, v_4 et v_8 entrent dans la zone de négociation. L'intersection perçoit v_4 en premier, donc les dates d'admission de v_4 et v_8 sont ajoutées à c_3 avec FCFS dans cet ordre. La configuration courante est maintenant : $c'_3 = \{8, 9, 7, 14, -, -, -, 15\}$. Seuls deux véhicules sont capables de négocier, donc la nouvelle itération de la négociation ne commence pas aussitôt (étape 8).

À $t = 3$, v_5 entre dans la zone de négociation. Sa date d'admission est ajoutée à c'_3 avec FCFS, la configuration courante est maintenant $c'_3 = \{8, 9, 7, 14, 20, -, -, 15\}$. Trois véhicules sont dans la zone de négociation, donc une nouvelle itération de la négociation commence. $c_{cur} = c'_3$ et toutes les configurations proposées par les véhicules doivent respecter c_3 ($t_1 = 8, t_2 = 9, t_3 = 7$). Trois configurations Pareto-optimales sont possibles : $c_{31} = c''_3 = \{8, 9, 7, 14, 20, -, -, 15\}$, $c_{32} = \{8, 9, 7, 14, 15, -, -, 16\}$, $c_{33} = \{8, 9, 7, 15, 16, -, -, 10\}$.
Supposons que les solveurs respectifs des véhicules identifient les solutions de configuration suivantes : (c_{32}) pour v_4, (c_{32}) pour v_5 et (c_{32}, c_{33}) pour v_8. c_{32} et c_{33} sont respectivement les solutions préférées de v_5 et v_8, et ils proposent donc ces configurations (étape 9).

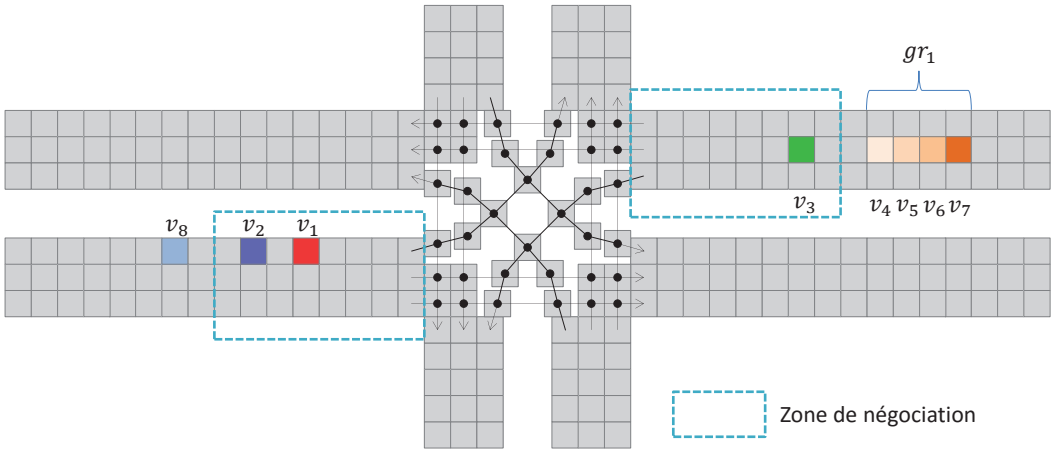


Figure 2.7: Un groupe de véhicules gr_1 et un véhicule v_8 approchent de la zone de négociation.

v_i	$\mathcal{K}_i$	$\mathcal{G}_i$
v_4	$\mathcal{K}_4 = \{k_4^1 = (t_4^{new} < t_4^{cur} \rightarrow improve(v_4))(\rho_4^1 = 1),$ $k_4^2 = (t_4^{new} - t_3^{new} \leq 3 \rightarrow group(v_4)(\rho_4^2 = 1)\}$	$\mathcal{G}_4 = \{g_4^1 = improve(v_4)(\lambda_4^1 = 0.3),$ $g_4^2 = group(v_4)(\lambda_4^2 = 0.5),$ $g_4^3 = weight(v_4)(\lambda_4^3 = 0.2)\}$
v_5	$\mathcal{K}_5 = \{k_5^1 = (t_5^{new} < t_5^{cur} \rightarrow improve(v_5))(\rho_5^1 = 1),$ $k_5^2 = (t_5^{new} - t_4^{new} \leq 3 \rightarrow group(v_5)(\rho_5^2 = 1)\}$	$\mathcal{G}_5 = \{g_5^1 = improve(v_5)(\lambda_5^1 = 0.3),$ $g_5^2 = group(v_5)(\lambda_5^2 = 0.1),$ $g_5^3 = weight(v_5)(\lambda_5^3 = 0.6)\}$
v_8	$\mathcal{K}_8 = \{k_8^1 = (t_8^{new} < t_8^{cur} \rightarrow improve(v_8))(\rho_8^1 = 1),$ $k_8^2 = (t_8^{new} - t_2^{new} \leq 3 \rightarrow group(v_8)(\rho_8^2 = 1)\}$	$\mathcal{G}_8 = \{g_5^1 = improve(v_5)(\lambda_8^1 = 0.4),$ $g_8^2 = group(v_8)(\lambda_8^2 = 0.3),$ $g_8^3 = weight(v_8)(\lambda_8^3 = 0.3)\}$

Table 2.6: IP : États mentaux initiaux des agents

v_4 construit un argument négatif sur c_{33}, v_5 construit un argument positif sur c_{32}, et v_8 construit un argument positif sur c_{33}. L'agent intersection estime que les véhicules restants de gr_1 peuvent tirer partie de c_{33}, il donne donc un nouvel argument en faveur de c_{33} sur la base de cette information, avec un poids de $w_{gr2} = w_6 + w_7$ (étape 10). Les poids des arguments positifs sur c_{33} sont plus forts que les poids des arguments négatifs, donc l'intersection récompense les véhicules qui voteraient pour c_{33}, avec un poids égal à la force relative de ces arguments (étape 11). La récompense est assez élevée pour changer les préférences de v_5, et v_5 accepte c_{33}. v_8 était favorable à c_{33} et l'accepte également (étape 12). Les poids cumulés de v_5 et v_8 sont suffisamment élevés pour changer la configuration, donc c_{33} remplace c_{31} en tant que configuration courante de l'intersection (étape 13).

Étape	t_{cur}	v_4	v_5	v_8	it	c_{cur}	Coup(s)
...	1	-	-	-	-	c_3	...
8	2	-	-	-	-	c'_3	-
9	3	$c_{32} \sim c_{31}$	$c_{32} \succ c_{31}$	$c_{33} \succ c_{31}$ $\succ c_{32}$	-	c_{31}	$v_5 : m_{16}$; $v_8 : m_{17}$
10		$c_{32} \sim c_{31}$ $\succ c_{33}$	$c_{32} \succ c_{33}$ $\succ c_{31}$	-	-	c_{31}	$v_4 : m_{18}$; $v_5 : m_{19}$; $v_8 : m_{20}$; $it : m_{21}$
11		-	-	-	$c_{33} \succ c_{32}$ $\succ c_{31}$	c_{31}	$it : m_{22}$
12		-	$c_{33} \succ c_{32}$ $\succ c_{31}$	-	-	c_{31}	$v_5, v_8 : m_{23}$
13	4	-	-	-	-	c_{33}	...

Table 2.7: IP : Processus de négociation

Nom du coup	Description du coup	Argument positif	Argument
m_{16}	Offer(c_{32}, c_{31})	/	/
m_{17}	Offer(c_{33}, c_{31})	/	/
m_{18}	Argue(c_{33}, Arg_9)	non	Arg_9
m_{19}	Argue(c_{32}, Arg_{10})	oui	Arg_{10}
m_{20}	Argue(c_{33}, Arg_{11})	oui	Arg_{11}
m_{21}	Argue(c_{33}, Arg_{12})	oui	Arg_{12}
m_{22}	Argue($c_{33}, Reward_2$)	oui	$Reward_2$
m_{23}	Accept(c_{33}, c_{31})	/	/

Table 2.8: IP : Coups utilisés durant la négociation

Nom	Argument
Arg_9	$< \{t_4^{c_{33}} \geq t_4^{cur}, t_4^{c_{33}} \geq t_4^{cur} \rightarrow \neg improve(v_4)\}, \{improve(v_4)\}, c_{33}, w_4 >$
Arg_{10}	$< \{t_5^{c_{32}} - t_4^{c_{32}} \leq 3, t_5^{c_{32}} - t_4^{c_{32}} \leq 3 \rightarrow group(v_5)\}, \{group(v_5)\}, c_{32}, w_5 >$
Arg_{11}	$< \{t_8^{c_{33}} < t_8^{cur}, t_8^{c_{33}} < t_8^{cur} \rightarrow improve(v_8)\}, \{improve(v_8)\}, c_{33}, w_8 >$
Arg_{12}	$< \{(t_8 < t_5) \wedge (t_8 < t_6), (t_8 < t_5) \wedge (t_8 < t_6) \rightarrow group(gr_2)\},$ $\{group(gr_2)\}, c_{33}, w_{gr2} >$
$Reward_2$	$< \{weight(any)\}, \{weight(any)\}, c_{33}, w_8 + w_{gr2} - w_4 >$

Table 2.9: IP : Arguments utilisés pendant la négociation

Continuous Policy (CP)

Quand cette politique est appliquée, les véhicules rejoignent dynamiquement la négociation en cours en entrant dans la zone interne, $V^{neg} = V^{inn}$ à tout moment. Quand un véhicule v_{new} rejoint V^{inn}, toutes les informations utiles sur l'état courant de la négociation (configuration et arguments) sont communiquées à v_{new} afin qu'il puisse rejoindre la négociation. La configuration courante de l'intersection peut être totalement révisée par une décision collective, excepté pour les véhicules concernés par le seuil de sécurité.

Lorsque de nouveaux véhicules rejoignent V^{inn}, la configuration courante de l'intersection et les configurations actuellement proposées dans la négociation ne disposent pas de dates pour ces véhicules car ces configurations ont été émises avant que ces véhicules ne rejoignent V^{inn}. En revanche l'intersection fournit un ordre d'admission sur ces véhicules. Avec cet ordre, il est possible pour n'importe quel véhicule dans la négociation d'étendre n'importe quelle proposition de configuration. Étendre une configuration consiste à y ajouter une date d'admission pour chaque nouveau véhicule avec la stratégie FCFS, dans l'ordre fourni par l'intersection. Les agents considèrent donc que toute proposition de configuration dans la négociation, ou toute configuration courante, qui ne fournit pas de date d'admission pour chaque véhicule de V^{inn} est étendue avec FCFS. Cela garantit que l'intersection ait toujours une date d'admission pour chaque véhicule de V^{inn}. Le fonctionnement de CP est décrit dans l'Algorithme 3.

Une amélioration de cette politique consisterait à étendre CP avec une nouvelle politique CPA (Continuous Policy with Anticipation). Dans CP, lorsqu'un véhicule construit une configuration, celle-ci n'intègre des dates d'admission que pour les véhicules de V^{inn}. Dans CPA, chaque véhicule $v_1 \in V^{neg}$ peut prendre en compte n'importe quel véhicule de $v_2 \in V^{ext}$ en construisant ses configurations, afin de pouvoir produire des arguments différents et d'en tirer avantage. Ensuite, lorsque ce véhicule v_2 rejoint V^{inn}, certaines configurations (dont la configuration courante de l'intersection) sont susceptibles de déjà inclure une date d'admission pour ce véhicule. En fonction du résultat des négociations précédentes, ces configurations peuvent être meilleures que celles produites en étendant CP avec FCFS.

Exemple illustratif de la politique CP

Nous continuons ici l'exemple commencé en 2.3.2 en appliquant la politique de continuité CP.

Algorithme 3 Politique CP

$c_{cur} \leftarrow \emptyset$
$V_{prec} \leftarrow \emptyset$
$V_{neg} \leftarrow \emptyset$
while $(true)$ **do**
  $V_{prec} \leftarrow V_{neg}$
  $V_{neg} \leftarrow V^{inn}$
  $V_{new} \leftarrow V_{neg} \setminus V_{prec}$
  $giveInformation(V_{new})$
  $c_{cur} \leftarrow FCFS(c_{cur}, V_{new})$
  $c_{new} \leftarrow negotiate(V_{neg}, c_{cur})$
end while

Un nouveau véhicule $v_4 \in gr$ entre dans la zone interne au pas de temps suivant. Puisque v_4 rejoint V^{inn}, il rejoint immédiatement la négociation. Son poids individuel est $w_4 = 10$. v_4 reçoit toutes les informations sur l'état de la négociation, et sa date d'admission est ajoutée à chaque configuration, avec FCFS. Nous avons maintenant : $c_1 = \{5, 7, 12, 13\}$, $c_2 = \{5, 11, 10, 16\}$, $c_3 = \{8, 9, 7, 14\}$. c_1 est la solution préférée de v_4 donc v_4 fournit un nouvel argument en faveur de c_1 (étape 8).

Puisque le poids total des véhicules qui préfèrent c_1 à c_3 est plus grand que le poids total des véhicules qui préfèrent c_3 à c_1, v_3 s'expose à une pénalité de poids sans aucune récompense s'il ne change pas son refus en une acceptation, alors il accepte c_1. De plus, c_1 est la configuration préférée de v_4. v_3 et v_4 acceptent c_1 (étape 9). c_1 remplace c_3 en tant que configuration courante de l'intersection (étape 10).

Au pas de temps suivant, $v_5 \in gr$ entre dans la zone interne. Durant le pas de temps précédent, v_4 a produit deux nouvelles configurations en utilisant son solveur : $c_4 = \{11, 12, 7, 10\}$, $c_5 = \{5, 12, 10, 11\}$. En étendant les cinq configurations pour ajouter v_5, nous avons maintenant : $c_1 = \{5, 7, 12, 13, 14\}$, $c_2 = \{5, 11, 10, 16, 17\}$, $c_3 = \{8, 9, 7, 14, 15\}$, $c_4 = \{11, 12, 7, 10, 17\}$ et $c_5 = \{5, 12, 10, 11, 17\}$. Cependant la date d'admission de v_1 dans c_{cur} est 5, $t_{cur} = 3$ et nous avons défini un seuil de sécurité $th_{safe} = 3$. Avec ce seuil, nous ne pouvons pas accorder ou retirer une date d'admission à un véhicule au dernier moment, donc toutes les configurations doivent respecter $t_1 = 5$ pour être éligibles. Les seules configurations éligibles sont c_1, c_2 et c_5. Puisque c_5 et la configuration la moins préférée de v_5, v_5 ne la propose pas dans la négociation. c_1 est sa configuration préférée, mais c_1 est déjà la configuration courante, donc v_5 n'a pas besoin de construire d'argument en faveur de c_1 (étape 11).

La négociation continue ainsi étape par étape.

v_i	$\mathcal{K}_i$	$\mathcal{G}_i$
v_4	$\mathcal{K}_4 = \{k_4^1 = (t_4^{new} < t_4^{cur} \rightarrow improve(v_4))(\rho_4^1 = 1),$ $k_4^2 = (t_4^{new} - t_3^{new} \leq 3 \rightarrow group(v_4))(\rho_4^2 = 1)\}$	$\mathcal{G}_4 = \{g_4^1 = improve(v_4)(\lambda_4^1 = 0.3),$ $g_4^2 = group(v_4)(\lambda_4^2 = 0.5),$ $g_4^3 = weight(v_4)(\lambda_4^3 = 0.2)\}$
v_5	$\mathcal{K}_5 = \{k_5^1 = (t_5^{new} < t_5^{cur} \rightarrow improve(v_5))(\rho_5^1 = 1),$ $k_5^2 = (t_5^{new} - t_4^{new} \leq 3 \rightarrow group(v_5))(\rho_5^2 = 1)\}$	$\mathcal{G}_5 = \{g_5^1 = improve(v_5)(\lambda_5^1 = 0.3),$ $g_5^2 = group(v_5)(\lambda_5^2 = 0.1),$ $g_5^3 = weight(v_5)(\lambda_5^3 = 0.6)\}$

Table 2.10: CP : États mentaux initiaux des agents

Étape	t_{cur}	v_1	v_2	v_3	v_4	v_5	it	c_{cur}	Coup(s)
...	...	$c_1 \succ c_3$ $\succ c_2$	$c_1 \succ c_3$ $\succ c_2$	$c_3 \succ c_2$ $\succ c_1$	-	-	$c_1 \succ c_3$ $\succ c_2$	c_3	...
8	2	-	-	-	$c_1 \succ c_3$	-	-	c_3	$v_4 : m_{16}$
9		-	-	$c_1 \succ c_3$ $\succ c_2$	-	-	-	c_3	$v_3, v_4 : m_{14}$
10		-	-	-	-	-	-	c_1	
11	3	$c_1 \succ c_2$	$c_1 \succ c_2$	$c_1 \succ c_2$	$c_1 \succ c_2$	$c_1 \succ c_2$ $\succ c_5$	$c_1 \succ c_2$	c_1	...

Table 2.11: CP : Processus de négociation

Nom du coup	Description du coup	Argument positif	Argument
m_{16}	Argue(c_1, Arg_9)	oui	$Arg_9 =< \{t_4^{c_1} - t_3^{c_1} \leq 3, t_4^{c_1} - t_3^{c_1} \leq 3 \rightarrow$ $group(v_4)\}, \{group(v_4)\}, c_1, w_4 >$

Table 2.12: CP : Coups utilisés dans la négociation

2.4 Conclusion

Nous avons présenté dans ce chapitre un mécanisme de régulation du droit de passage des véhicules à une intersection, s'appuyant sur un mécanisme de négociation automatique entre les véhicules. Nous présentons dans le chapitre suivant un nouveau mécanisme s'appuyant sur ce dernier, permettant de prendre en compte la présence de transports en commun, et en particulier de bus, dans le trafic.

3 Coordination des intersections pour le passage du bus

Dans les zones urbaines, l'utilisation de transports publics est largement encouragée. Dans certaines zones les bus disposent de lignes dédiées, mais ils sont souvent intégrés au trafic de la même manière que les véhicules particuliers. Pour rendre les bus plus attractifs, les exploitants cherchent à assurer leur efficacité et leur fiabilité, néanmoins les bus sont soumis aux conditions de trafic. Afin de diminuer l'impact de celles-ci, il est possible d'accorder aux bus la priorité aux intersections, cependant une simple priorité n'est pas efficace. Dans un contexte de trafic dense, la priorité simple n'améliore pas nécessairement la progression du bus, et peut même parfois dégrader l'état du trafic.

Les technologies de communication dans les véhicules sont de plus en plus perfectionnées et répandues, ce qui offre de nouvelles perspectives pour la régulation du trafic. Des mécanismes de régulation s'appuyant sur les informations en temps réel sont possibles et permettent aux intersections d'appliquer une politique de régulation plus fine. Dans ce chapitre, nous développons un mécanisme distribué fondé sur la coordination permettant aux bus de réserver un droit de passage à travers l'intersection de manière anticipative. Avec ce mécanisme, chaque intersection sur la trajectoire du bus adapte sa politique de régulation afin de fournir un itinéraire relativement dégagé au bus, et de permettre ainsi au bus d'atteindre son prochain arrêt à la date prévue. La politique d'une intersection s'appuie sur un modèle bi-niveau de coordination avec les véhicules.

3.1 Introduction

Une approche courante pour avantager les bus dans le trafic consiste à leur accorder la priorité à l'intersection. Avec la priorité simple, les bus approchant d'une intersection obtiennent le droit de traverser l'intersection sans s'arrêter. Cependant cette approche est très locale : le bus peut traverser une intersection pour se rendre dans une congestion, qu'il peut aussi renforcer. Élargir le cadre du problème et considérer l'environnement du bus est une tâche difficile car cela nécessite de prendre en compte un environnement complexe avec de nombreux véhicules ayant différents

objectifs, sans créer trop de délais et de congestions sur le réseau, et ce pour chaque bus présent sur le réseau.

Pour ce type de problème, les approches distribuées présentent de bonnes propriétés puisque ces approches permettent de prendre en compte les différentes informations locales. Dans les approches distribuées, et particulièrement dans les systèmes multi-agents, chaque entité ou "agent" peut prendre en compte son propre contexte et ses buts individuels et locaux et les mettre en avant dans un processus de décision collective. Dans notre approche, les intersections sont des agents et communiquent au bus leur capacité à appliquer certaines politiques compte tenu de leur propre contexte local. Cette capacité dépend de différents critères, dont l'état du trafic autour de l'intersection mais aussi autour des intersections voisines, qui sont potentiellement localisées en dehors de l'itinéraire du bus. Communiquer l'ensemble des différentes possibilités aurait un coût communicationnel important. Il faut donc explorer efficacement l'espace de recherche des politiques jointes des intersections. Une exploration complète de cet espace, pour tout le réseau, serait trop coûteuse en termes de communication et de calcul, et une exploration trop courte ne permettrait pas d'obtenir une solution efficace, d'où l'intérêt des approches distribuées qui permettent de trouver un compromis dans cette exploration.

L'itinéraire d'un bus est fait d'une succession d'arrêts de bus, et chaque bus a une table horaire à respecter pour chacun de ces arrêts. Afin de respecter le critère de régularité des bus sur une ligne particulière, la table horaire d'un bus est susceptible d'être modifiée dynamiquement. Cet aspect ne rentre pas dans le cadre de ce travail, nous supposons ici que la table horaire n'est pas dynamique au point de pouvoir modifier la date à laquelle le bus cherche à atteindre son prochain arrêt.

Dans ce chapitre, nous décrivons le procédé par lequel un bus peut réserver son droit de passage jusqu'à son prochain arrêt et obtenir un itinéraire où il limite le risque de congestion. Dans cette approche, le bus se synchronise avec les intersections sur son itinéraire, et les intersections se coordonnent pour laisser le chemin suffisamment dégagé pour que le bus atteigne sa destination à la date prévue. Avec notre méthode, les intersections utilisent le mécanisme de négociation présenté dans le chapitre précédent pour choisir avec les véhicules une politique de régulation permettant le passage du bus.

Ce chapitre est organisé ainsi : La section 3.2 présente le problème abordé dans ce chapitre. Nous le décrivons premièrement à partir d'un scénario simple de progression du bus, qui est ensuite étendu à plusieurs intersections, puis nous décrivons la forme que prend une solution à ce problème. La section 3.3 présente le mécanisme de coordination que nous proposons en réponse à ce problème, en expliquant comment celui-ci s'applique en s'appuyant sur le mécanisme présenté dans le chapitre

précédent, et comment une politique appliquée par une ou plusieurs intersections est évaluée. Ensuite deux mécanismes, ascendant et descendant, sont présentés. Enfin, nous présentons les perspectives de recherche pour ce problème.

3.2 Description du problème

3.2.1 Description d'un scénario basique

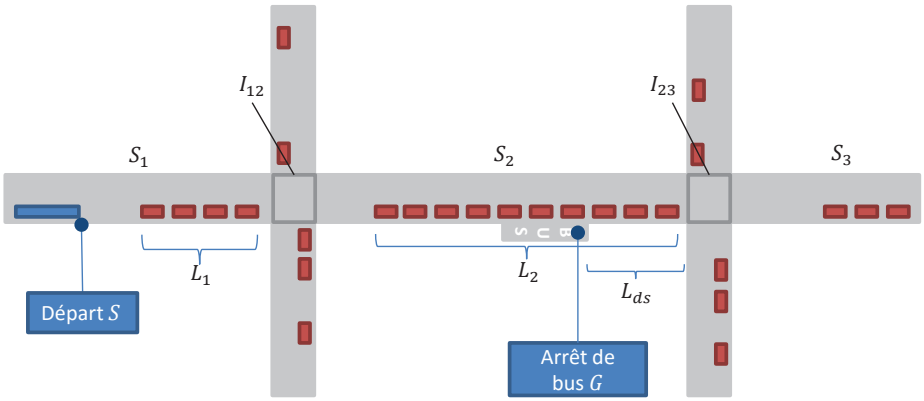


Figure 3.1: Scénario simple pour le problème du bus avec 2 intersections

Un scénario basique est introduit dans la Figure 3.1. Soit un bus b sur une section S_1 à l'instant $t = 0$, qui cherche à atteindre son prochain arrêt G sur la section S_2 à $t_G \leq t_G^{max}$, où t_G^{max} est la date correspondant à son objectif maximum. Plusieurs véhicules circulent sur les sections S_1 et S_2, les charges initiales de ces sections étant, respectivement, L_1 et L_2. $I_{1,2}$ et $I_{2,3}$ sont, respectivement, les intersections entre S_1 et S_2, et entre S_2 et S_3. La voie L_2 peut contenir une charge L_{ds} entre G et $I_{2,3}$.

Soit t L_2^{max} le nombre maximum de véhicules qui peuvent être retenus sur S_2 en aval de G. Tant que $|L_2| \leq L_2^{max}$, le bus b est capable d'atteindre son arrêt G sans aucun ralentissement. Soient $d_S^{I_{1,2}}$ le temps nécessaire à b pour aller de S à $I_{1,2}$ si le trafic est fluide sur S_1, et $d_{I_{1,2}}^{G}(b)$ le temps nécessaire à b pour aller de $I_{1,2}$ à G si le trafic est fluide sur S_2. Soit $t_{in}^{1,2}(b)$ la date d'admission de b dans $I_{1,2}$. Avec des sections fluides, nous avons les contraintes suivantes :
$c_1 : t_{in}^{1,2}(b) \geq t_0 + d_S^{I_{1,2}}(b)$
$c_2 : t_{in}^{1,2}(b) \leq t_G - d_{I_{1,2}}^{G}(b)$

De plus la progression du bus dépend des véhicules qui le précèdent. Soient $Out_{t_0}^{t_G}(S_2)$ et $In_{t_0}^{t_{in}^{1,2}(b)}(S_2)$, respectivement, l'ensemble des véhicules sortant de S_2 (par $I_{2,3}$) durant $[t_0, t_G]$ et l'ensemble des véhicules qui entrent sur la section S_2 (par $I_{1,2}$) durant $[t_0, t_{in}^{1,2}(b)]$. $Out_{t_0}^{t_G}(S_2)$ dépend de la régulation appliquée par $I_{2,3}$ et $In_{t_0}^{t_{in}^{1,2}(b)}(S_2)$ dépend de la régulation appliquée par $I_{1,2}$.

Pour assurer que b sera capable d'atteindre G à temps, nous devons nous assurer qu'il n'y aura pas plus de L_2^{max} véhicules devant b sur S_2 à $t = t_G$, car dans le cas contraire la congestion ne permettra pas à b d'atteindre G. Nous ajoutons donc la contrainte suivante :
$c_3 : |L_2| + |In_{t_0}^{t_{in}^{1,2}(b)}(S_2)| - |Out_{t_0}^{t_G}(S_2)| \leq L_2^{max}$
La valeur de $|In_{t_{in}^{1,2}(b)}^{t_G}(S_2)|$ n'est pas importante ici puisque ces véhicules succèdent à b et ne peuvent donc pas bloquer son accès à G.

On identifie également des contraintes sur $|In_{t_0}^{t_{in}^{1,2}(b)}(S_2)|$ et $|Out_{t_0}^{t_G}(S_2)|$. $I_{2,3}$ a un débit maximal de $T_{2,3}^{max}$ véhicules/pas de temps. De plus certains véhicules de L_1 peuvent chercher à se rendre sur S_2, ils vont donc nécessairement entrer sur S_2 avant le bus b et appartiennent donc à $In_{t_0}^{t_{in}^{1,2}(b)}(S_2)$. Soit $V(L_1, S_2) \subseteq L_1$ l'ensemble des véhicules de L_1 se rendant sur la section S_2 après avoir traversé l'intersection $I_{1,2}$ (cf. Figure3.1). Nous avons donc les contraintes suivantes :
$c_4 : |Out_{t_0}^{t_G}(S_2)| \leq T_{2,3}^{max} * (t_G - t_0)$
$c_5 : |In_{t_0}^{t_{in}^{1,2}(b)}(S_2)| \geq |V(L_1, S_2)|$

À partir de ce scénario, nous observons 5 types de contraintes. c_1 et c_2 sont triviales, et si $d_S^{I_{1,2}}(b) + d_{I_{1,2}}^G(b) \geq t_G - t_0$, alors t_G n'est pas une date réalisable par le bus. De plus c_4 et c_5 fournissent une contrainte de faisabilité additionnelle car si $L_2 + V(L_1, S_2) - Out_{t_0}^{t_G}(S_2) > L_{ds}$ alors la contrainte c_3 ne peut pas être satisfaite.

3.2.2 Généralisation à n intersections

Le scénario représenté par la Figure 3.2 est une généralisation du scénario précédent à n sections : le bus a des sections additionnelles à franchir avant d'atteindre sont but. Les raisonnements qui ont servi à construire les contraintes précédentes peuvent être appliqués de la même manière à ce scénario, ce qui nous permet d'obtenir les contraintes suivantes, en s'appuyant sur les notations précédentes.

- c_1 : le bus b doit traverser la distance le séparant de chaque intersection. Nous avons donc :

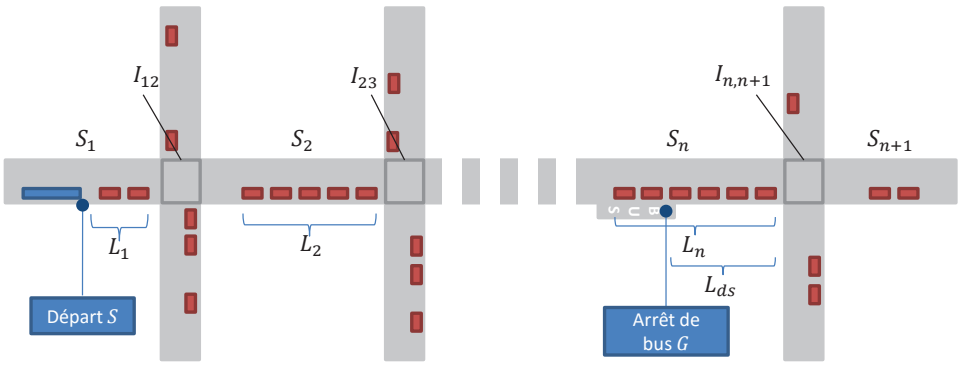


Figure 3.2: Généralisation du problème à n intersections

- pour $I_{1,2}$: $t_{in}^{1,2}(b) \geq t_0 + d_S^{I_{1,2}}(b)$

- pour les autres intersections : $\forall k \in [3, n]$,

$t_{in}^{k-1,k}(b) \geq t_{in}^{1,2}(b) + \sum_{j=3}^{k}(d_{I_{j-2,j-1}}^{I_{j-1,j}}(b))$

(car $\forall k \in [3, n], t_{in}^{k-2,k-1} + d_{I_{k-2,k-1}}^{I_{k-1,k}} \leq t_{in}^{k-1,k}$)

- c_2 : après avoir traversé l'intersection, le bus b doit traverser la distance le séparant de sa destination avant la date objectif. Nous avons donc :

- pour $I_{n-1,n}$: $t_{in}^{n-1,n}(b) \leq t_G - d_{I_{n-1,n}}^{G}(b)$

- pour les autres intersections : $\forall k \in [2, n-1]$,

$t_{in}^{k-1,k}(b) \leq t_{in}^{n-1,n}(b) - \sum_{j=k+1}^{n}(d_{I_{j-2,j-1}}^{I_{j-1,j}}(b))$

- c_3 : pour permettre au bus b d'atteindre un point particulier (une intersection ou sa destination), la section précédant ce point doit être dégagée. Nous avons donc :

- pour S_1 : $|L_1| = |Out_{t_0}^{t_{in}^{1,2}(b)}(S_1)|$

- pour S_n : $|L_n| + |In_{t_0}^{t_{in}^{n-1,n}(b)}(S_n)| - |Out_{t_0}^{t_G}(S_n)| \leq L_n^{max}$

- pour les autres sections : $\forall k \in [2, n-1]$,

$|L_k| + |In_{t_0}^{t_{in}^{k-1,k}(b)}(S_k)| = |Out_{t_0}^{t_{in}^{k,k+1}(b)}(S_k)|$

- c_4 : chaque intersection a un débit maximal qui doit être pris en compte. Nous avons donc :

  - pour S_n : $|Out_{t_0}^{t_G}(S_n)| \leq T_{n,n+1}^{max} * (t_G - t_0)$

  - pour les autres sections : $\forall k \in [2, n-1]$,

  $|Out_{t_0}^{t_{in}^{k,k+1}(b)}(S_k)| \leq T_{k,k+1}^{max} * (t_{in}^{k,k+1}(b) - t_0)$

- c_5 : chaque section doit admettre tous les véhicules en aval d'un bus ayant cette intersection sur son itinéraire avant d'admettre ce bus. Nous avons donc :

  - pour S_n :

  $|In_{t_0}^{t_{in}^{k-1,k}(b)}(S_k)| \geq |\bigcup_{j=2}^{k} V(L_{j-1}, \bigcap_{i=j}^{k} S_i)|$

  - pour les autres sections : $\forall k \in [2, n]$,

  $|In_{t_0}^{t_{in}^{k-1,k}(b)}(S_k)| \geq |\bigcup_{j=2}^{k} V(L_{j-1}, \bigcap_{i=j}^{k} S_i)|$

  (où $V(L_j, \bigcap_{i=p}^{q} S_i) \subseteq L_j$ est l'ensemble des véhicules de L_j pour lesquels les prochaines sections sur leur itinéraire après $I_{p-1,p}$ sont $S_p, S_{p+1}, ..., S_q$)

3.2.3 Politique de l'intersection

Pour résoudre ce problème, notre objectif est d'identifier une progression $P(b)$ pour le bus b, et une politique jointe $\Pi(b)$ fondée sur cette progression. $P(b)$ est l'ensemble des dates d'admission du bus b dans toutes les intersections sur son itinéraire jusqu'à

$G : P(b) = \{t_{in}^{1,2}(b), ..., t_{in}^{n-1,n}(b), t_g(b)\}$, où $t_g(b)$ est la date à laquelle le bus b atteint effectivement G. La progression doit respecter toutes les contraintes décrites ci-dessus. $\Pi(b)$ est l'ensemble des politiques de ces intersections : $\Pi(b) = \pi_{1,2}, ..., \pi_{n-1,n}$ où chaque $\pi_{k,k+1}$ est la politique de $I_{k,k+1}$.

En première approximation, une politique est un tuple $(t_{in}^{k,k+1}(b), Add_{k,k+1}, Pass_{k,k+1})$ où $Add_{k,k+1}$ est l'ensemble des véhicules traversant $I_{k,k+1}$ depuis une section autre que S_k qui vont sur S_{k+1} durant $[0, t_{in}^{k,k+1}(b)]$, et $Pass_{k,k+1}$ est l'ensemble des véhicules traversant $I_{k,k+1}$ depuis une autre section que S_k et qui ne vont pas sur S_{k+1} pendant $[0, t_{in}^{k,k+1}(b)]$. Un ensemble de politiques $\pi_{1,2}, ..., \pi_{k,k+1}$ détermine $Out_{t_0}^{t_{in}^{k,k+1}(b)}(S_k)$ et $In_{t_0}^{t_{in}^{k,k+1}(b)}(S_{k+1})$ car on peut affirmer :

$$|Out_{t_0}^{t_{in}^{k,k+1}(b)}(S_k)| = |In_{t_0}^{t_{in}^{k-1,k}(b)}(S_k)| + |L_k| \text{ (voir } c_3)$$

et

$$|In_{t_0}^{t_{in}^{k,k+1}(b)}(S_{k+1})| = |V(Out_{t_0}^{t_{in}^{k-1,k}(b)}(S_{k-1}), S_{k+1})| + |Add_{k,k+1}|.$$

Nous avons donc récursivement :

$$|Out_{t_0}^{t_{in}^{k,k+1}(b)}(S_k)| = |L_k| + \sum_{j=1}^{k-1} |V(Out_{t_0}^{t_{in}^{j,j+1}(b)}(S_j), \bigcap_{i=j}^{k} S_i)|$$
$$= \sum_{j=1}^{k} |V(L_j, \bigcap_{i=j}^{k} S_i)| + \sum_{j=1}^{k-1} |V(Add_{k,k+1}, \bigcap_{i=j}^{k} S_i)|$$

Cette équation montre que la régulation au niveau d'une intersection a des conséquences immédiates sur les autres intersections et montre l'existence d'interdépendances entre les intersections. Notre cadre de travail nécessite une définition de la politique d'une intersection, qu'on appelle politique "individuelle". On appelle "politique jointe" un ensemble de politiques individuelles concernant des intersections différentes. Une politique jointe est dite "totale" si elle contient une politique pour chaque intersection de la trajectoire du bus, et elle est dite "partielle" sinon.

Les politiques individuelles doivent assurer leur compatibilité, les politiques de deux intersections étant considérées comme compatibles si une politique jointe partielle constituée de ces deux politiques ne viole aucune des contraintes décrites précédemment. Nous définissons donc la politique individuelle $\pi_{k,k+1}$ d'une intersection $I_{k,k+1}$ en bornant les valeurs des paramètres ci-dessus, de la manière suivante :

$$\pi_{k,k+1} = (t_{in}^{k,k+1}(b), MaxOut_{t_0}^{t_{in}^{k,k+1}(b)}(S_k), MaxPass_{k,k+1},$$
$$\bigcup_{j=k}^{n} MaxV(In_{t_0}^{t_{in}^{k,k+1}(b)}(S_{k+1}), \bigcap_{i=k}^{j} S_i)) \text{ , où :}$$

- $MaxOut_{t_0}^{t_{in}^{k,k+1}(b)}(S_k)$ est la taille maximale de $Out_{t_0}^{t_{in}^{k,k+1}(b)}(S_k)$
- $MaxPass_{k,k+1}$ est la taille maximale de $Pass_{k,k+1}$
- $MaxV(In_{t_0}^{t_{in}^{k,k+1}(b)}(S_{k+1}), \bigcap_{i=k}^{j} S_i)$ est la taille maximale de $V(In_{t_0}^{t_{in}^{k,k+1}(b)}(S_{k+1}), \bigcap_{i=k}^{j} S_i)$

3.3 Mécanisme de coordination

Nous cherchons à identifier et à appliquer des politiques aux intersections sur l'itinéraire du bus. Avant de détailler la méthode de recherche utilisée afin de déterminer ces politiques, nous allons d'abord expliquer comment une politique peut être choisie à une intersection donnée.

3.3.1 Mécanisme de régulation des intersections

Le chapitre 2 propose une méthode pour la régulation de trafic à l'échelle de l'intersection s'appuyant sur la négociation. Dans cette méthode fondée sur un modèle multi-agent, les véhicules à l'approche d'une intersection négocient une "configuration", i.e. un ensemble de dates d'admission dans l'intersection pour chaque véhicule à l'approche de celle-ci. Dans cette négociation, chaque véhicule est capable de proposer des configurations. Les véhicules doivent s'accorder sur une de ces configurations, utilisant des arguments sur l'état du trafic afin d'influencer les décisions des autres. Ces configurations sont construites par les véhicules en utilisant des contraintes obligatoires (s'appuyant sur la position physique de chaque véhicule) et des contraintes de sécurité (évitant les collisions au sein de la zone de conflit en utilisant un délai de sécurité). De plus, les véhicules peuvent utiliser des contraintes optionnelles pour guider leurs explorations individuelles de l'espace de recherche des configurations.

Négociation du droit de passage

Tous les véhicules sur le réseau (présents et futurs) ont des intérêts communs, comme éviter les congestions, qui sont difficiles à mesurer individuellement. Pour représenter ces "intérêts du réseau", les infrastructures, dont les intersections, peuvent fournir des arguments et générer des configurations pour guider le processus de décision collective des véhicules. Une fois une configuration choisie, et jusqu'à ce que les véhicules décident collectivement de la remplacer par une autre, cette configuration fait office d'instruction de régulation. De la même manière que les couleurs des feux de signalisation sur des intersections à feux, cette politique détermine qui a le droit de passage et doit donc être respectée par les véhicules. Des mécanismes de sécurité

garantissent la prévention des conflits même si aucun accord n'est trouvé par les véhicules.

Attribution des poids aux véhicules

Dans ce mécanisme, chaque véhicule dispose d'un poids déterminant son importance dans les négociations. Ce poids est fixé par l'infrastructure en fonction du niveau de coopération de chaque véhicule : un véhicule choisissant une configuration pour laquelle les arguments fournis sont globalement positifs obtiendra une augmentation de poids, et inversement une pénalité de poids s'il choisit une configuration ayant des arguments globalement opposés à elle. Ce système d'attribution de poids incite les véhicules à la coopération mais laisse à chaque agent la possibilité de participer comme il l'entend à une décision collective et à prendre des décisions de manière indépendantes. Signalons que ce mécanisme ne prend en compte que les véhicules approchant directement de l'intersection, c'est-à-dire dans les secondes et les dizaines de secondes à venir. Cependant, les véhicules approchent de l'intersection de manière continue et rendent le problème dynamique en modifiant constamment l'ensemble des véhicules auxquels la configuration de l'intersection doit s'appliquer. Un mécanisme de gestion de la continuité traite cette difficulté.

Utilisation de ce mécanisme à plus large échelle

Le mécanisme décrit ci-dessus n'est pas suffisant pour résoudre le problème décrit dans la section 3.2, car il ne considère qu'une intersection, sur un horizon de temps court. Il peut cependant être utilisé pour intégrer toutes sortes de contraintes limitant l'espace de recherche de politiques pour l'intersection, ce qui est utile pour un mécanisme considérant un problème de plus large échelle, qui s'appuierait sur celui-ci. En effet il peut ainsi appliquer, ou au moins recommander, une politique individuelle telle que nous l'avons définie dans la section 3.2.3 sous la forme d'un ensemble de trois contraintes. Remarquons que ces politiques ne peuvent pas être appliquées de manière déterministe puisque les véhicules sont indépendants et peuvent collectivement décider de choisir une politique peu recommandée.

Dans ce chapitre, nous nous appuyons sur ce mécanisme pour appliquer des politiques aux intersections, ce qui nous permet de décrire une politique comme un ensemble de contraintes qui peuvent s'appliquer à cette intersection.

Pour donner un exemple, supposons que notre mécanisme ait produit un ensemble de politiques pour les différentes intersections situées sur la trajectoire du bus b.

Appelons $t_{in}^{I}(b)$ la date de passage attribuée à b pour la traversée de l'intersection I. Notre mécanisme a produit une politique $\pi_I = \{Add_I, Pass_I\}$. Soient S_{in} la section par laquelle le bus entrera dans l'intersection I et S_{out} la section par laquelle il en sortira. Les politiques des intersections précédant I nous permettent de connaitre le nombre de véhicules qui arriveront par S_{in} avant l'arrivée de b, c'est-à-dire le nombre de véhicules $|Out_{t_0}^{t_{in}^{I}(b)}(S_{in})|$ à admettre nécessairement avant $t_{in}^{I}(b)$. Nous savons donc que sur l'intervalle $[t_0, t_{in}^{I}(b)]$, I devra admettre $|Out_{t_0}^{t_{in}^{I}(b)}(S_{in})|$ véhicules de la voie S_{in}, et $|Add_I| + |Pass_I|$ véhicules des autres voies, en respectant $|In_{t_0}^{t_{in}^{I}(b)}(S_{out})| = |Out_{t_0}^{t_{in}^{I}(b)}(S_{in})| + |Add_I|$.

Si $t_{in}^{I}(b) \leq h_t$ où h_t est l'horizon de temps considéré par la négociation, il est très simple de transformer ceci sous la forme de contraintes qui peuvent être reprises telles quelles dans les arguments des agents. L'agent intersection ne se contente pas d'encadrer la négociation, et peut formuler des arguments en leur attribuant un poids très élevé pour guider la négociation entre les véhicules afin de les mener à produire une solution respectant ces contraintes.

Dans les cas où $t_{in}^{I}(b) > h_t$, l'application de la politique de l'intersection est plus difficile. En effet celle-ci doit assurer le respect des contraintes sur un horizon de temps supérieur à celui de la négociation. Il faut pour cela découper les différentes contraintes ci-dessus dans le temps. Nous proposons de faire ce découpage de manière proportionnelle à l'intervalle de temps considéré.
Par exemple, si $t_{in}^{I}(b) = k * h_t$ où $k > 1$, on aura $|Out_{t_0}^{h_t}(S_{in})| = \frac{1}{k} * |Out_{t_0}^{t_{in}^{I}(b)}(S_{in})|$.
On procède de cette manière pour les différentes contraintes ainsi considérées.
Il serait toutefois possible de concevoir des méthodes plus sophistiquées s'appuyant sur la position de chaque véhicule du réseau pour réaliser un découpage plus pertinent.

3.3.2 Politique et application

Les véhicules prenant part à la négociation à l'échelle de l'intersection ne sont pas strictement contraints à respecter les indications de l'agent intersection. Ils peuvent ne pas respecter ces indications, par exemple si trop de véhicules attendent devant l'intersection. Afin de prendre des engagements plus faciles à respecter, l'agent intersection doit anticiper le comportement des véhicules à l'approche.

Le comportement des véhicules n'est pas prévisible, mais plus il y a de véhicules approchant de l'intersection, plus un engagement de l'intersection à retenir ces véhicules lui est difficile à respecter.

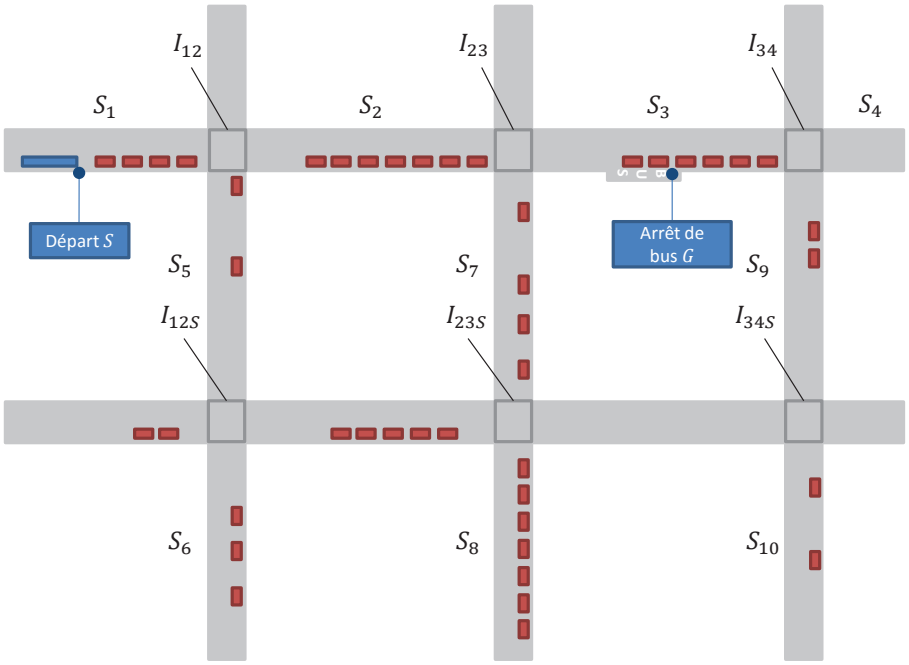


Figure 3.3: Dans ce scénario, la faisabilité de la politique de I_{23} dépend des véhicules sur S_7 à $t_{in}^{2,3}(b)$. La politique de I_{23} pourrait permettre de réduire le nombre de véhicules sur S_7, et donc augmenter la faisabilité de la politique de I_{23}. I_{23} a donc intérêt à se coordonner avec I_{23S}.

Par exemple, dans le scénario présenté Figure 3.3, il y a quelques véhicules approchant I_{12} à t_0. Si I_{12} prend un engagement sur $t_{in}^{1,2}$, et donc sur la valeur de $Out_{t_0}^{t_{in}^{1,2}(b)}(S_1)$, il est raisonnable de supposer que les véhicules sur S_5 ne pourront pas s'opposer avec succès à la politique de I_{12}. Il y a peu de véhicules sur S_5, les véhicules sur S_1 ne partagent probablement pas leurs intérêts, en particulier le bus qui a une importance particulière dans la négociation, donc à moins que les véhicules sur S_5 n'aient des poids particulièrement élevés, s'opposer à la politique de I_{12} serait coûteux et probablement inutile, c'est pourquoi cette situation reste improbable.

À l'inverse, de nombreux véhicules sont présents sur les sections S_7 et S_8. Il est donc moins improbable que les véhicules approchant de I_{23} s'opposent à cette politique, ce qui les forcerait à patienter, et réussissent à emporter la négociation, en empêchant ainsi la réalisation de cette politique.

L'anticipation du flux de véhicules dépend des intersection voisines. Nous appelons "intersections backbone" les intersections sur la trajectoire du bus, et "intersections alliées" les intersections dans le voisinage des intersections backbone disposées à adapter leurs politiques si nécessaire. Soient BI l'ensemble des intersections backbone et $All(I_x)$ l'ensemble des intersections alliées qui peuvent influencer l'entrée de véhicules sur l'intersection I_x, c'est-à-dire les intersections géographiquement proches de I_x.

3.3.3 Fonctions d'évaluation des politiques

Nous proposons un mécanisme distribué où chaque intersection backbone est capable de proposer des politiques individuelles la concernant, et de communiquer avec le bus qui choisit quelle politique doit être appliquée. Nous supposons que chaque intersection est coopérative avec le bus et réalise donc une exploration efficace de son espace de recherche et communique des informations exactes. Dans notre mécanisme, le bus fournit un ensemble de contraintes à chaque intersection, et chaque intersection doit lui proposer un ensemble de politiques. Elle cherche à fournir des solutions qui sont "efficaces" et "diverses". Nous expliquons ici comment évaluer la qualité d'une politique en lui attribuant un coût. Différents critères permettent d'évaluer le coût d'une politique :

- Le temps d'attente des véhicules (autres que le bus) : ce critère a un double avantage. Premièrement, nous préférons les solutions qui ne ralentissent pas le trafic environnant. Deuxièmement, plus les véhicules doivent attendre, plus ils sont susceptibles de s'opposer à la politique et à l'empêcher. Ce critère est notre critère principal, et il doit être minimisé. Nous appelons "coût principal"

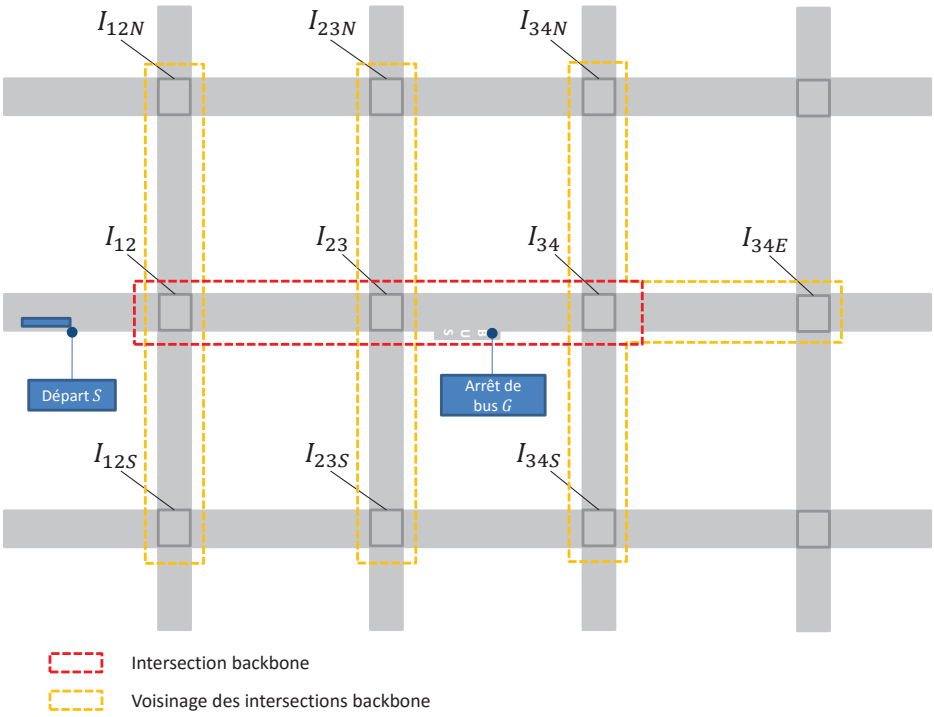


Figure 3.4: Les intersections I_{12},I_{23} et I_{34} sont des intersections backbone car elles sont sur la trajectoire du bus. Chacune de ces intersections a de possibles intersections alliées dans le voisinage, par exemple I_{12N} et I_{12S} sont des alliées de I_{12}.

d'une politique le temps d'attente total des véhicules de la zone englobant l'intersection et ses alliées.

- Urgence de la décision : si la date d'admission du bus dans les premières intersections de son itinéraire est proche, le mécanisme de coordination peut être trop lent pour l'appliquer. Le coût des politiques ayant une date d'admission proche (sous un seuil particulier ad_{th}) peut être réduit afin de privilégier ces solutions durant l'exploration.

- Difficulté de trouver une solution locale (pour une intersection particulière) : si l'espace de recherche pour cette intersection est très restreint, il peut être intéressant de choisir rapidement une de ses politiques et de construire la solution jointe sur la base de celle-ci. Ainsi les politiques pour une intersection ayant un espace de recherche limité ont une réduction de coût.

- Coût de la coopération : certaines configurations peuvent nécessiter un effort de la part des intersections alliées en leur demandant d'appliquer une certaine politique. Quand cela a lieu, la politique recommandée par ces intersections est susceptible de ne pas être suivie par les véhicules, ce qui pourrait détériorer la politique de l'intersection backbone. Les politiques nécessitant une action d'une intersection alliée prennent donc un risque additionnel, qui peut-être pris en compte comme une augmentation du coût.

- Adaptabilité de la solution à des changements possibles (nouveau bus, prévision du trafic non réalisée) : dans le cas où la prévision de la progression des véhicules serait incorrecte, une modification dynamique de la politique jointe peut être nécessaire. Les bus avec des dates d'admission basses sont plus susceptibles de pouvoir s'adapter pour passer plus tôt aux intersections suivantes, ce qui n'est pas possible avec des dates d'admission hautes. C'est pourquoi les politiques où le bus a une date d'admission élevée reçoivent une pénalité proportionnelle à cette date.

Tous ces critères peuvent être utilisés pour évaluer la politique d'une intersection. Il y a différents paramètres et le réglage de ces paramètres peut dépendre du scénario (nombre d'intersections, nombre de voies sur chaque section, trafic moyen, etc). Nous appelons $indivEval(\pi)$ cette fonction objectif fournissant le coût de la politique π d'une intersection backbone.

Une politique jointe est un ensemble de politiques de plusieurs intersections backbone, et doit aussi être évaluée. La stratégie d'exploration utilisée par le bus pour explorer l'espace de recherche des politiques jointes est inspirée des méthodes

"branch and bound", la fonction d'évaluation d'une politique jointe doit donc avoir la propriété suivante :

$$\forall(A,B),(A \subseteq B) \Rightarrow (eval(A) \leq eval(B))$$

Nous utilisons donc $eval(\Pi) = max(\bigcup_{\pi \in \Pi}(indivEval(\pi)))$, avec $eval(null) = +\infty$ pour qu'une politique réalisable soit toujours préférée à une politique irréalisable.

3.3.4 Stratégie de proposition par les intersections

Dans notre mécanisme, le bus demande aux intersections backbone des politiques possibles, prenant en compte une politique jointe partielle et les contraintes en résultant. Les intersections doivent fournir un ensemble de solutions qui sont bonnes d'après la fonction d'évaluation utilisée (voir 3.3.3) et aussi variées. Nous allons maintenant expliquer comment ces politiques sont générées et choisies.

Pour une intersection $I_{k,k+1}$, l'espace des politiques possibles est un espace à $|insec_{k,k+1}|$ dimensions, où $insec_{k,k+1}$ est l'ensemble des sections d'entrée de $I_{k,k+1}$. Une dimension correspond aux dates d'admissions possibles pour le bus dans l'intersection, bornée par les contraintes décrites en 3.2. Les $|insec_{k,k+1}| - 1$ autres dimensions correspondent au nombre de véhicules en provenance de chaque section $S_x \neq S_k$ qui sont admis dans $I_{k,k+1}$ avant $t_{in}^{k,k+1}(b)$.

Plusieurs heuristiques peuvent être utilisées pour explorer cet espace de recherche. Nous présentons ici une d'entre elles, développée dans l'Algorithme 4. L'intersection produit les politiques obtenues par l'appel de PROPOSEPOLICIES($partialJP$), où $partialJP$ est la politique jointe partielle sur laquelle s'appuient ces politiques.

La date d'admission du bus est bornée par les contraintes et un pas de temps est défini pour cette date, nous obtenons donc un ensemble de dates d'admission possibles. Pour chacune de ces dates, et pour chaque voie d'entrée, nous procédons de la manière suivante.

Soient $insec_{k,k+1}$ l'ensemble des sections entrant dans $I_{k,k+1}$ et $nv(s)$ l'ensemble des véhicules de la section s, telle que $s \in I_{k,k+1}$ et $s \neq S_k$, qui peuvent s'engager dans $I_{k,k+1}$ avant $t_{in}^{k,k+1}(b)$ (i.e., avant t_{max}). Nous considérons différents sous-ensembles $firstV(x, nv(s)) \in nv(s)$ représentant les x véhicules de $nv(s)$ qui sont les plus proches de l'intersection (les x premiers véhicules approchant de l'intersection sur s). Nous cherchons des groupes notables de véhicules, par exemple des pelotons, pour proposer une division significative et pertinente des véhicules. La fonction $pointOfInterst(firstV(x, nv(s)))$ détermine si le fait d'admettre les x premiers

véhicules de $nv(s)$ représente un intérêt particulier (fin ou début de peloton, découpage représentant un intérêt particulier pour une intersection adjacente, etc). Cette fonction peut opérer de différentes manières, néanmoins nous supposons ici qu'il existe au moins un point d'intérêt dans $nv(s)$ pour tout s. Ces découpages particuliers pour la section s sont ajoutés à l'ensemble $sectionGroups$. Les ensembles $sectionGroups$ correspondant à chaque section s sont ajoutés à l'ensemble $groups$. La fonction $combine(groups)$ retourne l'ensemble des combinaisons de tous ces découpages pour chacune des sections.

Algorithme 4 Algorithme de proposition d'une intersection

function PROPOSEPOLICIES($partialJP$)
  for $(t = t_{min}; t \leq t_{max}; t \leftarrow t + timestep)$ **do**
    $groups \leftarrow \emptyset$
    for all $s \in insec_{k,k+1}$ **do**
      if $s \neq S_k$ **then**
        $sectionGroups \leftarrow \emptyset$
        for $(x = 0; x < |nv(s)|; x \leftarrow x + 1)$ **do**
          if $(pointOfInterst(firstV(x, nv(s))))$ **then**
            $sectionGroups \leftarrow sectionGroups \cup \{firstV(x, nv(s))\}$
          end if
        end for
        $groups \leftarrow groups \cup \{sectionGroups\}$
      end if
    end for
  end for
  $policies \leftarrow combine(groups)$
  return $policies$
end function

3.3.5 Stratégie descendante

Dans notre approche, nous supposons que l'espace de recherche d'une solution jointe est trop large pour une exploration exhaustive (à cause des coûts communicationnels). Chaque intersection backbone peut construire des propositions pour ses politiques individuelles en utilisant l'anticipation et la communication avec ses alliées.

Le bus utilise une stratégie "branch and bound" s'appuyant sur les propositions des intersections backbone pour construire un arbre dont chaque noeud est une politique individuelle pour une intersection, décrit dans l'algorithme 5. La fonction CHOOSESET est une fonction récursive retournant une politique jointe, possiblement partielle, définie ainsi :
CHOOSESET($partialJP, proposals, receivers, bestScore$), où :

- $partialJP$ est une politique jointe partielle ayant déjà été choisie pour cette iteration : la politique jointe retournée inclut nécessairement $partialJP$ (exceptée si celle-ci est $null$).

- $proposals$ est l'ensemble des propositions envisagées. Si $proposals = null$, le bus peut demander des propositions de politiques à l'intersection backbone, sinon il explore uniquement les possibles politiques jointes construites par l'union de $partialJP$ et d'une des politiques $\pi \in proposals$.

- $receivers$ est l'ensemble des intersections de BI pour lesquelles il n'y a pas encore de politique définie dans $partialJP$.

- $bestScore$ est le score de la meilleure politique jointe totale qui a été trouvée dans l'arbre.

L'algorithme de recherche est récursif, et le premier appel est fait de la manière suivante : ChooseSet$(\emptyset, \emptyset, BI, +\infty)$.

Dans cet algorithme, le bus demande à chaque intersection backbone de proposer un ensemble de politiques individuelles la concernant puis il choisit de manière gloutonne la meilleure politique de toutes les intersections backbone et l'ajoute à la politique jointe partielle, construisant ainsi un arbre. Le bus envoie la politique jointe ainsi construite à chaque intersection backbone et recommence jusqu'à avoir une politique jointe complète, ou jusqu'à ce qu'il ne reçoive plus de proposition de politique individuelle compatible avec la politique jointe partielle. Cela constitue la "branche gauche" de l'arbre d'exploration. L'algorithme explore ensuite différentes "branches droites", c'est-à-dire des branches pour lesquelles certaines bonnes politiques proposées par les intersections ont été ignorées. La politique jointe choisie à chaque noeud de l'arbre peut être évaluée, et une fois qu'une solution est trouvée, il est possible d'ignorer les noeuds pour lesquels la borne inférieure (i.e. l'évaluation de la politique jointe partielle correspondant à ce noeud) est supérieure à la meilleure solution trouvée.

Les actes de langage utilisés entre le bus et les intersections dans cette méthode sont les suivants : $Acts = \{Request, Propose, Accepted\}$.

Request$(bi, t_G, partialJP)$: avec cet acte, le bus demande à l'intersection bi à qui ce message est envoyé (pour laquelle $partialJP$ ne dispose pas encore de politique) de proposer un ensemble de politiques permettant de compléter la politique jointe partielle $partialJP$ de manière à ce que le bus atteigne son prochain arrêt avant la date t_G. Le bus ne fait qu'une requête à la fois à chaque intersection.

Propose$(policies, t_G, partialJP)$: avec cet acte, une intersection propose un ensemble de solutions $policies$ répondant à la demande $Request(bi, t_G, partialJP)$ du bus. Si $policies = \emptyset$, alors l'intersection ne dispose d'aucune solution pour cette demande. Ce message est envoyé au bus une fois par chaque intersection après réception d'un message $Request$.

Accepted(JP) : avec cet acte, le bus indique aux intersections que la politique jointe JP a été retenue et doit être appliquée. Ce message est envoyé par le bus aux intersections après avoir complété sa politique jointe, c'est-à-dire quand celle-ci n'est plus partielle et contient bien une politique par intersection, ces politiques étant compatibles entre elles.

Les algorithmes "branch and bound" sont généralement des méthodes de résolution exactes. Cependant dans notre problème nous n'explorons que les politiques fournies par les intersections, il peut donc arriver que certaines solutions ne puissent pas être atteintes par cette méthode. Cependant si ce cas est suffisamment rare, alors cet algorithme est une heuristique acceptable.

Exemple Considérons une série de sections S_1, S_2, S_3 et d'intersections $BI = \{I_{1,2}, I_{2,3}\}$ de débit maximal $T_{1,2}^{max} = T_{2,3}^{max} = 1$ véhicule par unité de temps. Soit un bus b devant atteindre l'arrêt G situé sur la section S_3 avant la date $t_G = 90$, sachant que les durées de traversée dans des conditions fluides sont $d_S^{I_{1,2}} = d_{I_{1,2}}^{I_{2,3}} = d_{I_{2,3}}^{G} = 20$ et sachant que les charges initiales sont $L_1 = 5, L_2 = 30, L_3 = 0$, et que les itinéraires des véhicules constituant L_1 et L_2 passent tous par $I_{2,3}$. Par souci de clarté, on suppose pour ce scénario qu'il n'y a pas de congestion possible sur S_3. Pour simplifier la lecture, on définit $Add_{k,k+1} = V(In_{t_0}^{t_{in}^{k,k+1}(b)}(S_{k+1}), \bigcap_{i=k+1}^{3} S_i) \setminus Out_{t_0}^{t_{in}^{k,k+1}(b)}(S_k)$, c'est-à-dire l'ensemble des véhicules entrant dans l'intersection I_k^{k+1} durant $[t_0, t_{in}^{k,k+1}(b)]$ qui ne viennent pas de S_k et pour qui S_3 fait partie de leur itinéraire. En particulier, les véhicules de $Add_{1,2}$ sont des véhicules qui emprunteront $I_{2,3}$ avant le bus et doivent par conséquent être pris en compte. On appelle $MaxAdd_k^{k+1}$ la taille maximale de Add_k^{k+1}. Par souci de simplicité, nous ne détaillons pas la fonction objectif utilisée et décrirons les préférences générées au cas par cas.

Le bus b appelle la fonction ChooseSet$(\emptyset, \emptyset, BI, +\infty)$. Il envoie à chaque intersection $bi_{i,i+1} \in BI$ la requête $Request(bi_{i,i+1}, t_G, \{null, null\})$. Compte tenu des distances entre les intersections, les intersections sont capables de déterminer que $20 \leq t_{in}^{1,2}(b) \leq 50$ et que $40 \leq t_{in}^{2,3}(b) \leq 70$. Chaque intersection $bi_{i,i+1} \in BI$ retourne le message $Propose(pol_{i,i+1}, t_G, \{null, null\})$ avec $pol_{1,2} = \{\pi_{11}, \pi_{12}\}, pol_{2,3} = \{\pi_{21}, \pi_{22}\}$, sachant que :
- π_{11} est décrite ainsi : $t_{in}^{1,2}(b) = 20$, $MaxOut_{t_0}^{t_0+20}(S_1) = 10$, $MaxAdd_{1,2} = 5$, $MaxPass_{1,2} = 5$

Algorithme 5 Algorithme du bus

```
function ChooseSet(partialJP, proposals, receivers, bestScore)
    bestEval ← +∞
    bestProposal ← null
    chosenInters ← null
    if (proposals = ∅) then
        for all bi ∈ receivers do
            Request(bi, t_G, partialJP)
            proposals ← proposals ∪ {Propose(bi).policies}
        end for
    end if
    for all prop ∈ proposals do
        if (eval(partialJP ∪ prop) < bestEval) then
            bestProposal ← prop
            bestEval ← eval(partialJP ∪ prop)
            chosenInters ← author(prop)
        end if
    end for
    if ((bestProposal = null) ∨ (bestEval > bestScore)) then
        return null
    end if
    if (receivers \ chosenInters = ∅) then
        return partialJP ∪ bestProposal
    end if
    leftProposal ← ChooseSet(partialJP ∪ bestProposal, ∅, receivers \
chosenInters, bestScore)
    newScore ← min(eval(leftProposal), bestScore)
    newProposals ← proposals \ bestProposal
    rightProposal ← null
    if newProposals ≠ ∅ then
        rightProposal ← ChooseSet(partialJP, proposals \
bestProposal, receivers, newScore)
    end if
    if eval(leftProposal) ≤ eval(rightProposal) then
        return leftProposal
    else
        return rightProposal
    end if
end function
```

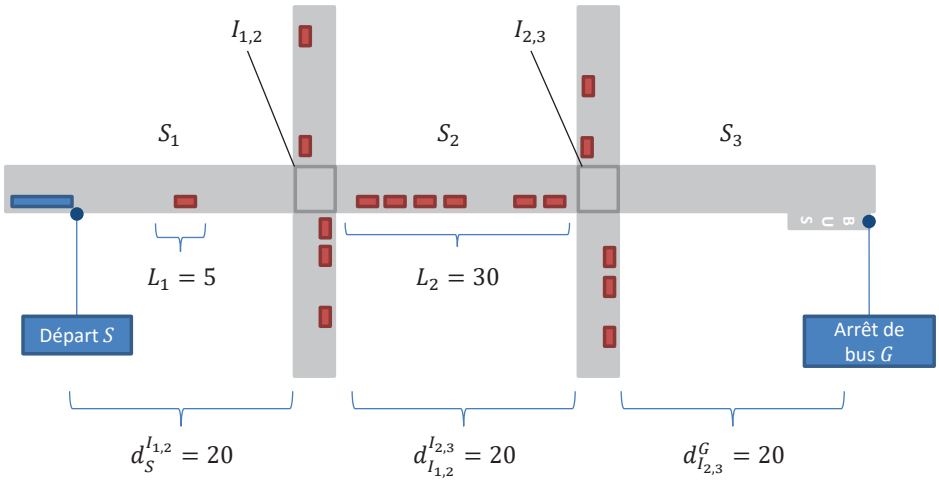


Figure 3.5: Scénario à deux intersections. Les véhicules sont uniquement représentés à titre illustratif.

- π_{12} est décrite ainsi : $t_{in}^{1,2}(b) = 40$, $MaxOut_{t_0}^{t_0+40}(S_1) = 25$, $MaxAdd_{1,2} = 10$, $MaxPass_{1,2} = 5$
- π_{21} est décrite ainsi : $t_{in}^{2,3}(b) = 40$, $MaxOut_{t_0}^{t_0+40}(S_2) = 40$, $MaxAdd_{2,3} = 0$, $MaxPass_{2,3} = 0$
- π_{22} est décrite ainsi : $t_{in}^{2,3}(b) = 60$, $MaxOut_{t_0}^{t_0+60}(S_2) = 50$, $MaxAdd_{2,3} = 0$, $MaxPass_{2,3} = 10$

Le bus b évalue les 4 politiques jointes partielles $\{\pi_{11}, null\}$, $\{\pi_{12}, null\}$, $\{null, \pi_{21}\}$, $\{null, \pi_{22}\}$ et choisit $\{null, \pi_{21}\}$ qui obtient le meilleur score d'après la fonction d'évaluation utilisée. Il choisit donc cette proposition et demande aux intersections restantes, ici $I_{1,2}$, de faire des propositions adaptées à la politique jointe partielle en envoyant $Request(I_{1,2}, 90, \{null, \pi_{21}\})$. $I_{1,2}$ doit donc faire des propositions compatibles avec π_{21}. π_{21} impose $t_{in}^{1,2} = 20$, et les charges initiales de S_1 et S_2 imposent $Add_{1,2} + Pass_{1,2} \leq 5$. $I_{1,2}$ ne réussit pas à générer de solution vraisemblablement réalisable, c'est-à-dire respectant ces contraintes. Elle envoie donc $Propose(\emptyset, t_G, \{null, \pi_{21}\})$ pour signaler l'absence de politiques compatibles.

En recevant ce message, le bus b déduit qu'il n'existe pas de solutions possibles incluant $\{null, \pi_{21}\}$. Il reprend donc sa recherche sur la base des propositions soumises précédemment à l'exception de celle-ci, à savoir : $\{\pi_{11}, null\}$, $\{\pi_{12}, null\}$, $\{null, \pi_{22}\}$. $\{\pi_{11}, null\}$ obtient le meilleur score par la fonction d'évaluation. Le bus b choisit donc cette proposition et demande aux intersections restantes, ici $I_{2,3}$, de suggérer des propositions adaptées à la politique jointe partielle en envoyant le message $Request(I_{2,3}, 90, \{\pi_{11}, null\})$.

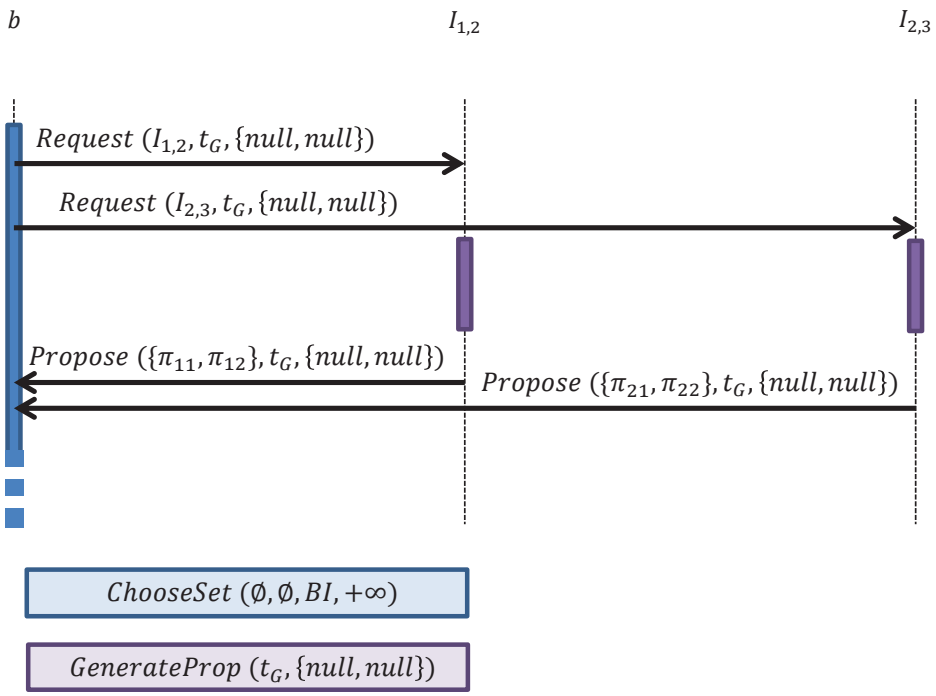


Figure 3.6: Le bus fait une requête auprès des intersections, puis chaque intersection fait deux propositions.

$I_{2,3}$ repropose donc π_{22} qui est compatible avec la politique jointe partielle et suggère une nouvelle politique π_{23}. π_{23} est décrite ainsi : $t_{in}^{2,3}(b) = 50$, $MaxOut_{t_0}^{t_0+50}(S_2) = 45$, $MaxAdd_{2,3} = 0$, $MaxPass_{2,3} = 5$. Pour effectuer ces propositions, $I_{2,3}$ envoie $Propose(\{\pi_{22}, \pi_{23}\}, t_G, \{\pi_{11}, null\})$.

Le bus b évalue $\{\pi_{11}, \pi_{22}\}$ et $\{\pi_{11}, \pi_{23}\}$ et choisit $\{\pi_{11}, \pi_{22}\}$. Cette politique jointe n'étant pas partielle, la recherche peut s'arrêter. Le bus b envoie donc le message $Accepted(\{\pi_{11}, \pi_{22}\}$ pour indiquer que c'est cette politique jointe qui doit être appliquée par les intersections.

Complexité communicationnelle

Soit C le nombre d'appels de $Request$ et $Propose$ réalisés. Soient n le nombre d'intersections backbone et k le nombre maximal de propositions données par une intersection avec $wait$.

Le noeud racine et ses $n*k-1$ fils droits (plus précisément : son fils droit, le fils droit de son fils droit... et ainsi de suite $n*k_1$ fois) forment une structure en peigne qu'on appelle la première couche de l'arbre. Les $n*k$ noeuds dans cette structure en peigne

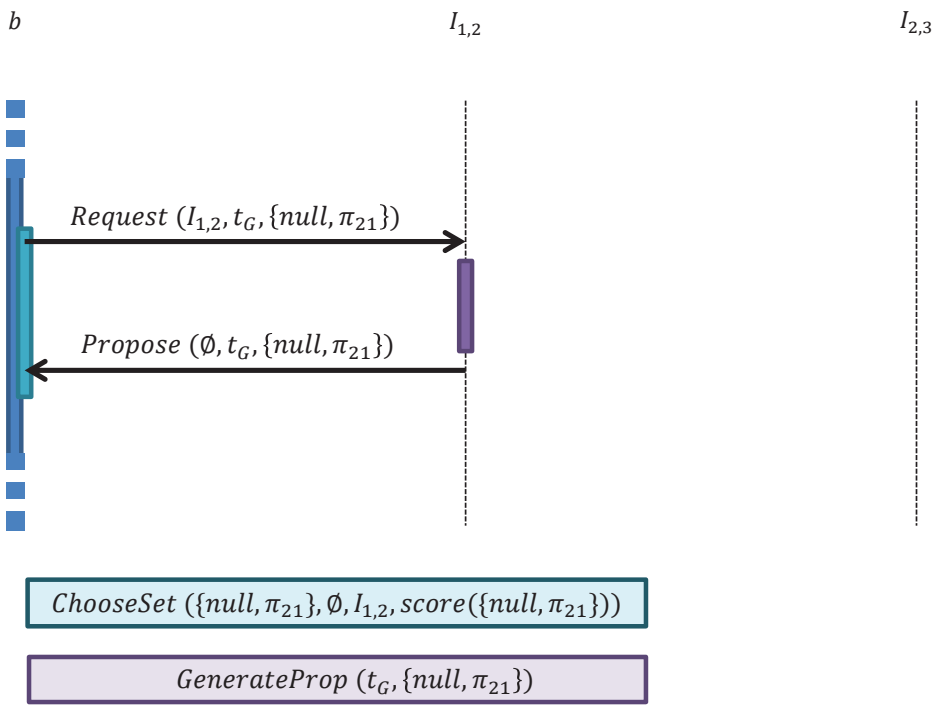


Figure 3.7: Après avoir choisi π_{21}, le bus effectue une requête auprès de $I_{1,2}$, qui n'a aucune proposition à faire pour cette demande. Le bus abandonne donc π_{21}.

n'engendrent qu'un appel à $Request$ et un à $Propose$ (dans le noeud racine). Chaque noeud de cette première couche a un fils gauche qui est la racine d'une structure en peigne similaire. Pour chacune de ces $n * k$ structures, on obtient aussi un appel à $Request$ et $Propose$. Les noeuds de ces structures sont appelés la deuxième couche. La structure en peigne ayant pour racine un noeud de cette deuxième couche n'a pas $n * k$ noeuds mais $(n-1) * k$ noeuds car il y a une intersection de moins dans la liste des destinataires. Récursivement, nous pouvons montrer qu'il y a n couches. Dans une couche, il y a un appel pour chaque noeud racine dans cette couche, c'est-à-dire pour chaque noeud de la couche précédente.

$$C = 1 + (k * n) + (k * n) * (k * (n-1)) + (k * n) * (k * (n-1)) * (k * (n-2)).. + ... + (k * n) * (k * (n-1)) * (k * (n-2)) * (...) * (k * 1)$$

$$C = 1 + k^1 * \frac{n!}{(n-1)!} + k^2 * \frac{n!}{(n-2)!} + ... + k^n * \frac{n!}{1}$$

$$C \leq n * k^n * n!$$

$$C \leq k^n * n^{(n+1)}$$

$$C \leq (k * n)^{(n+1)}$$

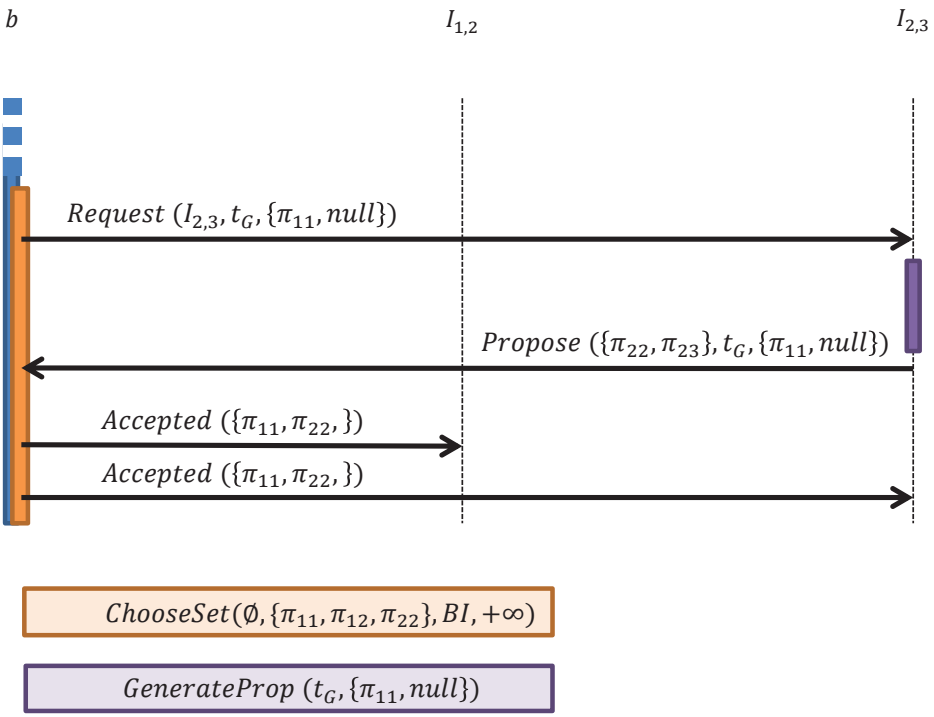


Figure 3.8: Après avoir choisi π_{11}, le bus fait une requête auprès de $I_{2,3}$, qui soumet deux propositions. Le bus accepte π_{22}, il a donc choisi sa politique jointe et signale son acceptation aux deux intersections.

Avec $k = 10$ et $n = 10$, $C \simeq 4 * 10^{16}$

3.3.6 Stratégie ascendante

Une autre approche possible est une approche ascendante, ou "bottom-up". En effet dans le mécanisme décrit précédemment le bus est seul responsable de la décision finale. Cependant cette approche ne permet pas d'aborder toute la complexité du problème, puisque par exemple elle ne permet pas de gérer une situation où les intersections doivent satisfaire les requêtes simultanées de plusieurs bus. Nous allons maintenant présenter une seconde approche dite "bottom-up", c'est-à-dire une approche où les intersections reçoivent les requêtes d'un (ou plusieurs) bus, et se coordonnent elles-mêmes afin de produire une politique jointe permettant de satisfaire la (ou les) requête(s) reçue(s).

Dans cette approche, la recherche de solutions est réalisée de manière arborescente par des intersections ou des groupes d'intersections appelés "coalitions". L'arbre est constitué de politiques jointes partielles, chaque noeud étant une politique jointe partielle constituée de la politique jointe partielle contenue dans son noeud "père"

et d'une proposition concernant une intersection non concernée par la politique jointe partielle "père". À chaque instant, un agent ou une coalition peut se proposer d'enrichir une proposition de politique jointe partielle qui est une feuille de l'arbre avec une proposition reprenant celle-ci et y ajoutant une politique pour chaque intersection appartenant à la coalition.

Exemple Un bus doit franchir 4 intersections I_1, I_2, I_3, I_4. Les intersections I_1 et I_3 forment une coalition C_{13}, alors que I_2 et I_4 restent isolées. La construction de la politique jointe peut se faire dans n'importe quel ordre à partir du noeud racine, par exemple I_4, puis C_{13}, puis enfin I_2. La politique jointe (partielle) constituant la feuille de l'arbre à chaque étape du processus serait donc successivement de la forme $\Pi = \{?, ?, ?, \pi_4\}$, puis $\Pi = \{\pi_1, ?, \pi_3, \pi_4\}$, puis enfin $\Pi = \{\pi_1, \pi_2, \pi_3, \pi_4\}$.

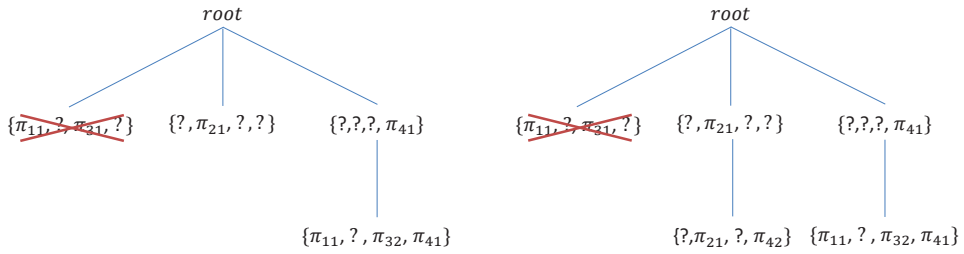


Figure 3.9: Exemple d'arbre de recherche. L'intersection I_4 peut ajouter une proposition pour enrichir le noeud $\{?, \pi_{21}, ?, ?\}$ car celui-ci ne contient aucune politique pour I_4 pour l'instant et n'a pas été refusé. Il propose la politique π_{42} , et ajoute donc un noeud fils $\{?, \pi_{21}, ?, \pi_{42}\}$

La constitution de ces coalitions est utilisée comme une heuristique permettant de limiter l'espace de recherche lorsqu'il y a un certain nombre d'interdépendances fortes entre les politiques de plusieurs intersections. Par exemple, lorsqu'une intersection I_y est très saturée et qu'une intersection I_x en amont de I_y est susceptible de laisser passer un grand nombre de véhicules, alors il semble efficace de coordonner ces deux intersections en priorité afin que celles-ci proposent un ensemble de politiques jointes restreint qui réduisent rapidement l'espace de recherche de solutions globales. Ainsi une coalition capable de produire peu de solutions est une "bonne" coalition puisqu'elle permet de limiter le nombre de branches à explorer dans l'arbre de recherche. Elle sera donc privilégiée.

Ce mécanisme fonctionne donc en deux étapes. Durant la première, les intersections dialoguent afin de former des coalitions les plus efficaces possibles, c'est-à-dire des coalitions capables de générer peu de propositions en proportion du nombre d'agents la constituant. Une fois ces coalitions formées, chaque coalition ou agent isolé agit

comme un agent capable de produire des propositions en ajoutant des noeuds à l'arbre des solutions ou en écartant certains d'entre eux.

Formation des coalitions

Afin de déterminer avec quelle autre intersection il peut être intéressant de former des coalitions, chaque intersection $I_x \in BI$ constitue, de la même manière que dans l'approche descendante proposée précédemment, un ensemble de propositions P_x qui se veut représentatif des solutions possibles. Ces propositions sont envoyées à toutes les autres intersections de BI. Chaque intersection va chercher à trouver les partenaires de coalition potentiels en identifiant ceux proposant le maximum de propositions incompatibles avec les siennes. Pour sélectionner les intersections avec qui cette évaluation s'opère, le problème posé à chaque intersection peut être modélisé comme un problème de bandit à n bras.

Le problème des bandits à n bras est un problème classique dans lequel un joueur se retrouve face à n machines à sous ayant des espérances de gain différentes mais inconnues. Le joueur cherche à maximiser son gain à long terme sachant qu'il ne peut jouer que sur une machine à la fois. Une partie a deux utilités : obtenir un gain par cette partie et extraire une information supplémentaire sur la machine, ce qui permet d'affiner l'estimation de l'espérance de gain de celle-ci. La difficulté de ce problème réside dans la manière de trouver un compromis entre l'exploration, qui consiste à jouer sur des machines qu'on connaît peu afin d'affiner notre estimation de leur espérance, et l'exploitation c'est-à-dire la maximisation du gain en jouant sur les machines qui semblent les meilleures dans l'état actuel de nos connaissances.

Ce problème ressemble à celui rencontré par les intersections dans la situation décrite précédemment. Pour une intersection I_x, chaque autre intersection $I_k \in BI \setminus I_x$ peut être vue comme une machine. Estimer la faisabilité de la politique jointe contenant π_x^i et π_k^j, avec $\pi_x \in P_x$ et $\pi_k \in P_k$, peut être vu comme l'action de jouer une partie sur une machine qui va consommer une ressource (du temps) et obtenir un gain : 1 si les propositions sont incompatibles, 0 sinon. Plusieurs stratégies existent pour aborder ce type de problème [63]. Dans notre cas, nous utilisons comme stratégie l'exploration de Boltzmann, ou SoftMax. Dans cette méthode, la probabilité de choisir chaque levier $k \in L$ est la suivante :
$p(k) = \frac{\epsilon^{\hat{\mu}_k/\tau}}{\sum_{i \in L} \epsilon^{\hat{\mu}_i/\tau}}$, où $\hat{\mu}_k$ est l'espérence estimée de I_k et τ est un paramètre appelé température. Plus ce paramètre est élevé, plus la probabilité d'exploration est grande. On peut utiliser la méthode nommée "decreasing Softmax" dans laquelle cette température diminue après chaque itération.

Le but de cette méthode étant de ne pas explorer l'intégralité des combinaisons possibles, on arrêtera après $nbTest$ essais. Par cette méthode, chaque intersection identifie les intersections ayant potentiellement le plus petit espace de recherche joint. Un couple d'intersections atteignant un seuil $minFail$ de combinaisons incompatibles sera alors susceptible de former une coalition. Chaque intersection a une préférence sur son partenaire de coalition préféré. Si une préférence est bidirectionnelle alors les deux agents forment une coalition. Chaque agent de la coalition peut alors faire une nouvelle série de propositions individuelles à son partenaire, choisies à partir des contraintes de celui-ci. La coalition construit alors un ensemble de propositions de politiques jointes partielles en combinant ces propositions individuelles. Puis la coalition se comporte de la même manière qu'un agent en faisant des propositions, à la différence que les propositions faites par la coalition ne sont pas des politiques individuelles mais des politiques jointes partielles.

Construction de la solution

Une fois les coalitions formées, celles-ci génèrent des propositions de politiques jointes partielles. Ces coalitions sont considérées comme des agents au même titre que les intersections hors coalition. Ces agents forment l'ensemble Ag.

Cette méthode reprend en partie les actes utilisées dans la méthode précédente : $Acts = \{Request, Propose, Accept, Refuse\}$. Cependant ceux-ci n'ont pas exactement la même signification.

Request(BI, t_G) : avec cet acte, le bus demande à toutes les intersections BI à qui ce message est envoyé de se coordonner afin de proposer une politique jointe de manière à ce qu'il atteigne son prochain arrêt avant la date t_G. Le bus ne fait qu'une requête à l'ensemble des intersections.

Propose$(node, partialJP)$: avec cet acte, un agent enrichit l'état du noeud $node$ en lui ajoutant un fils représentant la politique jointe partielle $partialJP$. Ceci représente l'enrichissement de la proposition de politique jointe partielle. Cet acte n'est réalisable que si le noeud père $node$ n'a pas été refusé (voir ci-après).

Accept$(node)$: avec cet acte, un agent signale que la proposition $node$ lui convient, c'est-à-dire que la politique jointe partielle portée par le noeud $node$ est réalisable. Chaque agent envoie exactement une fois, à propos de chaque feuille, le message $Accept(node)$ ou $Refuse(node)$. Si ce message est reçu pour un noeud portant sur une solution jointe complète en quantité égale au nombre d'intersections, alors la solution portée par ce noeud est adoptée.

Refuse$(node)$: avec cet acte, un agent signale que la proposition $node$ ne lui convient pas, c'est-à-dire que la politique jointe partielle portée par le noeud $node$ n'est pas réalisable. Chaque agent envoie exactement une fois, à propos de chaque feuille, le message $Accept(node)$ ou $Refuse(node)$. Si ce message est reçu au moins une fois au sujet d'un noeud, celui-ci n'est pas réalisable.

La stratégie utilisée par les agents est un parcours en profondeur de l'arbre, décrit dans l'algorithme 6. $getLeaves(root)$ retourne l'ensemble des feuilles de l'arbre de racine $root$ ainsi que des noeuds dont tous les fils ont été refusés au moins une fois. $AcceptedAndRefusedLeaves(A_i)$ retourne l'ensemble des noeuds qui ont déjà été acceptés ou refusés par A_i. $depth(l)$ retourne la profondeur du noeud l dans l'arbre. $containsProp(l, A_i)$ retourne vrai si et seulement si le noeud l ou un de ses ascendants est une proposition de A_i. $compatible(\pi_j, pjp)$ retourne vrai si et seulement si la politique π_j est compatible avec la politique jointe partielle pjp.

Soit P_i l'ensemble des propositions de l'agent A_i triées par coût croissant. À chaque étape, l'agent A_i considère une feuille l de l'arbre à laquelle il n'a pas participé, c'est-à-dire pour laquelle $containsProp(l, A_i)$ est faux, et qu'il n'a pas déjà accepté ou refusé, en commençant par les feuilles ayant les profondeurs les plus grandes, et fait une proposition pour celle-ci en commençant par la proposition de coût minimal dans P_i. Si aucune proposition de P_i n'est compatible avec Π_x alors A_i envoie un refus pour cette feuille. Les feuilles pour lesquelles aucune proposition n'est compatible sont refusées. Dans cette première version, les feuilles ayant la profondeur maximale sont acceptées.

On note que cet algorithme est susceptible d'accepter plusieurs politiques jointes, on considère que la première politique jointe acceptée par toutes les intersections est choisie. Ce mécanisme pourrait être enrichi de manière à augmenter le contrôle des intersections sur l'issue du mécanisme, par exemple en n'acceptant pas automatiquement les feuilles de profondeur maximum, et en utilisant un mécanisme de négociation entre les intersections permettant de choisir une solution parmi l'ensemble des noeuds représentant des solutions potentielles, c'est-à-dire l'ensemble des noeuds de profondeur maximale.

Exemple Un bus b traverse 3 intersections, dont 2 ont formé une coalition. On a donc 2 agents A_1 et A_2, ayant chacun des propositions individuelles ordonnées par préférence décroissante, respectivement $\pi_{11} \succeq \pi_{12} \succeq \pi_{13}$ et $\pi_{21} \succeq \pi_{22}$.

Initialement l'arbre de solutions est vide. A_1 initialise le processus, et récupère toutes les feuilles de l'arbre, pour l'instant il n'y a que le noeud racine. A_1 propose donc π_{11}, qui est sa proposition préférée et qui est compatible avec le noeud $root$.

Algorithme 6 Algorithme de proposition des agents

```
function ChooseProp(root, A_i)
    leaves ← getLeaves(root) \ AcceptedAndRefusedLeaves(A_i)
    for (depth ← |Ag|; depth ≥ 0; depth ← depth − 1) do
        for (l ∈ leaves) do
            if depth(l) = depth then
                if (¬containsProp(l, A_i)) then
                    pjp ← l.politic
                    for (π_j ∈ P_i) do
                        if compatible(π_j, pjp) then
                            Propose(l, pjp ∪ π_j)
                            return
                        end if
                    end for
                    Refuse(l)
                    leaves ← leaves ∪ father(l)
                else
                    if depth = |Ag| then
                        Accept(l)
                        return
                    end if
                end if
            end if
        end for
    end for
    return FAIL
end function
```

Figure 3.10: État de l'arbre de recherche

A_2 a maintenant la main. Il récupère les noeuds feuilles, ici uniquement $root \rightarrow \pi_{11}$. A_2 vérifie si certaines de ses propositions sont compatibles avec $root \rightarrow \pi_{11}$. Ni π_{21} ni π_{22} ne sont compatibles avec π_{11}, le noeud $root \rightarrow \pi_{11}$ est donc refusé par A_2. Le noeud père de $root \rightarrow \pi_{11}$, c'est-à-dire $root$, est ajouté à la liste des noeuds à explorer. Ce noeud est ainsi exploré. A_2 propose donc sa proposition préférée compatible avec $root$, c'est-à-dire π_{21}.

A_1 a de nouveau la main. Il récupère les noeuds feuilles non refusés, ici uniquement $root \rightarrow \pi_{21}$. A_1 vérifie si certaines de ses propositions sont compatibles avec $root \rightarrow \pi_{21}$. Ni π_{11}, ni π_{12}, ni π_{13} ne le sont, le noeud $root \rightarrow \pi_{21}$ est ainsi refusé par A_1. Le

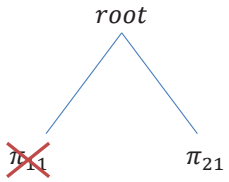


Figure 3.11: État suivant de l'arbre de recherche

noeud père de $root \rightarrow \pi_{11}$, c'est-à-dire $root$, est ajouté à la liste des noeuds à explorer. Ce noeud est donc exploré. A_1 propose donc sa proposition préférée compatible avec $root$, c'est-à-dire π_{11}. Cependant celle-ci a déjà été refusée, A_2 propose donc sa deuxième proposition préférée, π_{12}.

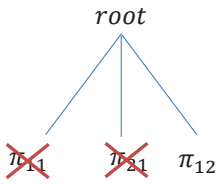


Figure 3.12: État suivant de l'arbre de recherche

A_2 a de nouveau la main. Il récupère les noeuds feuilles non refusés, ici uniquement $root \rightarrow \pi_{12}$. A_2 vérifie si certaines de ses propositions sont compatibles avec $root \rightarrow \pi_{12}$. π_{21} est incompatible avec π_{12}, mais π_{22} est compatible. A_2 propose donc π_{22}.

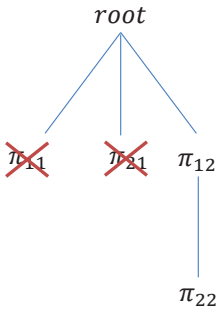


Figure 3.13: État suivant de l'arbre de recherche

A_1 a la main, et accepte le noeud $root \rightarrow \pi_{12} \rightarrow \pi_{22}$ puisqu'il a la profondeur maximale et n'est pas refusé. A_2 obtient la main et accepte également ce noeud.

3.3.7 Perspectives

Les travaux présentés dans ce chapitre sont susceptibles de faire l'objet de recherches supplémentaires, et nous présentons ici quelques perspectives possibles.

Seconde stratégie ascendante

On pourrait envisager une seconde stratégie ascendante, dans laquelle les agents prendraient en compte le nombre d'échecs relatif à chaque noeud en réalisant le parcours de l'arbre, avec une approche similaire à celle utilisée dans le cadre du problème des bandits à n bras : les feuilles ayant le moins d'échecs ont plus de chances d'être sélectionnées pour proposer un fils.

Dans cette stratégie comme dans la précédente, l'agent courant examine des noeuds et fait des propositions concernant ceux-ci lorsque c'est possible, les refuse si aucune proposition n'est possible, et accepte les politiques jointes qui conviennent. La principale différence avec la stratégie précédente est l'ordre dans lequel les noeuds sont choisis. L'agent traite en priorité les noeuds ayant essuyé le moins de refus. En partant de la racine, il sélectionne à chaque étape un fils ayant de bons résultats, ou crée un nouveau noeud fils.

Cette méthode se rapproche par certains aspects de la méthode Monte Carlo Tree Search ou MCTS [14], qui est une heuristique utilisée dans les processus de décision, principalement dans les jeux, pour déterminer l'espérance d'une succession de choix. À la différence de notre problème, il est possible dans MCTS de connaître les coups jouables par les autres joueurs, alors qu'une intersection ne connaît pas les propositions des autres. MCTS fonctionne en 4 étapes :

- la sélection : à partir du noeud racine, on sélectionne le noeud fils le plus urgent à explorer, puis on recommence plusieurs fois jusqu'à atteindre un noeud feuille qui n'est pas terminal et peut être étendu, c'est-à-dire qu'il peut avoir un nouveau fils.

- l'expansion : on ajoute un ou plusieurs noeuds fils au noeud sélectionné lors de la sélection.

- la simulation : à partir des noeuds ajoutés lors de l'expansion, on simule un certain nombre de suites de parties aléatoires qui sont ajoutées à l'arbre. Si des fins de parties sont atteintes, l'état des noeuds ajoutés est mis à jour pour prendre en compte le succès ou l'échec de la partie.

- la rétropropagation : mise-à-jour des informations de chaque noeud pour prendre en compte le résultat de la simulation : sur tous les noeuds concernés, on met à jour le nombre de fois où ce noeud a été choisi et le nombre de fois où ce noeud a obtenu une issue gagnante.

Cette méthode ne s'applique pas à notre problème pour plusieurs raisons, notamment à cause de l'étape de simulation qui n'est pas possible dans le cadre de notre problème. En effet, chaque agent participe exactement une fois à une politique jointe et ne peut donc pas ajouter plus d'un niveau de noeuds fils. De plus, le fonctionnement de la rétropropagation ne peut pas être celui utilisé classiquement. Par ailleurs nous nous plaçons dans le cadre d'une recherche de satisfaction du problème, c'est-à-dire que notre recherche peut s'arrêter lorsqu'on trouve une solution satisfaisant la demande du bus.

On s'intéressera en revanche aux méthodes de sélection utilisées dans MCTS. La plus couramment utilisée est Upper Confidence Bounds for Trees (UCT), qui s'appuie sur la méthode Upper Confidence Bounds (UCB) utilisée dans les problèmes de bandits multi-bras. UCB consiste à choisir à chaque itération le bras maximisant : $UCB1 = \overline{X}_j + \sqrt{\frac{2\ln n}{n_j}}$, où $\overline{X}_j$ est la récompense moyenne d'un bras j, n_j est le nombre de fois où le bras j a été joué et n le nombre de coups joués. De la même manière, la méthode UCT maximise à chaque noeud : $UCT = \overline{X}_j + 2C_p\sqrt{\frac{2\ln n}{n_j}}$, où n est le nombre de fois où le noeud courant a été visité, n_j est le nombre de fois où le noeud fils j a été visité, et $C_p > 0$ est une constante. $\overline{X}_j$ réalise l'exploitation (visiter les noeuds qui ont prouvé qu'ils obtenaient des succès) et $2C_p\sqrt{\frac{2\ln n}{n_j}}$ réalise l'exploration (visiter des noeuds moins visités, sur lesquels on a le moins d'informations).

$\overline{X}_j \in [0;1]$ représentant la récompense moyenne d'un noeud, celui-ci vaut 0 tant qu'aucun résultat n'aura été obtenu. Notre but étant de trouver une solution, cette méthode n'est pas appropriée car elle ne réalise que l'exploration. En revanche on peut adapter cette formule aux caractéristiques de notre problème, par exemple en considérant qu'obtenir un noeud valide de niveau $|Ag|$ est un succès, et qu'obtenir un noeud de niveau élevé inférieur à $|Ag|$ est un résultat positif qui rend l'exploration de ce noeud potentiellement plus pertinente que celle d'un noeud de plus bas niveau.

Adaptation dynamique au trafic

Le mécanisme que nous avons présenté ici, quelle que soit la version, est entièrement anticipatif et ne permet pas de s'ajuster à la progression dynamique des bus. Cependant le réseau routier est hautement dynamique, et il se peut que l'anticipa-

tion réalisée ne soit pas correcte. Une perspective serait d'étudier la manière dont ce système pourrait être adapté à une utilisation dynamique. Une manière naïve consisterait à réeffectuer cette procédure régulièrement afin d'ajuster fréquemment la politique jointe des intersections. Cependant il est surement possible d'envisager une autre manière de gérer cette dynamicité qui permettrait de réutiliser les étapes de négociation des itérations précédentes, qui serait ainsi plus efficace.

Par ailleurs, on peut imaginer qu'une intersection révise la faisabilité de ses propositions. En cas de révision à la hausse de la qualité de ses propositions (le bus franchit l'intersection plus tôt que prévu), aucune dégradation de la solution n'est possible. En revanche en cas de révision à la baisse de la qualité (le bus franchit l'intersection plus tard que prévu) un ajustement est nécessaire. Idéalement, la politique jointe décidée précédemment aura pris en compte ce risque et pris suffisamment d'avance dans les dates de passage à chaque intersection pour que le bus puisse trouver une nouvelle solution en relançant le mécanisme.

Véhicules d'urgence

Une des perspectives possibles pour ces travaux consiste à étendre ceux-ci aux véhicules d'urgence. En effet, le mécanisme actuel laisse l'application de chaque politique d'intersection à l'intersection elle-même, et donc à la négociation entre les véhicules tel que décrit dans le chapitre précédent. Cependant on peut supposer que l'agent intersection puisse se rendre plus autoritaire afin de garantir de bonnes conditions de circulation pour les véhicules d'urgence. Ceci nécessiterait cependant d'examiner les particularités de ce type de véhicules. De plus, le fait que les véhicules prennent part à la décision par la négociation permet de limiter les risques d'interblocage entre les véhicules. Dans le cas où les intersections effectueraient une régulation de manière totalement autoritaire, il faudrait garantir que ceux-ci n'aient pas lieu, en particulier afin de ne pas gêner la progression du véhicule d'urgence.

3.4 Conclusion

Nous avons présenté dans ce chapitre un mécanisme permettant à un bus de se coordonner avec les intersections de sa trajectoire en prenant en compte le trafic environnant, afin de laisser le chemin du bus dégagé et ainsi de lui permettre d'atteindre sa destination à la date prévue. Nous détaillons dans le chapitre suivant les implémentations faites des mécanismes présentés dans ce chapitre et dans le chapitre précédent, et présentons les résultats expérimentaux obtenus.

4 Implémentations et résultats

Dans ce chapitre nous détaillons l'implémentation réalisée pour chacun des deux mécanismes présentés dans les chapitres précédents, et présentons les résultats obtenus pour chacun de ces mécanismes.

4.1 Négociation du droit de passage à l'intersection

Afin d'évaluer le modèle de régulation proposé dans le chapitre 2 nous avons implémenté ce modèle en langage Java. La Figure 4.1 présente une visualisation de la version la plus récente du simulateur. Cette section présente les choix d'implémentation réalisés et les résultats obtenus.

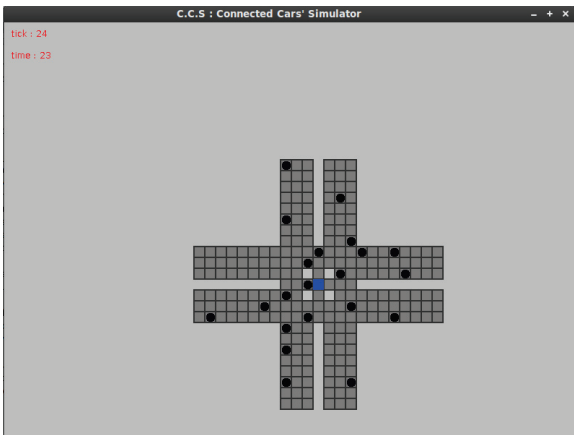


Figure 4.1: Vue actuelle du simulateur

4.1.1 L'environnement

L'environnement de simulation est modulable afin de générer différents scenarii. Celui-ci est composé d'une ou plusieurs infrastructures (sections de route ou intersections) liées entre elles, les agents pouvant se déplacer au sein d'une infrastructure et passer d'une infrastructure à une autre.

Description de l'environnement

L'environnement du système multi-agents est formé d'un ensemble d'infrastructures liées entre elles. Une infrastructure peut être une intersection ou une route. Ces infrastructures sont structurées en une grille. Cependant dans les scenarii présentés dans cette section, seule une intersection est considérée.

Chaque intersection est décrite à deux niveaux, macroscopique et microscopique. Au niveau macroscopique, l'intersection est décrite par sa position dans la grille, le nombre de ses voies et leurs caractéristiques, et par son type de croisement. Le type de croisement d'une intersection détermine si cette intersection est traversée par les véhicules "à l'Indonésienne" (les trajectoires de deux véhicules face-à-face tournant à gauche ne se coupent pas) ou non (les trajectoires de deux véhicules face-à-face tournant à gauche se coupent et forcent ces véhicules à se contourner). Le type de croisement d'une intersection définit à la fois sa structure ainsi que les trajectoires empruntées par les véhicules. Les points de conflit entre les différentes trajectoires traversant l'intersection sont déterminés par le type de croisement de l'intersection qu'ils traversent ("à l'Indonésienne" ou non).

Une intersection possède trois ou quatre points d'entrée/sortie, chacun de ces points étant défini par :

- Sa position par rapport à l'intersection (Nord, Sud, Ouest ou Est).
- Son nombre de voies d'entrée $\in [0..N]$.
- Son nombre de voies de sortie $\in [0..N]$.
- La longueur de ses voies d'entrées et de sorties (en nombre de cellules).

Nous avons utilisé le langage XML pour la description d'un réseau d'intersections. Le langage XML est un langage de balisage qui permet la description de données complexes, ainsi que des liens entre les données décrites.

Chaque intersection de la grille constituant l'environnement est décrite par :

- Ses coordonnées x et y sur la grille.
- Son type (croisement à l'Indonésienne ou non).
- La description de chacun de ses points d'entrée/sortie.

Un point d'entrée/sortie d'une intersection se décrit par sa position par rapport au centre de l'intersection (Nord, Est, Sud ou Ouest), ainsi que par la description de son flux d'entrée et son flux de sortie. Un flux d'entrée ou de sortie est décrit par son type (in ou out), sa taille et son nombre de voies.

Génération de l'environnement

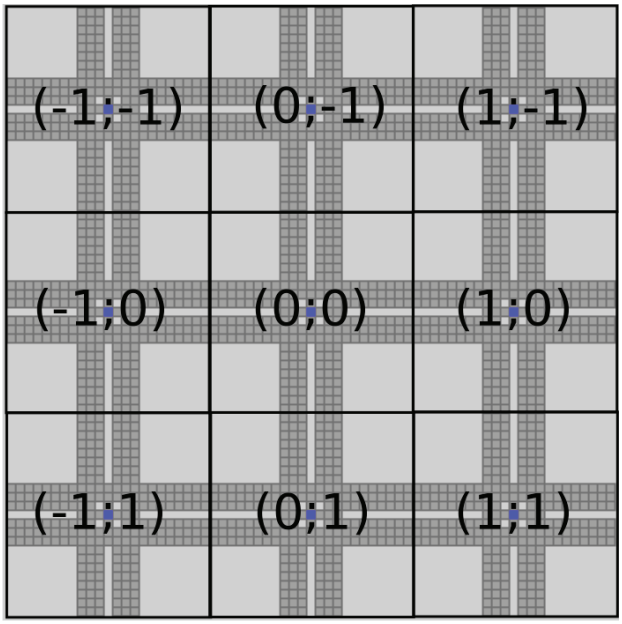


Figure 4.2: L'environnement est défini par deux matrices. La première, au niveau macroscopique, est la matrice des intersections. La seconde, au niveau microscopique, est formée par les cellules des différentes voies et intersections.

Au niveau microscopique, une intersection est composée de cellules. La Figure 4.2 montre la relation entre les niveaux macroscopique et microscopique de l'environnement. Le document XML donné à l'initialisation de l'environnement nous permet de connaitre les caractéristiques d'une intersection : son nombre de voies, leurs longueurs. À partir de ces informations, nous pouvons générer les cellules composant l'intersection.

Pour générer l'intérieur de l'intersection, là où les véhicules se croiseront, nous générons la trajectoire que devront suivre les véhicules pour traverser l'intersection. Les trajectoires possibles pour traverser une intersection s'appuient sur le type de croisement de l'intersection.

Ainsi, les trajectoires possibles des véhicules pour traverser l'intersection sont générées cellule par cellule et permettent la génération de l'intersection en connaissant à l'avance les trajectoires disponibles pour les véhicules afin qu'ils puissent traverser

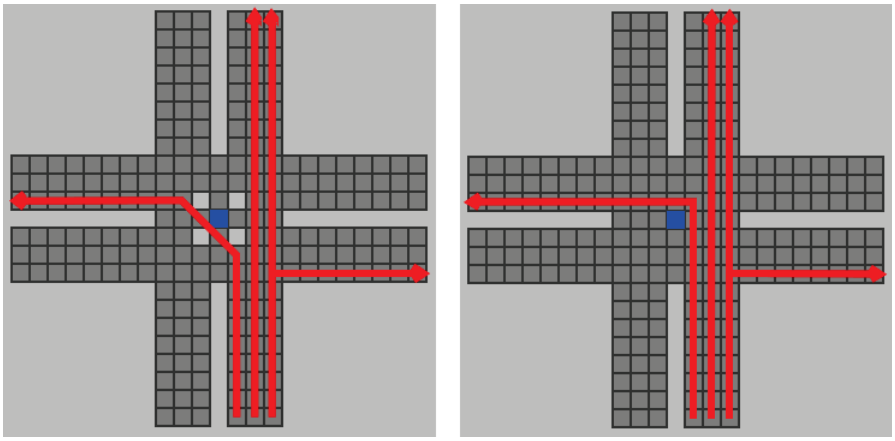


Figure 4.3: Trajectoires empruntables pour un véhicule approchant une intersection par le sud. Sur l'intersection de gauche le croisement se fait "à l'Indonésienne" contrairement à l'intersection de droite.

ladite intersection. La Figure 4.3 représente les différentes trajectoires possibles pour les véhicules voulant traverser une intersection à partir du sud de celle-ci.

4.1.2 Simulation d'un système multi-agents

Dans la section 4.1.1, nous avons vu comment l'environnement du système multi-agents est décrit et généré durant l'initialisation de la simulation. Cette section présente nos choix d'architecture pour le simulateur, son fonctionnement global et celui de ses différentes parties.

Nous avons choisi de développer notre propre simulateur plutôt que nous appuyer sur une plate-forme multi-agents existante afin que notre simulateur puisse s'adapter au mieux à nos besoins. L'architecture de notre simulateur doit permettre l'indépendance de la simulation de l'environnement et du raisonnement des agents. Dans les simulations présentées ici, les véhicules progressent cellule par cellule, cependant le fonctionnement du mécanisme de négociation proposé n'est pas dépendant de ce choix. Nous avons donc séparé le comportement physique des véhicules et leur raisonnement. L'architecture du programme est donc divisée en différentes couches, comme représenté en Figure 4.4.

La couche *Environment* implémente le moteur physique, c'est donc cette partie du simulateur qui implémente la gestion des déplacements des véhicules. La couche *Agents* est divisé en deux niveaux par agent, le *Corps* et le *Raisonnement*. Le *Corps* implémente les caractéristiques physiques de l'agent utilisé par la couche *Environnement* pour la gestion des déplacements. Le *Raisonnement* implémente le comportement de l'agent, il utilise le *Corps* pour récupérer les données perçues par l'agent, indiquer au *Corps* le comportement choisi (accélération, arrêt, etc.) et pour communiquer avec

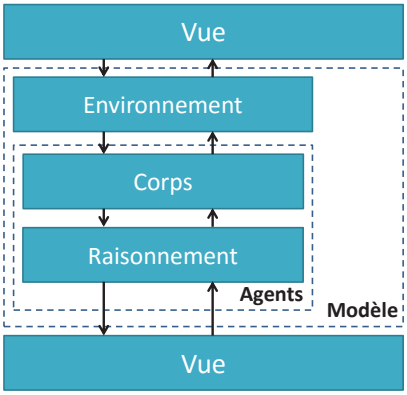


Figure 4.4: Architecture en couches du simulateur

d'autres agents. Les couches *Vue* permettent la récupération de données nécessaires à l'affichage de l'état actuel de la simulation et des agents, et servent à la récupération des données de simulation.

Simulation itérative

Une méthode d'implémentation possible pour un système multi-agents est d'en faire un programme multi-threads pour avoir des interactions en temps réel entre les différents agents du système. Cependant, cette première approche comporte un certain nombre de problèmes, parmi lesquels :

- Le partage du temps d'exécution entre les différents threads dépendant du nombre de coeurs sur le processeur. Pour que les threads fonctionnent réellement en même temps il faudrait au minimum un coeur par thread.

- Le temps d'exécution variable, difficile à mesurer et pas forcément significatif si les threads ne fonctionnent pas parfaitement en parallèle.

- Les conflits possibles lors de l'accès à une ressource par deux threads différents.

- Le besoin de verrous et de sémaphores pour éviter les conflits lors de l'accès, ce qui ralentit d'autant plus le processus puisque les threads en conflit attendent pour avoir accès à la ressource visée.

Pour ces raisons, nous avons décidé de développer notre simulateur de manière itérative. Cette approche est moins proche de la réalité que l'approche multi-thread, mais permet d'éviter les conflits d'accès à une variable et permet une mesure du temps d'exécution plus précise puisqu'elle est basée sur le nombre de pas de temps

(exemple : nombre de pas de temps nécessaires à n véhicules pour traverser une intersection). De plus, une approche itérative permet de suivre plus facilement le fonctionnement global du système. Le pseudo-code donné en algorithme 7 présente le fonctionnement général du simulateur. Les détails du fonctionnement de ce mécanisme sont donnés dans le chapitre 2.

Algorithme 7 Fonctionnement général du simulateur

```
initialisation de l'environnement
while fin de simulation non-atteinte do
    for all intersection I ∈ environnement do
        I transmet les données recues aux véhicules proches
        I.comportement()          /*I décide de son comportement dans la
négociation et l'applique*/
        for all véhicule v ∈ V_I do
            v.comportement()              /*les véhicules décident de leur
comportement dans la négociation et l'appliquent le cas échéant*/
        end for
    end for
    /*Une fois que les véhicules ont décidé s'ils se déplacent ou
non, l'environnement déplace les véhicules et vérifie l'absence de
collision*/
    environnement.déplacerVehicules()
    environnement.collisionManager()
    if flux continu de véhicules then
        genererVehicules()    /*Quand le scénario simule un flux continu
de véhicules, les véhicules sont rajoutés pour le prochain pas de
temps*/
    end if
end while
```

Agents

Comme expliqué ci-dessus, les agents sont séparés en deux parties. La première, le *Corps*, implémente les méthodes et les attributs permettant de représenter l'agent dans l'environnement et de le rendre accessible aux autres agents du système. Le *Corps* est, en quelque sorte, l'interface entre un agent et son environnement, ainsi qu'entre un agent et les autres agents du système.

La seconde, le *Raisonnement*, implémente le comportement de l'agent pour la prise de décision. Il traite les messages perçus par l'agent et décide des interactions de l'agent

avec les autres agents et de ses interactions avec l'environnement. Le *Raisonnement* diffère donc entre chaque type d'agents implémenté dans le système.

L'intérêt de diviser le fonctionnement d'un agent entre *Corps* et *Raisonnement* est de pouvoir modifier le fonctionnement de la physique du système et des protocoles d'interactions, gérés par le *Corps*, sans pour autant modifier le comportement de l'agent, géré par le *Raisonnement*, et inversement. Ces deux parties sont donc indépendantes.

Agent véhicule Le comportement de l'agent véhicule, décrit en pseudo-code dans l'Algorithme 8, peut se décomposer en plusieurs parties. Premièrement l'agent véhicule traite les messages reçus lors de la précédente itération, afin de mettre à jour ses données et de décider du comportement à adopter. Les messages reçus peuvent être de différents types :

- Message de "Bienvenue", envoyé par l'intersection lors de l'entrée du véhicule dans l'intersection lui indiquant la date à laquelle le véhicule devra s'engager dans l'intersection, appelée "réservation".

- Message de "Nouvelle Configuration", envoyé par l'intersection si une nouvelle configuration a été choisie par les véhicules. L'agent véhicule met donc à jour sa réservation si besoin est.

- Message de "Proposition de Configuration", envoyé par l'intersection, il contient les propositions de configuration des autres véhicules se trouvant dans l'intersection.

Une fois les messages traités, si l'agent véhicule se trouve dans la zone de négociation, celui-ci recherche d'abord une nouvelle configuration lui permettant d'améliorer son utilité par rapport à la configuration actuelle. La génération d'une nouvelle proposition de configuration se fait par la résolution d'un problème d'optimisation sous contraintes, par le biais d'un solveur. Si une nouvelle configuration est trouvée par le véhicule et satisfait celui-ci, alors il l'envoie à l'intersection pour qu'elle soit transmise aux autres véhicules se trouvant sur l'intersection.

Ensuite, si le véhicule se trouve toujours dans la zone de négociation, il émet ses éventuels arguments portant sur les configurations proposées par les véhicules (lui inclus) lors de l'itération précédente. Puis il recherche parmi les configurations proposées par les véhicules lors des itérations précédentes celle proposant une utilité maximale pour le véhicule et qui est meilleure que l'actuelle. Si une telle

configuration a été proposée alors le véhicule indique son acceptation pour la configuration à l'intersection.

Enfin, l'agent véhicule gère son déplacement. Il met à jour sa direction et sa vitesse pour se déplacer, ou plutôt pour indiquer à l'environnement, qui joue ici le rôle d'un moteur physique, qu'il souhaite se déplacer. Si le véhicule passe d'une intersection à une autre, alors il indique sa sortie à l'intersection courante et indique son entrée à la prochaine intersection.

Algorithme 8 Comportement de l'agent véhicule

```
for all nouveau message m reçu do
    traiter message m
end for
if le véhicule est dans la zone de négociation then
    proposer une nouvelle configuration si disponible
    for all configuration config reçue ou proposée depuis la dernière itération do
        émettre d'éventuels arguments portant sur config
    end for
    for all configuration config reçue ou proposée depuis 2 itérations do
        Accepter ou refuser config
    end for
end if
if la cellule actuelle est la cellule destination courante then
    changer la cellule destination courante
end if
if le véhicule entre dans une nouvelle intersection then
    Envoyer un message à l'ancienne intersection pour notifier la sortie du véhicule
    Envoyer un message à la nouvelle intersection pour notifier l'arrivée du véhicule
end if
if la cellule destination est libre then
    demander à l'environnement une mise-à-jour de la position
end if
```

Intersections Le comportement de l'agent intersection, décrit en pseudo-code dans l'Algorithme 9, peut se diviser en deux phases. Premièrement, tout comme l'agent véhicule (cf. 4.1.2), l'agent intersection traite les différents messages reçus lors de l'itération précédente. Les messages reçus par une intersection peuvent être de différents types :

- Message "d'entrée de véhicule", envoyé par un véhicule arrivant dans l'intersection. L'agent intersection génère une réservation initiale pour le véhicule

et la lui envoie. La génération d'une réservation initiale par l'intersection est réalisée par l'algorithme *First Come First Served*.

- Message de "sortie de véhicule", envoyé par un véhicule sortant de l'intersection. L'agent intersection supprime donc le véhicule de sa liste de véhicules la traversant.

- Message de "Proposition de configuration", envoyé par un véhicule proposant une nouvelle configuration. L'agent intersection transmet la proposition aux autres véhicules traversant l'intersection.

- Message "Accepter/Refuser", envoyé par un véhicule donnant son opinion sur une configuration proposée. Les votes sont traités dans la deuxième phase du processus de l'agent intersection.

La deuxième phase du processus de l'agent intersection consiste à déterminer si une configuration proposée par l'un des véhicules a atteint le seuil d'acceptation. Si c'est le cas, alors l'agent intersection transmet cette nouvelle configuration aux véhicules et met à jour sa représentation de la configuration.

Algorithme 9 Comportement de l'agent Intersection

for all nouveau message m reçu **do**
    traiter message m
end for
if une configuration a obtenu suffisamment d'acceptations **then**
    indiquer la nouvelle configuration aux différents véhicules traversant l'intersection
end if

Utilisation de l'environnement en tant que moteur physique

Comme introduit précédemment, l'environnement du système multi-agents prend le rôle de moteur physique de la simulation. Il implémente les règles de déplacement des différents agents, ou plutôt des *Corps*, que le moteur physique doit déplacer. Le processus du moteur physique se divise en deux phases.

Premièrement, le moteur physique parcourt les différents véhicules prenant corps dans l'environnement du simulateur. Si un véhicule a une vitesse et une direction non nulle alors le moteur physique déplace le véhicule dans la direction souhaitée.

La deuxième phase du moteur physique consiste à identifier si deux véhicules ne se trouvent pas sur la même cellule ou si des véhicules ne se bloquent pas mutuellement. Avant qu'un véhicule ne mette à jour sa position, celui-ci identifie si la cellule visée est libre ou sera libre au prochain pas de temps. En d'autres termes, un véhicule ne met pas à jour sa position si un autre véhicule se trouve sur la cellule visée. Si une collision a lieu, que ce soit dû à deux véhicules présents sur la même cellule ou bien à un blocage mutuel de plusieurs véhicules, un message d'erreur est envoyé par le moteur physique et la simulation est arrêtée. Cependant cette situation n'est pas susceptible d'arriver tant que le comportement des agents respecte les contraintes de sécurité décrites dans le chapitre 2.

4.1.3 Réservations

Dans la sous-section 4.1.2, nous avons vu que les agents véhicules et intersections communiquent entre eux afin de négocier les temps de passage pour traverser l'intersection. Ces négociations et les solutions proposées par les agents sont basées sur un solveur CSP (satisfaction de contrainte).

ChocoSolver

ChocoSolver est une librairie java open-source destinée à la programmation par contrainte. ChocoSolver permet une définition rapide et simple des différentes contraintes à appliquer et est facilement intégrable à un programme Java. Ses performances lui ont valu 5 médailles au MiniZinc challenge en 2013 et 2014. Chaque agent du système possède son propre solveur de CSP qu'il peut utiliser avec différentes règles selon le contexte.

First Come First Served

Dans la sous-section 4.1.2, nous avons vu qu'une intersection, lors de l'entrée d'un véhicule, détermine une réservation initiale pour que le véhicule arrivant puisse traverser l'intersection. L'agent intersection applique pour cela une politique *First Come First Served* (ou FCFS) en utilisant son solveur. À l'ajout d'un nouveau véhicule, on crée un nouveau CSP ayant pout contraintes les règles physiques décrites dans le chapitre 2, ainsi qu'une contrainte sur les dates de passages des véhicules ayant déjà une réservation. Le solveur produit la date de passage minimale pour le nouveau véhicule sous ces contraintes, qui devient la date réservée pour ce véhicule.

4.1.4 Scenarii et résultats

Nous présentons ici les simulations réalisées et comparons les résultats de différentes politiques. Les politiques IP utilisées dans ces simulations sont la politique EIP (Extended IP, voir chapitre 2) avec les paramètres $d_{neg} = 2$ et $min_v = 10$, ce qui signifie qu'une négociation n'est lancée que lorsqu'au moins 2 nouveaux véhicules sont dans la zone de négociation et qu'ils ont 10 pas de temps pour réaliser leur négociation, ainsi que la politique EIP avec les paramètres $d_{neg} = 5$ et $min_v = 0$. Ces politiques sont respectivement notées IP(2,10) et IP(5,0). Dans le cadre de la politique FCFS (utilisée seule ou dans le cadre de CP pour enrichir une configuration), lorsque plusieurs véhicules entrent simultanément dans la zone de négociation, l'ajout de ces véhicules à la configuration courante se fait dans l'ordre de l'identifiant des véhicules, déterminé arbitrairement lorsque les véhicules sont générés.

Scénario 1 : Évacuation d'une intersection

Dans ce scénario, un groupe de véhicules ayant une taille fixe approche d'une intersection par différentes voies et avec différentes trajectoires. Une série de tests a été effectuée sur une intersection pour comparer les politiques FCFS, IP(5,0), IP(2,10) et CP. Le nombre de véhicules initial varie de 5 à 50 par pas de 5. Chaque cas est testé sur 100 itérations.

Nous avons mesuré les valeurs suivantes :

- Figure 4.5 : Le nombre de pas de temps nécessaires à l'évacuation de l'intersection
- Figure 4.6 : Le taux de véhicules ayant traversé l'intersection sans avoir à réaliser un arrêt
- Figure 4.7 : La durée d'attente moyenne des véhicules devant l'intersection
- Figure 4.8 : Le nombre de messages échangés par les agents

Sur les figures 4.5, 4.6 et 4.7, les résultats sont proches et il n'y a pas de différence notable entre les 3 politiques testées pour ce scénario. Ce résultat peut s'expliquer par le fait que ce scénario ne permet pas de mettre en avant une amélioration sensible en situation d'évacuation de l'intersection correspondant à une charge ponctuelle de trafic.

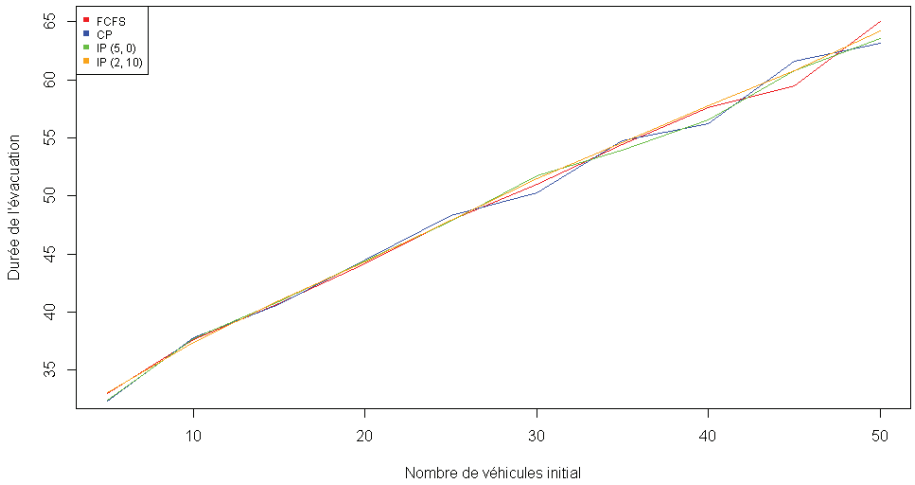


Figure 4.5: Nombre de pas de temps nécessaires à l'évacuation de l'intersection, en fonction du nombre initial de véhicules.

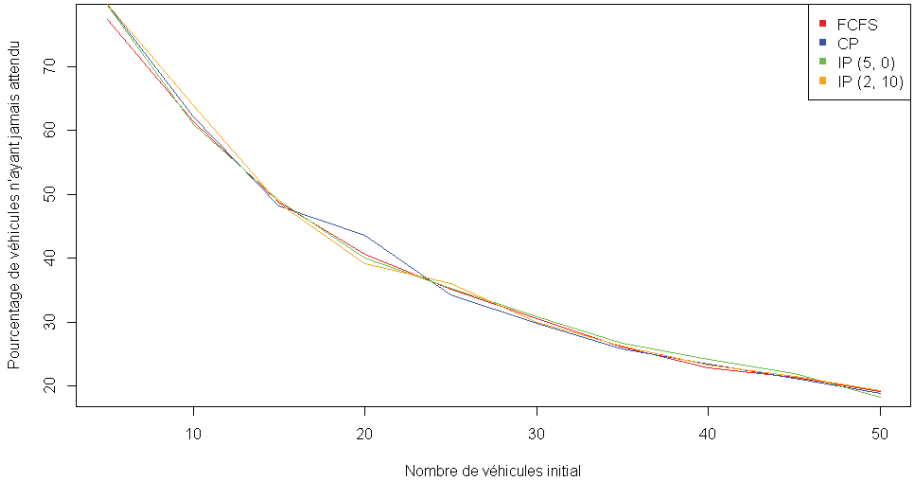


Figure 4.6: Pourcentage de véhicules ayant traversé l'intersection sans devoir s'arrêter, en fonction du nombre initial de véhicules.

La Figure 4.8 montre que le nombre de messages échangés par les agents est supérieur dans le cadre de CP par rapport aux autres politiques. Ce résultat était attendu puisque le nombre d'agents participant à la négociation courante est toujours supérieur dans le cadre de CP, ce qui augmente considérablement le nombre de messages envoyés par les véhicules aux autres véhicules.

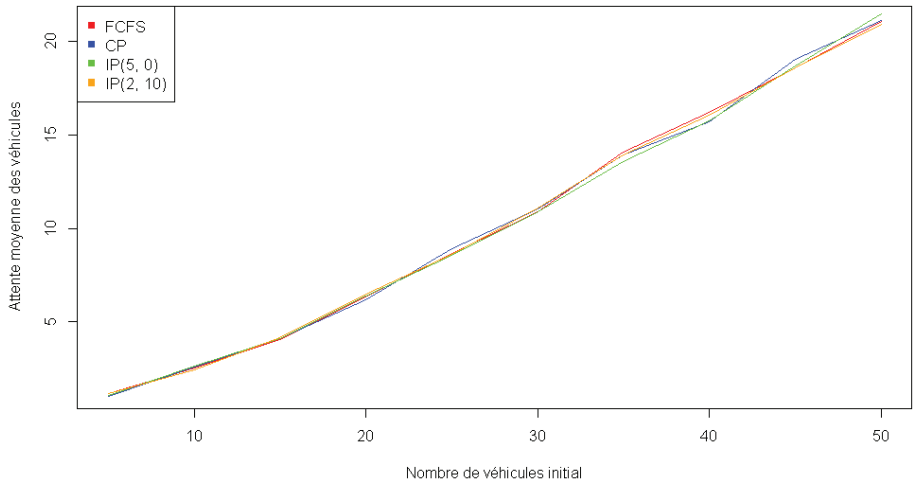


Figure 4.7: Temps d'attente moyen, en pas de temps, pour les véhicules ayant attendu pour pouvoir traverser l'intersection, en fonction du nombre initial de véhicules.

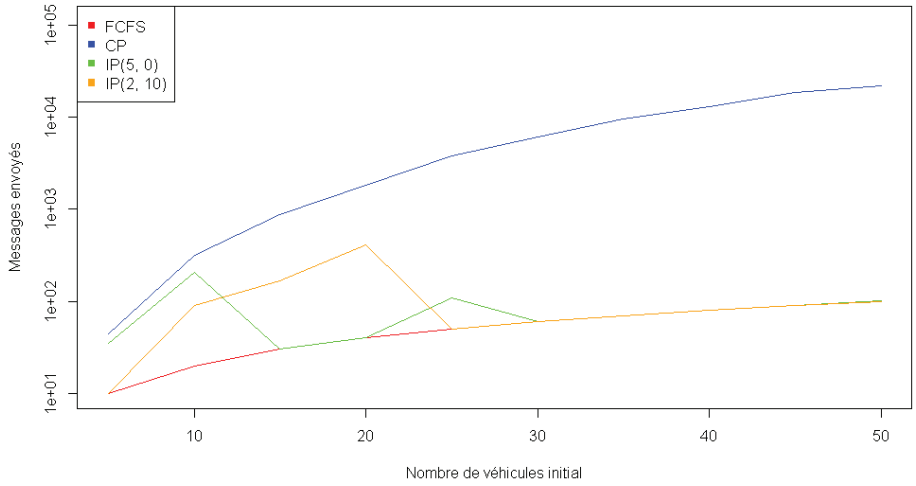


Figure 4.8: Nombre total de messages émis durant la simulation, en fonction du nombre initial de véhicules.

Scénario 2 : Flux continus de véhicules

Dans ce scénario, des flux continus de véhicules approchent d'une intersection par différentes voies et avec différentes trajectoires sur une durée de 1000 pas de temps. Une série de tests a été effectuée sur une intersection pour comparer les politiques FCFS, IP(5,0), IP(2,10) et CP. La densité de véhicules varie de 25% à 100%, par pas de 5%. Un taux de 100% signifie qu'en moyenne un nouveau véhicule approche de l'intersection à chaque pas de temps sur une des voies d'entrée. L'intersection

disposant de 12 voies, le taux maximum serait de 1200%. Chaque cas est testé sur 100 itérations.

Nous avons mesuré les valeurs suivantes :

- Figure 4.9 : taux de véhicules ayant traversé l'intersection sans avoir à réaliser un arrêt
- Figure 4.10 : durée d'attente moyenne des véhicules devant l'intersection
- Figure 4.11 : nombre de messages échangés par les agents

La Figure 4.9 représente le taux moyen de véhicules pouvant franchir l'intersection sans réaliser un arrêt. Cette figure nous montre que ce taux diminue lorsque la densité augmente. FCFS présente les résultats les moins bons sur ce critère, et les résultats d'IP, bien que globalement légèrement meilleurs, sont très proches. CP présente des résultats meilleurs de 10 à 20% du nombre total de véhicules par rapport aux deux autres politiques.

La Figure 4.10 représente le temps d'attente moyen des véhicules devant l'intersection. Ce temps augmente avec la densité, comme attendu. On constate que ce temps est significativement plus bas pour CP que pour FCFS et IP, approchant de la moitié du temps d'attente obtenu avec ces deux autres politiques. Ceci constitue une amélioration certaine des performances. En revanche les performances de IP sont très proches de celles de FCFS, IP ne se montre pas efficace sur ce scénario.

La Figure 4.11 représente le nombre moyen de messages envoyés par les agents durant une simulation. De manière prévisible, ce nombre augmente avec la densité du trafic. On constate que CP nécessite le plus de messages, suivi par IP(5,0), IP(2,10), puis FCFS.

Discussion

Les résultats présentés dans les deux scenarii précédents ne montrent pas d'amélioration sensible de la qualité du trafic par la politique IP, mais montrent une amélioration sensible par la politique CP. Cependant il faut nuancer ces résultats. En effet, dans ces scénarios les véhicules appuient l'ensemble des négociations sur des arguments cohérents entre eux générés aléatoirement et ont des fonctions objectif individuelles également générées aléatoirement. Or la génération d'arguments et de propositions doit s'appuyer sur un contexte de trafic réel pour se montrer pertinente

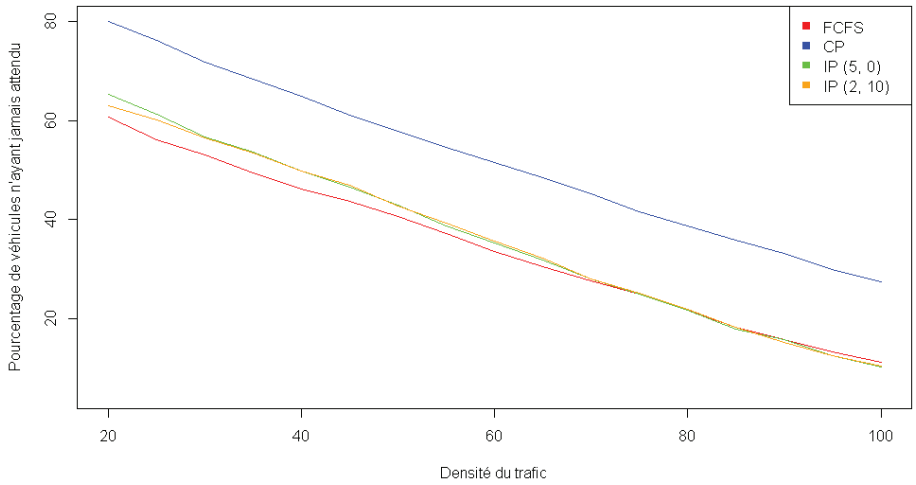


Figure 4.9: Pourcentage de véhicules ayant traversé l'intersection sans devoir s'arrêter, en fonction de la densité du trafic à l'entrée de l'intersection.

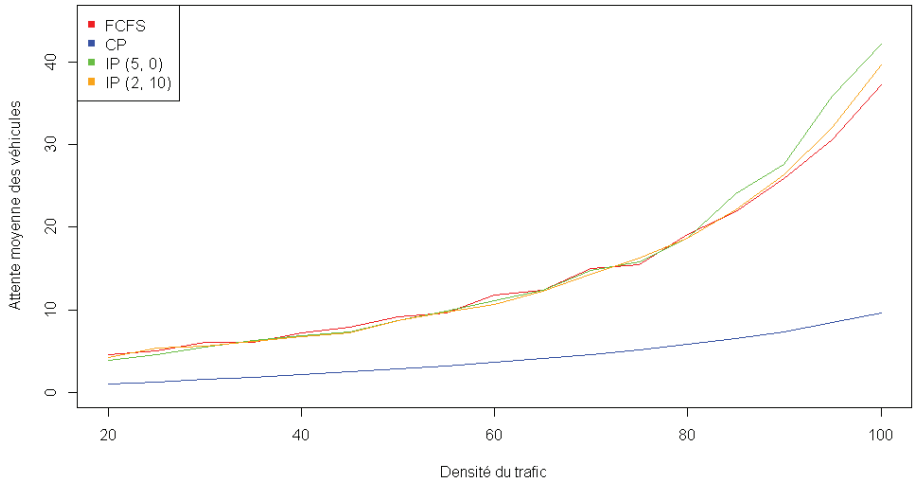


Figure 4.10: Temps d'attente moyen, en pas de temps, pour les véhicules ayant attendu pour pouvoir traverser l'intersection, en fonction de la densité du trafic à l'entrée de l'intersection.

et efficace, en particulier en s'appuyant sur des informations apparaissant à plus large échelle comme expliqué ci-après. Les résultats présentés ici mesurent donc la capacité des différentes politiques à construire des solutions efficaces, en évaluant à quel point chaque politique s'avère gloutonne, plutôt que l'efficacité réelle de chacune de ces politiques sur le trafic.

De plus CP permet une amélioration de la qualité du trafic en échange de coûts communicationnels plus élevés. Un compromis entre CP et FCFS en termes d'efficacité et de coûts de communication pourrait être intéressant car cela permettrait à une

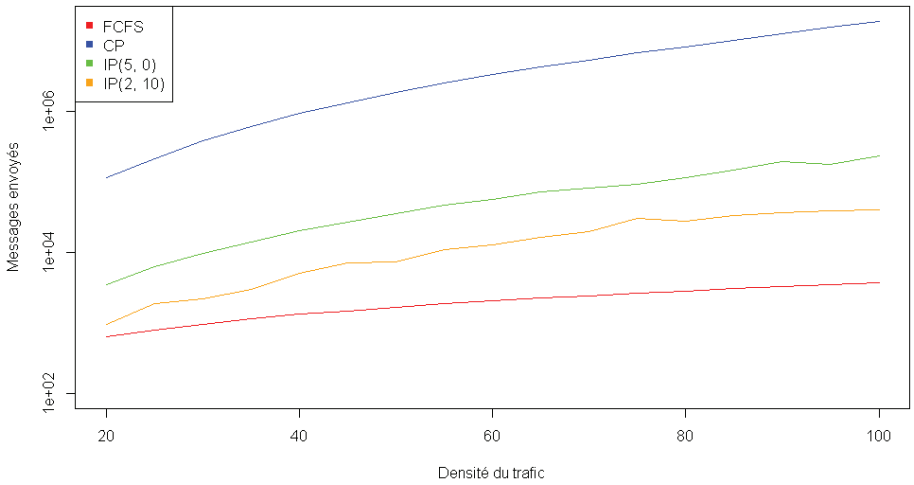


Figure 4.11: Nombre total de messages émis durant la simulation (ici 1000 pas de temps), en fonction de la densité du trafic à l'entrée de l'intersection (échelle logarithmique pour l'axe des ordonnées).

intersection d'utiliser plusieurs politiques en fonction de son niveau de saturation et de la qualité recherchée pour la solution, mais nos résultats semblent indiquer que IP échoue à remplir ce rôle.

Cependant la méthode proposée ici est un outil permettant de prendre en compte des problématiques apparaissant à petite échelle (régulation fine et prise en compte des individus) pour des méthodes utilisées à plus large échelle. Une meilleure évaluation de cette méthode et des différentes politiques proposées consisterait à utiliser cette méthode sur plusieurs intersections, avec des contextes de plus large échelle justifiant l'utilisation d'arguments spécifiques (congestions sur le réseau, présence de bus, zones d'habitation et de bureaux à différents horaires, etc). Cependant notre équipement actuel ne nous permet pas de réaliser de telles simulations. De plus cela nécessiterait un travail théorique complémentaire sur le trafic à plus large échelle que le cadre de cette thèse ne nous permet pas de réaliser entièrement, bien que l'approche proposée pour la coordination du bus avec le reste du trafic soit un premier pas dans cette direction.

4.2 Problème du bus

4.2.1 Implémentation

Afin d'évaluer le modèle de coordination du bus descendant proposé dans le chapitre 3, nous avons implémenté ce modèle en langage Java. Notre implémentation s'appuie

sur un modèle d'environnement différent de celui utilisé à l'échelle de l'intersection, moins détaillé et permettant un fonctionnement à plus large échelle. L'échelle de temps y est également différente, et un pas de temps dans ce modèle vaut 10 pas de temps dans le modèle utilisé pour l'intersection.

Dans ce modèle, les voies et les intersections composant l'environnement sont formées de cellules. Chaque cellule a une capacité maximale de 10 véhicules. Une intersection est représentée par une cellule et une voie est représentée par une série de cellules. La vitesse des véhicules est d'une cellule par pas de temps : à chaque pas de temps, les véhicules sur chaque voie sont déplacés cellule par cellule, en commençant par les cellules en aval de la voie. Lorsque la cellule suivante n'a pas la capacité nécessaire pour recevoir tous les véhicules sur la cellule courante, les véhicules supplémentaires restent sur la cellule courante. Lorsque la cellule suivante est une intersection, le nombre de véhicules admis dans la cellule dépend de la politique de régulation appliquée par cette cellule. Cette politique est déterminée par le mécanisme décrit dans le chapitre 3.

4.2.2 Expérimentation

Dans cette partie, nous présentons les résultats expérimentaux de notre approche sur le problème du bus. Toutes les expérimentations ont été réalisées sur un réseau composé de 5 intersections entre deux stations de bus. Chaque intersection a 2 voies d'entrées et 2 voies de sorties. Le bus se situe à t_0 sur la première cellule du système et sa destination suivante est la dernière cellule de la route principale, c'est-à-dire la route horizontale sur la Figure 4.12. L'objectif du bus est d'atteindre la destination avant $t = 35$. La distance entre les deux stations de bus est de 24 cellules. La vitesse de tous les véhicules est de 1 cellule par unité de temps. Nous assignons ensuite une charge de véhicules distribuée de façon aléatoire sur les différentes voies au Nord et au Sud de l'itinéraire du bus pour pouvoir tester différents cas. De plus, 40 véhicules sont ajoutés aléatoirement sur les voies constituant l'itinéraire du bus.

Stratégie descendante

Dans cette approche, le bus fait une demande de passage et les intersections font une ou plusieurs propositions de politiques individuelles. Ces propositions contiennent la date de passage de bus, le nombre de véhicules qui passent devant le bus et le nombre de véhicules qui s'ajoutent avant le bus dans sa voie. Le bus construit un arbre pour examiner la compatibilité de ces propositions, et le score de chaque composition possible. Il choisit ensuite une politique jointe réalisable qui sera mise en oeuvre par les intersections.

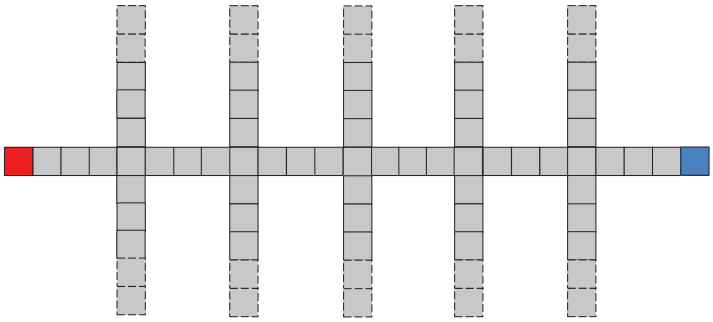


Figure 4.12: Environnement utilisé dans les simulations. Le bus (cellule en rouge) doit se rendre à son prochain arrêt (cellule en bleu) avant $t = 35$. Les voies d'entrée au Nord et au Sud ne sont pas intégralement représentées.

Nous expérimentons 4 versions différentes de cette stratégie. Premièrement, nous faisons varier la fonction objectif utilisée par les intersections pour évaluer leurs politiques individuelles. Deuxièmement, nous comparons la stratégie descendante utilisée seule avec une stratégie couplée à une formation de coalitions d'intersections.

Formation de coalitions

Avec cette approche, une formation de coalitions a lieu entre les intersections préalablement à la coordination entre le bus et les intersections. Pour cela, les intersections s'envoient mutuellement des propositions une fois que le bus a fait sa demande de passage. Les intersections les moins compatibles entre elles, c'est-à-dire ayant le ratio $\frac{nombre\ de\ propositions\ compatibles}{nombre\ de\ propositions\ total}$ le plus faible, forment des coalitions. Ces coalitions sont ensuite considérées comme des entités uniques. Une coalition ne propose pas au bus de politique individuelle d'intersection, mais directement des politiques jointes pour les intersections composant la coalition. Ceci a pour intérêt de réduire l'espace de recherche pour le bus.

Dans les résultats présentés ci-après, les approches A1 et A2 n'ont pas recours à la formation de coalitions préalable, contrairement aux approches B1 et B2.

Fonctions objectif

Dans les expérimentations présentées ici, nous présentons deux fonctions objectifs utilisées pour mesurer l'efficacité d'une politique individuelle. Dans la première approche, la fonction objectif à minimiser est simplement le temps d'attente moyen des véhicules situés sur les voies au sud et nord devant l'intersection. Tous les véhicules

sur le réseau ont un poids généré aléatoirement, et le temps d'attente moyen calculé par cette fonction objectif est pondéré par le poids de chaque véhicule. Dans la seconde approche, la fonction objectif est la somme de ce temps d'attente moyen et d'une pénalité. Cette pénalité a pour but d'inciter le bus à prendre de l'avance dans sa progression, et ainsi à pouvoir s'adapter à une éventuelle modification de l'état du trafic.

La valeur de cette pénalité est calculée de la manière suivante. Soient t_G la date d'arrivée maximale du bus à son arrêt (ici 35), t_{min}^{bus} la date d'arrivée du bus en condition fluide (ici 24), et t_g la date d'arrivée du bus prévue par une politique jointe. La pénalité est nulle si $t_g \leq \frac{t_G + t_{min}^{bus}}{2}$. Sinon, cette pénalité est égale à $p(t_g) = \frac{t_G + t_{min}^{bus}}{t_{min}^{bus} - t_G} + \frac{2 * t_g}{t_G - t_{min}^{bus}}$, qui est la fonction linéaire pour laquelle $p(\frac{t_G + t_{min}^{bus}}{2}) = 0$ et $p(t_G) = 1$. Ceci permet qu'une proposition ne subisse aucune pénalité avec une date d'arrivée suffisamment basse, et que cette pénalité augmente linéairement jusqu'à atteindre son maximum si la d'arrivée est la date limite.

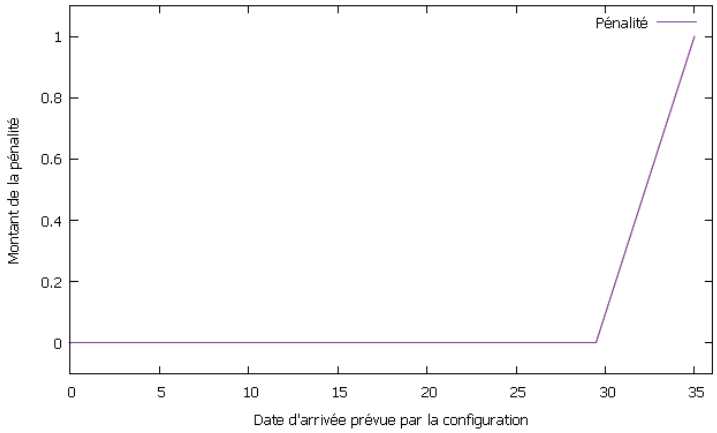


Figure 4.13: Fonction de pénalité pour ce scénario

Dans les résultats présentés ci-après, les approches A1 et B1 utilisent la première fonction objectif, sans pénalité, et les approches A2 et B2 utilisent la seconde, avec pénalité.

4.2.3 Résultats expérimentaux

Nous présentons ici les résultats expérimentaux pour 4 méthodes :

- A1 : sans formation de coalitions préalable, sans pénalité pour les dates tardives
- A2 : sans formation de coalitions préalable, avec pénalité pour les dates tardives
- B1 : avec formation de coalitions préalable, sans pénalité pour les dates tardives

- B1 : avec formation de coalitions préalable, avec pénalité pour les dates tardives

La figure 4.14 présente la date d'arrivée moyenne du bus à sa destination pour chacune des méthodes. On constate que cette date est plus élevée avec les méthodes A1 et B1, c'est-à-dire les méthodes sans pénalité. Cette pénalité remplit donc son objectif de réduction de la date d'arrivée du bus. On ne constate pas de différence nette entre les méthodes A1 et B1 (sans pénalité), en revanche on obtient une date de passage plus élevée avec la méthode B2 que la méthode B1. Pour toutes ces méthodes, on constate une augmentation de la date d'arrivée avec l'augmentation du nombre de véhicules.

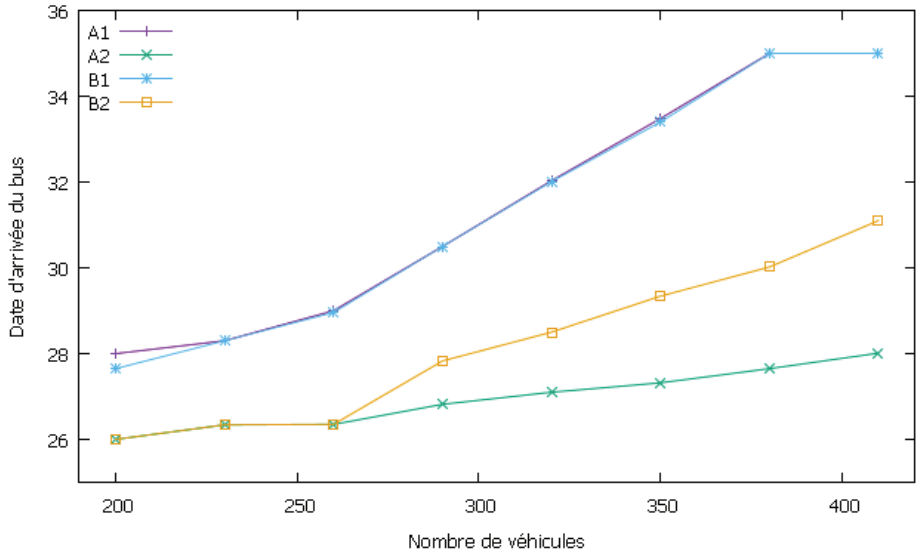


Figure 4.14: Date moyenne d'arrivée du bus à destination en fonction du nombre initial de véhicules sur le réseau.

La figure 4.15 représente la date de passage du bus à la troisième intersection de son itinéraire. Celle-ci est située à 12 cellules du point de départ du bus, et son point d'arrivée est situé à 24 cellules de ce point de départ, cette intersection représente donc la moitié du chemin. Les observations réalisables sur ce graphique sont les mêmes que pour le graphique précédent.

La figure 4.16 représente le rapport entre la date d'arrivée du bus à destination et sa date de passage à la troisième intersection. On constate, pour toutes les méthodes, un rapport proche de 2. Ceci indique qu'il n'y a pas, en moyenne, de différence importante entre le rythme de progression du bus sur le début de son itinéraire et sur la fin de celui-ci. Nous avons vu précédemment que les méthodes avec pénalité produisent bien une date d'arrivée à destination plus basse. Ce graphique nous indique que ceci passe par un rythme de progression globalement plus élevé, et non pas par une accélération du bus en début ou en fin d'itinéraire.

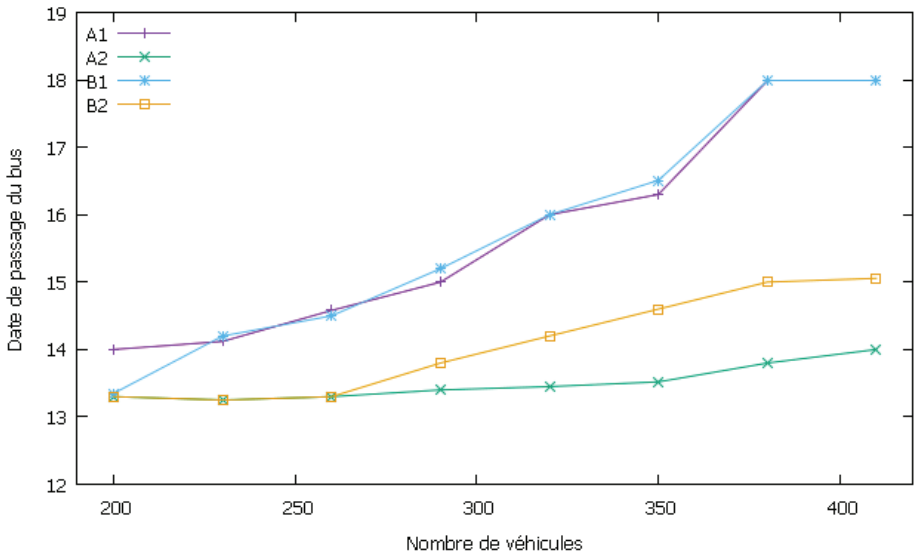


Figure 4.15: Date moyenne de passage du bus à la troisième intersection de son itinéraire en fonction du nombre initial de véhicules sur le réseau.

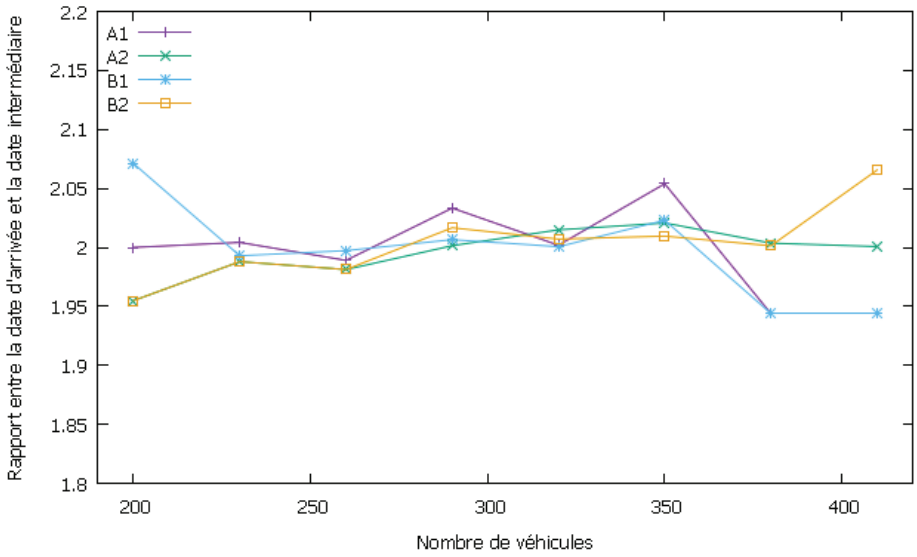


Figure 4.16: Ratio entre la date moyenne de passage du bus à la troisième intersection de son itinéraire et sa date d'arrivée à destination en fonction du nombre initial de véhicules sur le réseau.

La figure 4.17 présente, pour chacune des méthodes, le nombre moyen de messages émis par le bus et les intersections avant d'obtenir une première politique jointe valide. On constate que ce nombre de messages augmente peu avec le nombre de véhicules présents, ce qui indique que celui-ci ne complexifie pas nécessairement la recherche de solutions. On constate une différence prévisible du nombre de messages entre les méthodes avec et sans coalitions. On observe également que le nombre de messages émis est légèrement supérieur dans le cadre des méthodes avec pénalité.

La figure 4.18 présente, pour chacune des méthodes, le temps moyen de calcul avant d'obtenir une première politique jointe valide. Ces simulations ont été réalisées sur un ordinateur ayant un processus Core i3 4130 3.4Ghz 2 coeurs, disposant de 8GB de RAM.

Les résultats montrent à la fois un temps de calcul et un nombre de messages très bas, qui permettent sans difficulté l'utilisation de ce mécanisme en temps réel, voire une utilisation dynamique qui permettrait au bus d'utiliser ce mécanisme entre deux arrêts pour s'adapter à une évolution imprévue des conditions de trafic. On constate que le temps de calcul augmente assez peu, ce qui suggère que le mécanisme s'adapte facilement à des conditions de trafic denses.

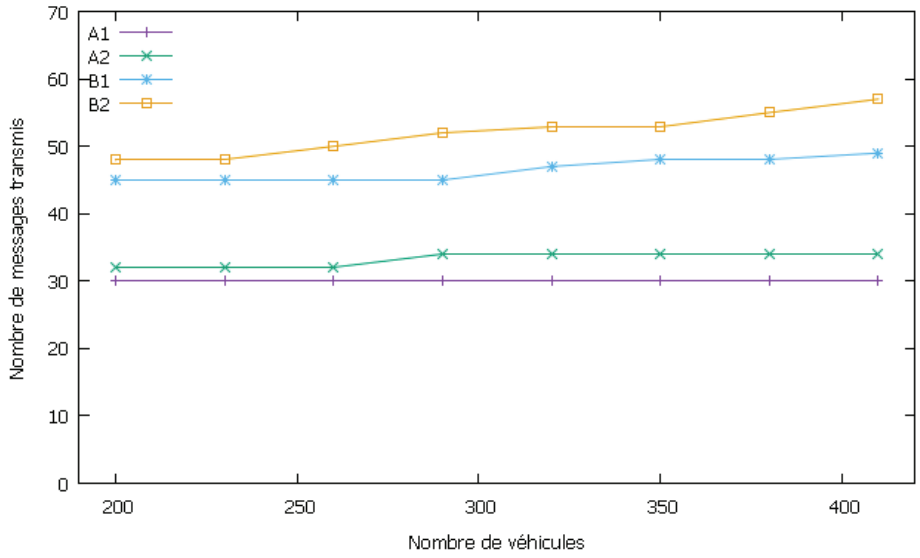


Figure 4.17: Nombre moyen de messages émis avant l'obtention d'une première politique jointe valide.

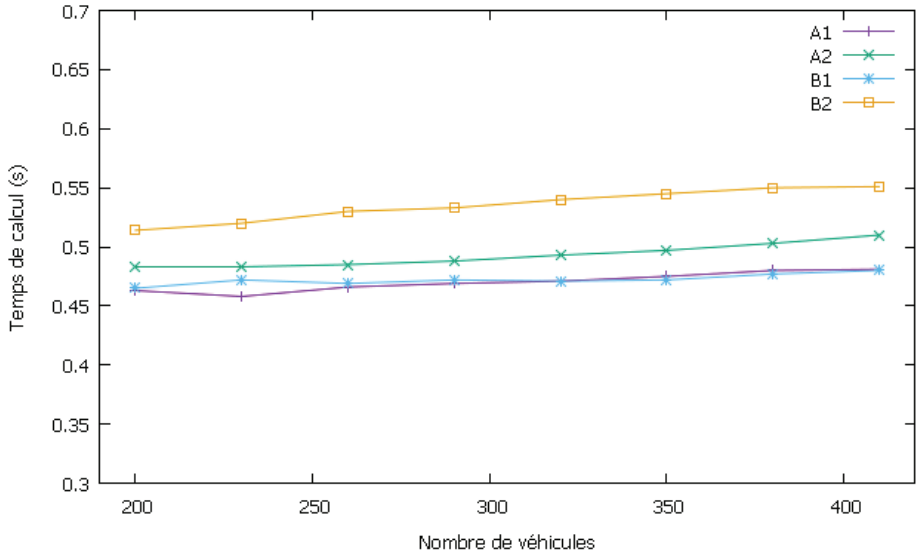


Figure 4.18: Temps moyen de calcul avant l'obtention d'une première politique jointe.

La figure 4.19 représente le score moyen des configurations choisies avec la fonction objectif, qui doit être minimisé. On constate que ce score est bien sur plus élevé pour les méthodes avec pénalité, ce qui est évident puisque le principe de cette pénalité est d'augmenter ce score. On constate que ce score augmente avec le nombre de véhicules. On observe également que le score de la méthode B2 est meilleur que celui de la méthode A2. Ceci indique que les coalitions ont un impact positif sur la qualité des solutions produites, et semble valider l'utilisation de celles-ci comme heuristiques de recherche.

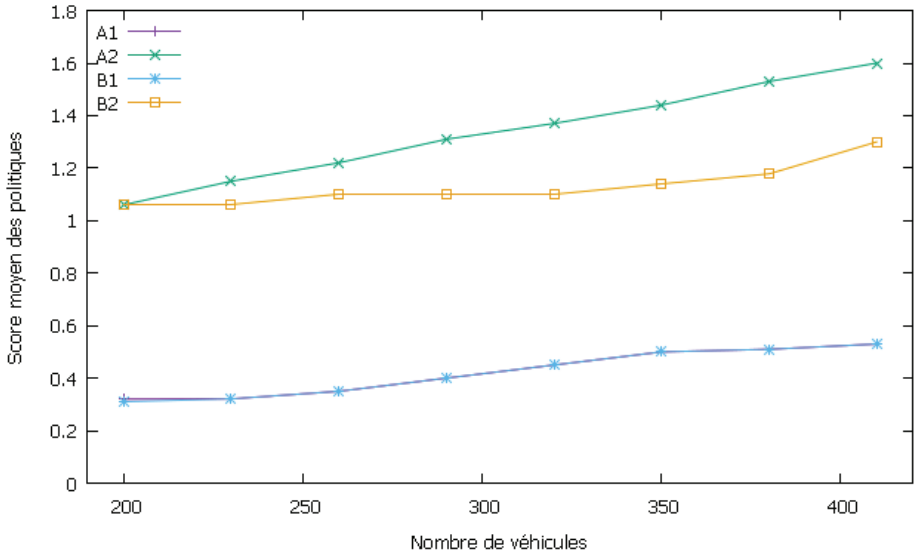


Figure 4.19: Score moyen des politiques adoptées.

Discussion

Les résultats pour ces simulations montrent plusieurs choses.

Premièrement, le temps de calcul et le nombre de messages échangés sont relativement bas, ce qui prouve qu'un mécanisme de ce type serait utilisable en temps réel.

Deuxièmement, on constate que l'utilisation d'une fonction objectif comportant une pénalité pour les dates d'arrivée tardives permet d'orienter la recherche de solutions vers des dates d'arrivée plus basses. Une telle stratégie permettrait au bus de conserver une marge de manoeuvre durant sa progression, qui rendrait possible l'utilisation de ce mécanisme de coordination même entre deux arrêts en cas de modification dynamique de l'état du trafic.

Troisièmement, on constate que l'utilisation de coalitions d'intersections préalablement à la recherche de solutions est une heuristique qui peut augmenter les coûts

communicationnels et computationnels. Bien que B2 produise des dates de passage plus tardives que A2, le score obtenu est plus bas avec B2 que A2. Ceci signifie que B2 réalise une meilleure optimisation que A2, en réduisant le temps d'attente moyen des véhicules par rapport à A2. Nous avons expliqué dans le chapitre 3 que différents éléments susceptibles d'être intégrés à la fonction objectif des intersections. On peut supposer que cette formation de coalitions constituerait une bonne heuristique avec une fonction objectif plus complexe.

Ces résultats semblent indiquer que la mise en oeuvre d'une telle méthode de coordination entre le bus et les intersections est possible et pourrait influencer positivement la progression des bus dans le trafic. La prise en compte dans les simulations des contraintes de mise en application de ce mécanisme à l'échelle de l'intersection, c'est-à-dire une implémentation qui ferait le lien entre les deux mécanismes présentés respectivement dans les chapitre 2 et 3 de cette thèse, permettrait de continuer la validation de cette méthode.

4.3 Conclusion

Nous avons détaillé dans ce chapitre les implémentations qui ont été faites des mécanismes présentés dans les deux chapitres précédents. Nous avons également présenté les résultats des simulations qui ont été effectuées. Le chapitre suivant conclut cette thèse et présente les différentes perspectives liées aux travaux qui y sont présentés.

5 Conclusion et perspectives

L'objectif de cette thèse est le développement de nouvelles méthodes pour la régulation de trafic tirant parti des capacités de communication apportées aux véhicules par les technologies récentes afin d'améliorer le trafic, en particulier pour les transports en commun. Dans notre approche, nous avons proposé deux mécanismes complémentaires fonctionnant à deux échelles différentes.

Notre premier mécanisme a pour but de permettre une coordination des véhicules à l'approche d'une intersection afin de déterminer une date d'admission dans l'intersection pour chacun de ces véhicules. Pour cela, notre méthode met en place une négociation automatique du droit de passage : les véhicules échangent des informations sur l'état du trafic et des arguments permettant de décider collectivement d'une configuration pour l'intersection, c'est-à-dire de l'ensemble des dates d'admission pour chacun des véhicules. Cette configuration concerne les véhicules approchant de l'intersection de manière immédiate, et la question de la continuité du flux de véhicule approchant est traitée.

Notre second mécanisme a pour but d'anticiper la progression d'un bus à l'échelle de quelques intersections et de permettre à ce bus d'atteindre son prochain arrêt sans subir de retard à cause de la congestion. Pour cela, le bus se coordonne avec les intersections sur son itinéraire et définit avec celles-ci une politique de régulation commune permettant au bus d'atteindre son objectif horaire en pénalisant le moins possible le trafic pour les véhicules environnants.

Ces deux mécanismes ont fait l'objet d'implémentation et des simulations ont permis d'effectuer les premières évaluations de ces mécanismes.

Différentes perspectives d'évolution de ces travaux sont possibles.

Premièrement le mécanisme de régulation de l'intersection a été implémenté avec deux politiques de continuité appelées CP (Continuous Policy) et IP (Iterated Policy). CP a l'inconvénient d'être relativement coûteuse en communications, et les résultats actuels d'IP semblent indiquer que celle-ci s'avère relativement peu efficace. Une politique de continuité intermédiaire en termes d'efficacité et de communication

permettrait probablement une mise-en-oeuvre d'un tel mécanisme plus adaptée à certains niveaux de saturation du trafic.

Deuxièmement le mécanisme de coordination entre les bus et les intersections ne permet, actuellement, l'anticipation de la progression que pour un unique bus. Or il est commun que plusieurs bus aient des trajectoires partiellement partagées ou conflictuelles, et il serait pertinent de s'interroger sur la manière dont se ferait une telle coordination. Il pourrait, par exemple, être pertinent de mettre en place un système de contrats entre les différents bus concernant leurs progressions respectives, ou de faire en sorte que ce mécanisme de coordination entre un bus et les intersections de sa trajectoire produise plusieurs solutions, et que chaque bus identifie les compatibilités entre les solutions produites par les intersections de sa trajectoire pour sa progression avec celles produites pour permettre la progression d'un autre bus.

De plus ce mécanisme de coordination entre le bus et les intersections s'appuie sur le mécanisme de régulation à l'échelle de l'intersection. Actuellement la manière dont une intersection met en oeuvre une politique décidée dans le cadre du mécanisme de coordination du bus est relativement simple, et il serait intéressant de se pencher plus sur la manière dont ces deux mécanismes peuvent fonctionner ensemble car une méthode plus sophistiquée pourrait améliorer le fonctionnement du mécanisme de coordination, en termes d'efficacité ou de possibilités.

Enfin, ces travaux mériteraient probablement d'être étendus pour traiter d'une autre problématique, à savoir celle des pelotons de véhicules. En effet, les systèmes coopératifs permettent aujourd'hui aux véhicules d'évoluer en pelotons partageant une portion d'itinéraire et adoptant un comportement commun, par exemple pour le démarrage des véhicules. Les deux mécanismes proposés dans cette thèse permettraient de répondre à certaines problématiques qui se posent aux pelotons, comme la facilitation de leur progression ou le maintien de l'unité du peloton. En effet, le mécanisme permettant actuellement aux bus de franchir une série d'intersections sans rencontrer de congestion pourrait être utilisé pour donner des ondes vertes à des pelotons de véhicules, et le maintien de la cohérence du peloton est un argument susceptible d'être utilisé dans la négociation du droit de passage à l'intersection et qui permettrait d'augmenter l'intérêt de ce phénomène lors de la traversée des intersections. Compte tenu de ces possibilités, il serait pertinent de s'intéresser à la formation, au maintien et à la dissolution de telles coalitions de véhicules, ainsi qu'à la manière dont ceux-ci progressent et peuvent être traités par les deux mécanismes présentés dans cette thèse. Un premier travail dans ce sens a été déjà amorcé dans le cadre d'un stage de Master 2 qui a été effectué dans notre groupe en collaboration avec le groupe Renault [36].

Bibliographie

[1]Abdeljalil ABBAS-TURKI, Florent PERRONNET, Jocelyn BUISSON et al. « Cooperative intersections for emerging mobility systems ». In : *15th meeting of the Euro Working Group on transportation*. 2012 (cité en page 36).

[2]Monireh ABDOOS, Nasser MOZAYANI et Ana LC BAZZAN. « Holonic multi-agent system for traffic signals control ». In : *Engineering Applications of Artificial Intelligence* 26.5 (2013), p. 1575–1587 (cité en page 20).

[3]Leila AMGOUD, Sihem BELABBES et Henri PRADE. « Towards a formal framework for the search of a consensus between autonomous agents ». In : *International Workshop on Argumentation in Multi-Agent Systems*. Springer. 2005, p. 264–278 (cité en page 37).

[4]Gabriel BALAN et Sean LUKE. « History-based traffic control ». In : *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. ACM. 2006, p. 616–621 (cité en pages 15, 22).

[5]Flavien BALBO, Neïla BHOURI et Suzanne PINSON. « Bimodal traffic regulation system : A multi-agent approach ». In : *Web Intelligence*. T. 14. 2. IOS Press. 2016, p. 139–151 (cité en pages 26, 27).

[6]Kevin BALKE, Conrad DUDEK et Thomas URBANIK II. « Development and evaluation of intelligent bus priority concept ». In : *Transportation Research Record : Journal of the Transportation Research Board* 1727 (2000), p. 12–19 (cité en page 25).

[7]J BARCELO. « CARS, A Demand-Responsive Traffic Control System ». In : *Applications of advanced technologies in transportation engineering : proceedings of the second international conference*. 1991 (cité en page 13).

[8]Ana LC BAZZAN et Franziska KLÜGL. « A review on agent-based technology for traffic and transportation ». In : *The Knowledge Engineering Review* 29.03 (2014), p. 375–403 (cité en pages 14, 29).

[9]Ana LC BAZZAN, Denise DE OLIVEIRA, Franziska KLÜGL et Kai NAGEL. « To adapt or not to adapt–consequences of adapting driver and traffic light agents ». In : *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*. Springer, 2008, p. 1–14 (cité en pages 20, 21).

[10]Neïla BHOURI, Florence BOILLOT et Pierre VINANT. « Régulation multimodale du trafic routier et des transports en commun de surface : Une classification des méthodes ». In : *Recherche, transports, sécurité* 98 (2008), p. 53–72 (cité en pages 24, 26).

[11]Chr Bielefeldt et F Busch. « MOTION-a new on-line traffic signal network control system ». In : *Road Traffic Monitoring and Control, 1994., Seventh International Conference on*. IET. 1994, p. 55–59 (cité en page 13).

[12]Florence Boillot, Sophie Midenet et Jean-Claude Pierrelee. « The real-time urban traffic control system CRONOS : Algorithm and experiments ». In : *Transportation Research Part C : Emerging Technologies* 14.1 (2006), p. 18–38 (cité en page 12).

[13]Elmar Brockfeld, Robert Barlovic, Andreas Schadschneider et Michael Schreckenberg. « Optimizing traffic lights in a cellular automaton model for city traffic ». In : *Physical Review E* 64.5 (2001), p. 056132 (cité en page 31).

[14]Cameron B Browne, Edward Powley, Daniel Whitehouse et al. « A survey of monte carlo tree search methods ». In : *IEEE Transactions on Computational Intelligence and AI in Games* 4.1 (2012), p. 1–43 (cité en page 94).

[15]Eduardo Camponogara et Werner Kraus Jr. « Distributed learning agents in urban traffic control ». In : *Progress in Artificial Intelligence*. Springer, 2003, p. 324–335 (cité en page 19).

[16]Gabriel Rodrigues de Campos, Paolo Falcone et Jonas Sjöberg. « Autonomous cooperative driving : a velocitybased negotiation approach for intersections crossing ». In : *16th International IEEE Conference on Intelligent Transportation Systems*. 2013 (cité en page 18).

[17]Alexis Champion. « Mécanisme de coordination multi-agent fondé sur des jeux : applicationa la simulation comportementale de trafic routier en situation de carrefour ». In : *P h D thesis, Université de V alenciennes et du H ainaut-Cambrésis* 2.00 (2003), p. 3 (cité en page 18).

[18]Min Chee Choy, Dipti Srinivasan et Ruey Long Cheu. « Cooperative, hybrid agent architecture for real-time traffic signal control ». In : *IEEE SMC* 33.5 (2003), p. 597–607 (cité en page 36).

[19]Denise De Oliveira, Ana LC Bazzan et Victor Lesser. « Using cooperative mediation to coordinate traffic lights : a case study ». In : *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. ACM. 2005, p. 463–470 (cité en page 29).

[20]Christina Diakaki. « Integrated control of traffic flow in corridor networks ». Thèse de doct. Technical University Of Crete, 1999 (cité en page 11).

[21]N Djemame et S Espie. « ARCHISIM : MULTI-ACTOR PARALLEL ARCHITECTURE FOR TRAFFIC SIMULATION ». In : *Steps Forward. Intelligent Transport Systems World Congress*. Volume 4. 1995 (cité en page 31).

[22]Arnaud Doniec, René Mandiau, Sylvain Piechowiak et Stéphane Espié. « Anticipation based on constraint processing in a multi-agent context ». In : *Autonomous Agents and Multi-Agent Systems* 17.2 (2008), p. 339–361 (cité en page 17).

[23]Kurt Dresner et Peter Stone. « A multiagent approach to autonomous intersection management ». In : *Journal of artificial intelligence research* 31 (2008), p. 591–656 (cité en pages 16, 21, 23, 29).

[24]Kurt M Dresner et Peter Stone. « Sharing the Road : Autonomous Vehicles Meet Human Drivers. » In : *IJCAI*. T. 7. 2007, p. 1263–1268 (cité en pages 16, 30).

[25]Peter DUERR. « Dynamic right-of-way for transit vehicles : integrated modeling approach for optimizing signal control on mixed traffic arterials ». In : *Transportation Research Record : Journal of the Transportation Research Board* 1731 (2000), p. 31–39 (cité en page 26).

[26]Edmund H. DURFEE, Victor R. LESSER et Daniel D. CORKILL. « Trends in cooperative distributed problem solving ». In : *IEEE Transactions on knowledge and data Engineering* 1.1 (1989), p. 63–83 (cité en page 22).

[27]Martin FELLENDORF. « VISSIM : A microscopic simulation tool to evaluate actuated signal control including bus priority ». In : *64th Institute of Transportation Engineers Annual Meeting*. Springer. 1994, p. 1–9 (cité en page 31).

[28]Yuqi FENG, Joseph PERRIN et Peter T MARTIN. « Bus priority of scoot evaluated in a vissim simulation environment ». In : *Transportation Research Board 82nd Annual Meeting*. 2003 (cité en page 26).

[29]John FRANCE et Ali A GHORBANI. « A multiagent system for optimizing urban traffic ». In : *Intelligent Agent Technology, 2003. IAT 2003. IEEE/WIC International Conference on*. IEEE. 2003, p. 411–414 (cité en page 19).

[30]Edith FROLOFF, Michel RIZZI et Antoine SAPORITO. « Bases et pratiques de la régulation ». In : *RATP, direction du réseau routier RC/MSE* (1989) (cité en page 24).

[31]Matthis GACIARZ, Samir AKNINE et Neïla BHOURI. « A coalition-based approach for cooperative urban traffic regulation ». In : *EUMAS (European Workshop on Multi-Agent Systems)*. 2013, p. 1–4 (cité en page 30).

[32]Matthis GACIARZ, Samir AKNINE et Neïla BHOURI. « A Continuous Negotiation Based Model for Traffic Regulation at an Intersection ». In : *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents et Multiagent Systems. 2015, p. 1791–1792 (cité en page 30).

[33]Matthis GACIARZ, Samir AKNINE et Neila BHOURI. « Automated negotiation for traffic regulation ». In : *Advances in Social Computing and Multiagent Systems*. Springer, 2015, p. 1–18 (cité en page 30).

[34]Matthis GACIARZ, Samir AKNINE et Neila BHOURI. « Constraint-based negotiation model for traffic regulation ». In : *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2015 IEEE/WIC/ACM International Conference on*. T. 2. IEEE. 2015, p. 320–327 (cité en page 30).

[35]Matthis GACIARZ, Neila BHOURI, Samir AKNINE et Champs sur MARNE. « Continuous Negotiation for a Vehicle-Regulated Intersection ». In : *ARTS ECR (Autonomic Road Transport Support Systems Early Career Researcher Conference)*. 2015 (cité en page 30).

[36]Alexandre GALDEANO. « La conduite en pelotons (platooning) ». Mém.de mast. EPU Lyon, 2016 (cité en page 122).

[37]Nathan H GARTNER, Farhad J POORAN et Christina M ANDREWS. « Implementation of the OPAC adaptive control strategy in a traffic signal network ». In : *Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE*. IEEE. 2001, p. 195–200 (cité en page 12).

[38]Jean GRÉGOIRE, Silvère BONNABEL et Arnaud de LA FORTELLE. « Optimal cooperative motion planning for vehicles at intersections ». In : *arXiv preprint arXiv :1310.7729* (2013) (cité en page 16).

[39]MR HAFNER, D CUNNINGHAM, L CAMINITI et D DEL VECCHIO. « Automated vehicle-to-vehicle collision avoidance at intersections ». In : *Proceedings of World Congress on Intelligent Transport Systems*. 2011 (cité en page 18).

[40]Matthew HAUSKNECHT, Tsz-Chiu AU et Peter STONE. « Autonomous intersection management : Multi-intersection optimization ». In : *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE. 2011, p. 4581–4586 (cité en page 20).

[41]Jean-Jacques HENRY, Jean Loup FARGES et J TUFFAL. « The PRODYN real time traffic algorithm ». In : *IFACIFIPIFORS conference on control in*. 1984 (cité en page 11).

[42]PB HUNT, DI ROBERTSON, RD BRETHERTON et M Cr ROYLE. « The SCOOT on-line traffic signal optimisation technique ». In : *Traffic Engineering & Control* 23.4 (1982) (cité en page 10).

[43]S KACHROUDI, N BHOURI, S MAMMAR et G SCEMAMA. « Modèle de prdiction semimacroscopique pour les véhicules de transport en commun en milieu urbain ». In : *Conférence Internationale Francophone d'automatique (CIFA)*. 2008 (cité en page 27).

[44]Sofiene KACHROUDI. « Commande et optimisation pour la régulation du trafic urbain mutimodale sur de grands réseaux urbains ». Thèse de doct. Evry-Val d'Essonne, 2010 (cité en page 27).

[45]Iisakki KOSONEN. « Multi-agent fuzzy signal control based on real-time simulation ». In : *Transportation Research Part C : Emerging Technologies* 11.5 (2003), p. 389–403 (cité en pages 19, 22).

[46]Vipin KUMAR. « Algorithms for constraint-satisfaction problems : A survey ». In : *AI magazine* 13.1 (1992), p. 32 (cité en page 34).

[47]Joyoung LEE et Byungkyu PARK. « Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment ». In : *Intelligent Transportation Systems, IEEE Transactions on* 13.1 (2012), p. 81–90 (cité en pages 18, 21, 22).

[48]Jennie LIORIS, Ramtin PEDARSANI, Fatma Yildiz TASCIKARAOGLU et Pravin VARAIYA. « Doubling throughput in urban roads by platooning ». In : () (cité en page 48).

[49]Sven MAERIVOET et Bart DE MOOR. « Cellular automata models of road traffic ». In : *Physics Reports* 419.1 (2005), p. 1–64 (cité en page 31).

[50]Ivan MARSA-MAESTRE, Enrique de la HOZ, Catholijn M JONKER et Mark KLEIN. « Using graph properties to enable better negotiations in complex self-interested networks ». In : *The Eighth International Workshop on Agent-based Complex Automated Negotiations (ACAN2015)* (cité en page 20).

[51]Vito MAURO et C DI TARANTO. « Utopia ». In : *Control, computers, communications in transportation* (1990) (cité en page 12).

[52]Pitu MIRCHANDANI et Larry HEAD. « A real-time traffic signal control system : architecture, algorithms, and analysis ». In : *Transportation Research Part C : Emerging Technologies* 9.6 (2001), p. 415–432 (cité en pages 13, 26).

[53]Javier MORALES, Maite LÓPEZ-SÁNCHEZ et Marc ESTEVA. « Using experience to generate new regulations ». In : *IJCAI*. Citeseer. 2011, p. 307–312 (cité en page 17).

[54]Markos PAPAGEORGIOU. « Traffic control ». In : *Handbook of transportation science*. Springer, 2003, p. 243–277 (cité en page 6).

[55]AJ RICHARDSON et KW OGDEN. « Evaluation of active bus-priority signals ». In : *Transportation research record* 718 (1979), p. 5–12 (cité en page 25).

[56]Dennis I ROBERTSON. « TRANSYT : a traffic network study tool ». In : (1969) (cité en page 7).

[57]Danko A ROOZEMOND. « Using intelligent agents for pro-active, real-time urban intersection control ». In : *European Journal of Operational Research* 131.2 (2001), p. 293–301 (cité en page 29).

[58]Heiko SCHEPPERLE et Klemens BÖHM. « Agent-based traffic control using auctions ». In : *Cooperative Information Agents XI*. Springer, 2007, p. 119–133 (cité en pages 17, 22).

[59]Heiko SCHEPPERLE, Klemens BÖHM et Simone FORSTER. « Traffic management based on negotiations between vehicles–a feasibility demonstration using agents ». In : *Agent-Mediated Electronic Commerce and Trading Agent Design and Analysis*. Springer, 2009, p. 90–104 (cité en page 17).

[60]Arthur G SIMS et Kenneth W DOBINSON. « The Sydney coordinated adaptive traffic (SCAT) system philosophy and benefits ». In : *IEEE Transactions on Vehicular Technology* 29.2 (1980), p. 130–137 (cité en page 10).

[61]*Systèmes coopératifs*. `http://www.transport-intelligent.net/technologies/systemes-cooperatifs/`. Accessed : 2016-09-01 (cité en page 10).

[62]Matteo VASIRANI et Sascha OSSOWSKI. « A market-inspired approach to reservation-based urban road traffic management ». In : *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents et Multiagent Systems. 2009, p. 617–624 (cité en page 17).

[63]Joannes VERMOREL et Mehryar MOHRI. « Multi-armed bandit algorithms and empirical evaluation ». In : *European conference on machine learning*. Springer. 2005, p. 437–448 (cité en page 89).

[64]RA VINCENT et JR PEIRCE. *'MOVA' : Traffic Responsive, Self-optimising Signal Control for Isolated Intersections*. Rapp. tech. 1988 (cité en pages 13, 25).

[65]Fei YAN, Jia WU et Mahjoub DRIDI. « A scheduling model and complexity proof for autonomous vehicle sequencing problem at isolated intersections ». In : *Service Operations and Logistics, and Informatics (SOLI), 2014 IEEE International Conference on*. IEEE. 2014, p. 78–83 (cité en page 16).

[66]Xi ZOU et David LEVINSON. « Vehicle-based intersection management With intelligent agents ». In : *ITS America Annual Meeting. Minneapolis, Minnesota*. 2003 (cité en page 15).

Table des figures

Liste des tableaux