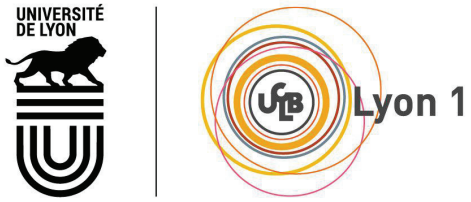
SmartGov : architecture générique pour la co-construction de politiques urbaines basée sur l'apprentissage par renforcement multi-agent

Simon Pageaud



N° d'ordre NNT : 2019LYSE1128

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

École Doctorale ED512
Informatique et Mathématiques

Spécialité de doctorat : Informatique

Soutenue publiquement le 06/09/2019, par :
Simon Pageaud

SmartGov : Architecture générique pour la co-construction de politiques urbaines basée sur l'apprentissage par renforcement multi-agent

---

Devant le jury composé de :

Amblard Frédéric, Professeur, Université Toulouse 3	Président du Jury
Chevrier Vincent, Professeur, Université de Lorraine	Rapporteur
Galland Stéphane, Professeur, Université de Technologie de Belfort-Montbéliard	Rapporteur
Amblard Frédéric, Professeur, Université Toulouse 3	Examinateur
Duchêne Éric, Maître de Conférences, Université Lyon 1	Examinateur
Dugdale Julie, Maître de Conférences, Université Grenoble Alpes	Examinatrice
Hassas Salima, Professeure, Université Lyon 1	Directrice de thèse
Deslandres Véronique, Maître de Conférences, Université Lyon 1	Co-encadrante de thèse
Lehoux Vassilissa, Chercheure, NAVER LABS EUROPE	Co-encadrante de thèse

UNIVERSITE CLAUDE BERNARD - LYON 1

Président de l'Université	**M. le Professeur Frédéric FLEURY**
Président du Conseil Académique	M. le Professeur Hamda BEN HADID
Vice-président du Conseil d'Administration	M. le Professeur Didier REVEL
Vice-président du Conseil Formation et Vie Universitaire	M. le Professeur Philippe CHEVALIER
Vice-président de la Commission Recherche	M. Fabrice VALLÉE
Directrice Générale des Services	Mme Dominique MARCHAND

COMPOSANTES SANTE

Faculté de Médecine Lyon Est — Claude Bernard	Directeur : M. le Professeur G. RODE
Faculté de Médecine et de Maïeutique Lyon Sud — Charles Mérieux	Directeur : Mme la Professeure C. BURILLON
Faculté d'Odontologie	Directeur : M. le Professeur D. BOURGEOIS
Institut des Sciences Pharmaceutiques et Biologiques	Directeur : Mme la Professeure C. VINCIGUERRA
Institut des Sciences et Techniques de la Réadaptation	Directeur : M. X. PERROT
Département de formation et Centre de Recherche en Biologie Humaine	Directeur : Mme la Professeure A-M. SCHOTT

COMPOSANTES ET DEPARTEMENTS DE SCIENCES ET TECHNOLOGIE

Faculté des Sciences et Technologies	Directeur : M. F. DE MARCHI
Département Biologie	Directeur : M. le Professeur F. THEVENARD
Département Chimie Biochimie	Directeur : Mme C. FELIX
Département GEP	Directeur : M. Hassan HAMMOURI
Département Informatique	Directeur : M. le Professeur S. AKKOUCHE
Département Mathématiques	Directeur : M. le Professeur G. TOMANOV
Département Mécanique	Directeur : M. le Professeur H. BEN HADID
Département Physique	Directeur : M. le Professeur J-C PLENET
UFR Sciences et Techniques des Activités Physiques et Sportives	Directeur : M. Y. VANPOULLE
Observatoire des Sciences de l'Univers de Lyon	Directeur : M. B. GUIDERDONI
Polytech Lyon	Directeur : M. le Professeur E. PERRIN
Ecole Supérieure de Chimie Physique Electronique	Directeur : M. G. PIGNAULT
Institut Universitaire de Technologie de Lyon 1	Directeur : M. le Professeur C. VITON
Ecole Supérieure du Professorat et de l'Education	Directeur : M. le Professeur A. MOUGNIOTTE
Institut de Science Financière et d'Assurances	Directeur : M. N. LEBOISNE

Résumé

Dans cette thèse, nous proposons un outil SmartGov, mixant simulation multi-agents et apprentissage multi-agents par renforcement profond, pour permettre la co-construction de politiques urbaines et inscrire les acteurs de la ville dans la boucle de conception. La Smart City permet à l'outil d'intégrer les données collectées par les capteurs présents dans la ville pour la modéliser de façon réaliste. Notre première contribution est une architecture générique pour construire une simulation multi-agents représentant la ville, et étudier l'émergence de comportement globaux avec des agents réalistes capables de réagir aux décisions politiques. Grâce à une modélisation multi-niveaux, et le couplage de différentes dynamiques, le système apprend les spécificités de l'environnement pour proposer des politiques pertinentes. Notre seconde contribution concerne l'autonomie et l'adaptation de la couche décisionnelle avec un apprentissage par renforcement multi-agents et multi-niveaux. Un ensemble d'agents, regroupés en clusters, est distribué dans le périmètre étudié pour apprendre des spécificités locales sans connaissance a priori de son environnement. L'attribution d'un score de confiance et de récompenses individuelles permettent d'atténuer l'impact de la non-stationnarité sur la réutilisation d'expériences nécessaire à l'apprentissage profond. Ces contributions conduisent à un système complet de co-construction de politiques urbaines dans le contexte de la Smart City. Nous comparons notre modèle avec d'autres approches de la littérature sur une politique de tarification du stationnement urbain, afin de mettre en évidence les apports et les limites de nos contributions.

Mots-clés

Simulation multi-agents ; Apprentissage par renforcement multi-agents profond ; Couplage de dynamiques ;

Abstract

In this thesis, we propose the SmartGov model, coupling multi-agent simulation and multi-agent deep reinforcement learning, to help co-construct urban policies and integrate all stakeholders in the decision process. Smart Cities provide sensor data from the urban areas to increase realism of the simulation in SmartGov. Our first contribution is a generic architecture for multi-agent simulation of the city to study global behavior emergence with realistic agents reacting to political decisions. With a multi-level modeling and a coupling of different dynamics, our tool learns environment specificities and suggests relevant policies. Our second contribution improves autonomy and adaptation of the decision function with multi-agent, multi-level reinforcement learning. A set of clustered agents is distributed over the studied area to learn local specificities without any prior on the environment. Trust score assignment and individual rewards help reduce non-stationary impact on experience replay in deep reinforcement learning. These contributions bring forth a complete system to co-construct urban policies in the Smart City. We compare our model with different approaches from the literature on a parking fee policy to display the benefits and limits of our contributions.

Keywords

Multi-agent simulation ; Multi-agent deep reinforcement learning ; Dynamic coupling ;

Remerciements

Le travail effectué pendant la thèse n'est pas mon unique production, mais résulte d'une continuité d'échanges, d'interactions, de conseils et de doutes pour en faire émerger une direction globale jusqu'à l'aboutissement de ce manuscrit de thèse. Ceux qui me connaissent savent que je suis concis et peu expressif, cependant je souhaite faire une petite exception afin de profiter des pages suivantes pour remercier les personnes qui m'ont accompagné dans cette expérience et qui m'ont, à travers les moments que nous avons partagés, permis de réaliser une synthèse de ma recherche dans la thèse que vous avez entre vos mains.

Une mention spéciale à deux entités : d'une part l'Allocation Doctorale de Recherche de la Région Auvergne-Rhône-Alpes ARC 7, qui grâce à leur financement a rendu possible ces trois premières années de thèse ; et d'autre part, Grenoble INP de m'avoir accueilli et financé en ATER pour la dernière année de ma thèse.

En premier lieu, je tiens tout particulièrement à remercier mon équipe encadrante : Salima Hassas, ma directrice de thèse, tu m'as confié suffisamment de liberté pour explorer les sujets qui me tenaient à cœur, pour les défendre et d'aller au bout de mes propositions. Tu m'as toujours dit que j'étais le *chef de projet* de ma thèse, mais j'ai à un certain moment ressenti le tournant dans la thèse entre le doctorant suiveur de propositions et le doctorant acteur, effectivement meneur du projet de sa thèse, avec le gain d'autonomie sur le sujet, des réflexions plus profondes et une meilleure prise de position. Véronique Deslandres, tu as toujours cru en notre approche et cherché à pousser le projet plus loin. Tu m'as conseillé dans mes moments de doutes. C'est grâce à ton investissement que plusieurs stagiaires nous ont rejoints pour participer au développement de notre modèle. Vassilissa Lehoux, ton regard différent sur les problématiques de la thèse était très utile pour prendre du recul sur notre approche et sur la faisabilité de nos idées. Merci également pour tes nombreux et exhaustifs retours et pour ton encadrement. Merci aussi pour ces pauses le jeudi midi où tu m'as fait découvrir d'excellents jeux de plateau. Cela n'a pas été toujours facile d'être partagé entre le LIRIS et NAVER LABS Europe (anciennement XRCE), mais je ne regrette pas un moment cette coopération. Merci à vous trois de m'avoir encadré et accompagné dans cette expérience, d'avoir été là pour me conseiller et m'orienter au long de ces années de thèse.

Je remercie mes deux rapporteurs de thèse Vincent Chevrier et Stéphane Galland pour la relecture et

l'évaluation de mon manuscrit de thèse ainsi que mes examinateurs de thèse Frédéric Amblard, Éric Duchêne, Julie Dugdale pour leurs questions et leurs remarques lors de ma soutenance de thèse.

Je remercie l'équipe HAwAI du LIG de m'avoir accueilli pendant mon année d'ATER à Grenoble et j'en profite pour remercier également mes différents collègues ATER et de l'IMAG.

Une pensée également pour l'équipe de Vassilissa qui m'a accueilli d'abord au sein du XRCE puis à NAVER LABS. Je remercie mon collègue doctorant Lizhi pour les journées de l'ARC et nos nombreuses discussions sur nos problématiques.

Merci également aux trois stagiaires de Master 2 et d'écoles d'ingénieurs Manon, Manon et Paul. Votre collaboration et vos retours ont amélioré notre modèle et permis de tester la validité de notre approche sur des contextes différents.

Je remercie également l'ensemble de mes collègues et amis doctorants (maintenant docteurs) que j'ai eu la chance de côtoyer au LIRIS. Merci à vous pour tous ces années, ces découvertes, le temps passé ensemble en dehors de la thèse comme en randonnée et en escalade ainsi que les nombreuses parties de jeux de plateau et jeux de rôles. Une pensée particulière pour Yohann et ta merveilleuse installation de Arch sur ma machine. Samuel, notre roi de l'optimisation et du macaron, pour ta contribution dans mon modèle avec tes voitures volantes. Thomas, j'ai toujours été surpris par ta capacité à faire tellement de choses dans des journées de 24h, j'y crois, un jour on le fera ce jeu ! Thibault pour m'avoir supporté quand je te demandais de me ramener, nos trajets en voiture et nos discussions vont me manquer. Alexis pour nos échanges, sur le monde que nous voulions refaire, pour ta vision de la recherche et le bureau que nous avons partagé pendant quelques temps. Arthur, notre guide des cendres au destin tragique, pour nos discussions en rentrant du travail quand on prenait le métro. Antoine W. pour notre intérêt commun dans les jeux de plateaux et vidéos, pour nos espoirs de se faire un cube un jour. Rémi pour ton aide sur la mise en place du serveur, pour l'amitié que l'on a construit au fil des années depuis que l'on se connaît. Maxime pour m'avoir fait découvrir de nouveaux jeux et surtout pour m'en avoir fait acheté plus que nécessaire, ce trajet en vélo Lyon-Grenoble que l'on doit se faire, merci pour tout ce que tu m'as fait découvrir pendant notre année en collocation et ce que tu m'as apporté en tant qu'individu. Antoine D. pour toutes nos parties de Terraforming Mars après le travail, pour ton chat des enfers qui nous réveillait le matin et pour le coup de main administratif lors de la thèse.

Je remercie Jonathan, mon compagnon de voyage pour les différentes conférences. Merci pour tes conseils et nos discussions, les pointeurs sur de nouvelles pistes de recherche, et surtout d'avoir pris le temps de me faire des retours et corrections sur ce que je t'envoyais. D'avoir pris le temps de m'aider sur la formalisation de certains problèmes. Et pour toutes nos parties de jeux de plateau.

Enfin, je tiens à remercier mes parents qui m'ont permis d'entreprendre ce que je voulais faire et m'ont donné les clés pour y arriver et qui m'ont toujours soutenu. Ma sœur pour nos discussions excentriques et tes recommandations pour de nouvelles séries à regarder. Et ma famille d'avoir été présente à mes côtés depuis tout ce temps.

Pour finir, je tiens à remercier ma compagne, la personne de ma vie, Mégane pour toutes nos années ensemble. De t'avoir eu à mes côtés, de ton soutien dans les périodes de doutes et les moments difficiles. Pour nos discussions sur les éventuels parallèles entre nos sujets de recherche respectifs. Pour tous nos fous rires, pour tous ces petits moments qui font que chaque moment à tes côtés est unique, je te remercie. La thèse n'aurait pas été la même sans toi. Merci pour tout.

A vous tous, et ceux que j'aurais oublié, et à toi lecteur, merci !

Bonne lecture.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction

La campagne de vaccination contre le virus H1N1 de 2009 en France, ou plus récemment la proposition de la loi sur la taxe carbone, sont des exemples de politiques qui sont mal reçues par une partie des acteurs concernés, notamment en raison de l'incompréhension ou de la non-pertinence des dépenses engendrées par ces politiques. Tout le monde a une opinion sur une politique existante : ce qui est satisfaisant, ce qui pourrait être amélioré, ce qui devrait être supprimé. Chacun souhaite ainsi non seulement pouvoir exprimer son avis, mais également que celui-ci soit pris en compte dans les processus politiques. Une politique peut aussi bien concerner un ensemble restreint d'acteurs, comme une tranche démographique, ou bien impacter un grand nombre d'individus (l'ensemble de la population d'une ville par exemple). Imaginons un monde où chaque personne pourrait facilement exprimer son avis. Cela simplifierait-il la conception de politiques ? Le choix des personnes à satisfaire est toujours délicat, et il est donc nécessaire de faire des compromis. À l'heure de la *Smart City*, du *Big Data* et du citoyen connecté, exprimer son avis n'a jamais été aussi simple. Dans ce contexte, comment les décideurs peuvent-ils utiliser facilement ces retours ? En effet, les retours des parties prenantes sont complexes à obtenir : difficulté de représenter le problème politique, faux avis (lobbyistes) ou avis orientés, biais dans les réponses, etc. Ces retours usagers, récoltés par des applications dédiées, peuvent être utilisés pour construire des politiques urbaines, comme par exemple des politiques d'urbanismes et d'aménagement du territoire, de mobilité avec l'extension d'un réseau de tram, d'environnement en favorisant les espaces verts. Ces politiques urbaines forment un écosystème complexe au sein duquel elles sont interdépendantes et ne peuvent donc pas être traitées indépendamment les unes des autres. Dans ce cadre, les décideurs ont besoins d'outils adaptés permettant de guider leur prise de décision. Ce besoin est renforcé par le récent transfert de compétences de l'État à destination des villes, ce qui leur confère davantage de responsabilités de décision, sans qu'elles ne soient pour autant préparées à y répondre. De plus, l'extension à l'échelle territoriale de la zone urbaine modifie le paysage urbain, avec des regroupements en intercommunalités, ce qui engendre une prolifération de nouveaux acteurs politiques.

Ce travail de thèse n'a pas pour objectif de produire la meilleure politique possible, mais plutôt de s'intéresser à la question suivante : si le décideur dispose de données décrivant les acteurs concernés de la

zone urbaine considérée, comment peut-il construire une politique urbaine satisfaisant ses objectifs ? Ou de manière plus générale, s'il souhaite apporter une modification particulière à une politique donnée, est-ce que celle-ci sera bien reçue par les différents acteurs concernés ? Les données du citoyen connecté, obtenues de manière active sur les réseaux sociaux, ou de manière passive avec les données d'usages et de déplacements, permet d'identifier la dynamique des usagers dans un contexte urbain. Le décideur aurait besoin d'un outil, s'inscrivant dans le cadre de la *Smart City*, capable non seulement d'assister le décideur pour produire et modifier des politiques, mais également d'utiliser et d'intégrer des retours dans la boucle de conception politique. L'analyse des effets d'une politique par les retours exprimés par les citoyens faciliterait l'évaluation des actions politiques appliquées. Cet outil proposerait alors un ensemble de politiques envisageables, que le décideur politique pourrait ajuster avant de réaliser des simulations permettant leur évaluation. Ce type de construction est appelée *co-construction*, et repose sur un échange entre le concepteur et l'outil d'aide à la décision, qui permet de structurer et d'éprouver un ensemble de décisions politiques.

Actuellement, les décideurs disposent de différentes méthodes pour estimer l'impact d'une politique urbaine. L'une d'entre elles est la simulation, véritable bac à sable permettant de tester différentes décisions politiques et d'en observer les conséquences sur un environnement contrôlé, afin de guider les mesures à mettre en place dans la ville réelle. La simulation multi-agents orientée individu permet en particulier de représenter les comportements humains et leurs interactions. Les populations d'agents peuvent refléter les sondages d'opinion du moment par exemple. La phase d'élaboration d'une politique repose sur son adaptation aux évolutions observées dans l'environnement suite à l'application d'actions politiques. Elle dépend également de l'horizon de temps de la politique. C'est-à-dire que pour atteindre un même objectif, la séquence d'actions risque d'être différente selon que la politique est appliquée à court ou à long terme. Sur le long terme, il est nécessaire d'ajuster les mesures prises afin de prendre en compte les évolutions de l'environnement qui apparaissent. Cependant, trouver les bons indicateurs pour améliorer la politique peut prendre du temps, et il est nécessaire de réaliser différentes expérimentations afin de bien comprendre les effets observés d'une politique. Une solution pour simplifier le processus de création repose sur l'auto-adaptation de la politique à l'aide d'un apprentissage automatique, et ce avec ou sans connaissances expertes. Par conséquent, l'objectif de cette thèse est d'élaborer une preuve de concept pour un outil générique d'aide à la décision, capable de s'auto-adapter à la construction du contexte politique par le décideur, pour l'assister dans la co-construction de politiques urbaines. La co-construction est utilisée aux différentes étapes de la conception par le décideur, afin de modéliser dans un premier temps la zone urbaine qu'il souhaite étudier, puis d'évaluer les politiques urbaines auto-adaptées proposées par l'outil dans un second temps. De cette façon, les allers-retours entre l'outil et le décideur permettent d'ajuster les actions politiques pour satisfaire les objectifs visés. La co-construction, envisagée entre le décideur et l'outil, peut également être un support de discussion pour le décideur lorsqu'il suggère des modifications aux utilisateurs. Cette co-construction permet également aux décideur de modifier directement les paramètres de l'outil, ce qui modifie sa perspective initiale pour produire de nouvelles politiques urbaines plus adaptées à la situation réelle.

1.1 Contexte

L'urbanisation de la société est en constante croissance. Les villes s'étendent rapidement et deviennent de plus en plus complexes. Le décideur politique doit prendre en compte les changements rapides qui interviennent dans la zone urbaine pour établir des politiques pertinentes. Les approches classiques de conception de politique urbaine (*top-down*) partent généralement d'un problème politique identifié par les concepteurs. Cependant, ces approches sont limitées par la difficulté de prendre en compte toutes les données disponibles, mais aussi d'intégrer l'utilisateur, qui intervient souvent à la dernière étape du processus. Pourtant, avec les moyens de communication actuels, l'utilisateur pourrait s'exprimer facilement dès l'élaboration initiale de la politique en question. La *Smart City* est présentée comme une zone urbaine connectée, disposant d'infrastructures et de capteurs pour collecter toutes sortes de données (d'usages, de pollution, etc.) et ainsi gérer la collectivité et ses ressources de manière plus efficace. Cette thèse s'inscrit dans l'effort réalisé autour de la *Smart City*.

La récente intégration des données ouvertes et l'explosion des objets connectés dans les *Smart Cities* modifient le paysage des politiques urbaines. Les données sont désormais plus facilement accessibles et utilisables, et peuvent être prises en compte pour structurer la politique. De plus, le récent virage sociétal urbain conduit à intégrer les usagers dans la conception même des politiques, à la fois pour connaitre leurs retours sur l'utilisation des réseaux et de l'infrastructure, mais aussi pour évaluer leurs besoins. Les différentes thématiques abordées dans la thèse sont décrites dans cette section.

1.1.1 Politique urbaine et décision politique

La politique urbaine peut être soit d'origine publique (représentées par des institutions étatiques décrites par Dewey [65] et à destination des citoyens qui votent), soit d'origine privée. Par exemple des entreprises comme Google investissent l'espace publique en instaurant et en régulant l'affichage publicitaire dans certaines villes américaines. On peut aussi citer les *town square* d'Apple, véritables aménagements de la ville qui ressemblent à des espaces publics tout en étant privés et à vocation commerciale.

Intéressons-nous plus précisément à la notion de politique urbaine. Walker propose dans les années 2000 la définition suivante du mot politique [250] :

Définition 1 : Les politiques sont l'ensemble des dispositions, applicables par les acteurs du domaine politique, qui affectent les structures et les performances d'un système. De manière générale, une politique est un ensemble d'actions effectuées par une entité (publique ou privée) pour contrôler le système, aider à résoudre les problèmes identifiés, ou visant un certain nombre de bénéfices.

Dans la définition que nous venons d'aborder, le *système* représente la cible d'une politique donnée. Dans le cadre de la politique urbaine, cette cible peut être une communauté de communes, une ville ou un quartier, mais aussi un élément de la ville : le réseau de transports en commun, par exemple. Cette définition générale ne traduit toutefois pas la notion d'adaptabilité d'une politique. Comme nous l'avons vu précédemment, il est complexe de planifier les mesures à prendre dans un futur incertain, et par conséquent il serait nécessaire de considérer des politiques adaptatives, capables de rester pertinentes même après des changements non envisagés. Nous proposons ainsi notre définition de la politique urbaine :

Définition 2 : Une politique est un ensemble d'actions effectuées par une entité (publique ou privée) afin de réguler le système étudié et de faciliter la résolution des problèmes identifiés. Cette politique s'adapterait donc en continu par un ajustement des actions préconisées, pour atteindre des bénéfices supérieurs aux coûts de mise en place.

Nous avons abordé en introduction le fait que l'application d'une politique est une tâche non triviale pour les décideurs, et ce pour plusieurs raisons. D'une part, le processus de conception d'une politique urbaine implique un certain nombre d'acteurs (publics et privés). Ainsi, l'élaboration d'une politique va de l'identification du problème à la réflexion de la pertinence des solutions choisies, en passant par la mise en application et par l'analyse des effets et des coûts générés (directs et différés) par cette politique. D'autre part, une politique donnée impacte nécessairement les usagers en modifiant leurs habitudes et leur façon d'interagir avec l'environnement. Afin d'augmenter les chances qu'une politique donnée soit acceptée, les acteurs doivent proposer une politique adaptée à la majorité des usagers concernés. Pour ces raisons, la conception d'une politique nécessite de prendre en compte l'ensemble de tous les paramètres que nous venons d'aborder.

1.1.2 Intérêt de la simulation

L'avantage de la simulation est de pouvoir tester des décisions politiques, et d'en observer les effets, sans avoir à les appliquer dans la réalité. Cette approche est généralement appréciée car elle permet d'éviter la perte de temps, et potentiellement d'argent, liée à l'application d'une mauvaise politique. Cependant, le problème posé par la simulation est sa pertinence par rapport à une situation réelle, qui conditionne la validité des observations effectuées au sein de cette simulation. Dans le cadre de la représentation d'acteurs humains, la simulation multi-agents orientée individu offre la possibilité d'avoir des comportements crédibles et réalistes. La crédibilité est nécessaire pour une meilleure acceptation des résultats de l'outil, et le réalisme est nécessaire pour l'obtention de meilleurs résultats à l'issue de la simulation. En utilisant les données recueillies au sein de la Smart City, on pourrait ainsi générer des populations synthétiques plus hétérogènes et réalistes Appliquée aux systèmes complexes, la simulation de politiques urbaines offrirait la possibilité de considérer un écosystème à part entière. Les systèmes complexes intègrent des sous-systèmes gérés par des acteurs publics (réseau de pistes cyclables, de voies routières, emplacements des parking) ou bien par des acteurs privés (compagnies de taxis, véhicules autonomes, boîtes logistiques urbaines pour les livraisons, etc.).

1.1.3 Adaptation aux changements de la société

La construction d'une politique urbaine nécessite, à défaut de pouvoir prédire l'avenir, d'anticiper une partie des évolutions futures. Le recours à l'apprentissage automatique est ainsi particulièrement adapté pour accroître l'autonomie et l'adaptabilité du processus décisionnel. Son utilisation est donc envisagée dans la construction de politiques, pour apprendre dans un premier temps les spécificités locales de l'environnement, puis dans un second temps pour utiliser cet apprentissage afin d' assister le décideur en lui suggérant des actions possibles sur le système. Cependant, il existe différentes manières d'exploiter l'apprentissage. Quelles sont-elles dans notre cadre ? Que doit apprendre le système ? Quel serait l'objectif de l'apprentissage ?

En plus de ces questions, la conception d'une politique urbaine satisfaisante requiert d'identifier les particularités de la zone urbaine étudiée. Par exemple, les mesures spécifiques à appliquer peuvent dépendre du quartier (résidentiel ou commercial), ou du cadre de vie (activités de jours ou vie de nuit). La conception d'un outil d'aide à la décision complet pour créer des politiques adaptatives devrait donc être capable de proposer un découpage pertinent de l'environnement, permettant de cibler les actions politiques sur le territoire.

L'utilisation de l'intelligence artificielle, en parallèle avec les données fournies par la *Smart City*, permet donc de répondre aux difficultés et besoins identifiés précédemment pour l'élaboration de politiques urbaines, notamment en impliquant le décideur dans un processus de co-construction des politiques. Nous souhaitons dans ce travail proposer une preuve de concept d'un outil d'aide à la décision assez générique, permettant de satisfaire un ensemble d'objectifs spécifiés par les utilisateurs (décideurs, acteurs de la ville) afin de produire des politiques urbaines spécifiques à différents domaines (mobilité, alimentation en électricité ou en fibre optique, canalisations d'eau, etc.).

1.2 Plan de ce manuscrit

Ce travail de thèse se trouve au croisement de plusieurs domaines de recherche : la conception de politiques, la simulation multi-agents et l'apprentissage automatique (figure 1.1).

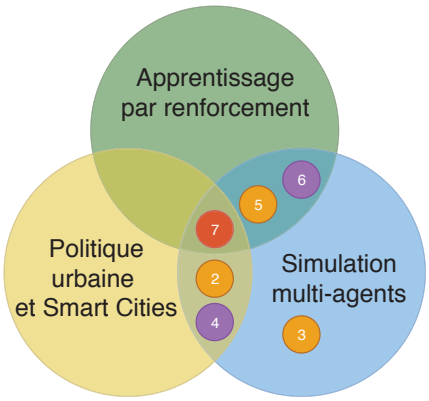


FIGURE 1.1 – La thèse regroupe ces trois thématiques. Chaque cercle numéroté correspond au chapitre associé, et illustre les thématiques qui y sont abordées. Les cercles orange sont des chapitres d'état de l'art, les cercles violets sont des contributions, et le cercle rouge se réfère aux expérimentations.

Les travaux de cette thèse portent sur l'adaptation des systèmes à des environnements inconnus et leur réaction lorsque ceux-ci évoluent. Un outil d'aide à la décision adaptatif doit être capable de considérer des problématiques diverses et variées. Ainsi, avec la même architecture, nous souhaitons que le décideur puisse produire différentes politiques de mobilité urbaine. Par exemple, le décideur pourrait évaluer l'impact de la création d'un parking relais en périphérie de la ville sur le comportement des travailleurs en voiture,

tout comme l'impact de la création d'une nouvelle ligne de métro sur le déplacement des usagers. D'autre part, nous souhaitons que notre outil puisse également permettre au décideur de gérer des problématiques d'aménagement urbain, comme l'alimentation des bâtiments en eau potable ou encore l'étude de la distribution de la fibre optique sur les prix de l'immobilier.

L'objectif de cette thèse est donc de proposer une architecture générique avec un outil d'aide à la décision assistant le concepteur dans la phase de construction d'une politique urbaine. Notre modèle contient tous les éléments nécessaires pour offrir un cadre d'étude réaliste permettant d'éprouver des problématiques urbaines variées. Cette thèse apporte ainsi deux contribution principales : le modèle générique SmartGov, avec un couplage de simulations multi-agents, et l'algorithme d'apprentissage ClusteredDeep Q-Network (CDQN), conduisant à l'organisation dynamique de groupes d'agents politiques adaptatifs.

Ce mémoire de thèse est constitué de sept chapitres répartis de la manière suivante :

La première partie présente la *Smart City* et la simulation multi-agents pour introduire notre architecture SmartGov, et contient trois chapitres :

— **Chapitre 2 – L'impact des *Smart Cities* sur le paysage politique.** Ce chapitre introduit un état de l'art sur les approches existantes de conception de politiques, et sur le futur de ces conceptions dans un cadre dynamique. L'apport de la *Smart City* et sa place dans le nouveau paysage de la conception politique est abordé. Dans ce Chapitre, nous nous intéresserons également à la prise de décision sous incertitude pour la construction de politiques.

— **Chapitre 3 – L'Intelligence Artificielle Distribuée au service de la conception de politiques** Ce chapitre d'état de l'art présente les possibilités offertes par l'intelligence artificielle distribuée et les approches multi-agents pour la modélisation d'une zone urbaine évoluant dynamiquement. La ville étant un système complexe où coexistent plusieurs échelles, sa représentation introduit des problématiques multi-échelles. Nous aborderons ensuite les niveaux de simulation et le couplage de dynamiques microscopiques et macroscopiques.

— **Chapitre 4 – Conception de politiques urbaines avec SmartGov.** Ce chapitre présente la première contribution de notre travail : l'architecture générique SmartGov. En couplant deux simulateurs multi-agents, nous proposons une description formelle des éléments constituant notre modèle. Dans ce chapitre, nous instancions notre modèle générique pour illustrer le processus de construction d'un environnement réaliste pour élaborer une politique urbaine.

La seconde partie s'intéresse à la problématique d'apprentissage automatique pour l'adaptation dynamique des politiques :

— **Chapitre 5 – Apprentissage par renforcement individualisé et décentralisé dans un environnement multi-agent non stationnaire.** Ce troisième chapitre d'état de l'art présente les verrous, identifiés par la communauté de l'apprentissage par renforcement multi-agents, concernant l'apprentissage d'agents autonomes dépourvus de moyens directs de communication.

— **Chapitre 6 – Une nouvelle méthode d'apprentissage multi-agents pour la coordination d'agents sans communication.** Ce chapitre introduit la seconde contribution de cette thèse avec la proposition d'un algorithme d'apprentissage multi-agents et multi-niveaux, le ClusteredDeep Q-Network

(CDQN), permettant de réaliser un découpage intelligent de l'environnement avec une gestion dynamique de clusters d'agents politiques.

Enfin, la troisième et dernière partie de ce manuscrit combine nos deux contributions et propose des expérimentations basées sur notre architecture SmartGov.

— **Chapitre 7 – Co-construction d'une politique urbaine avec SmartGov.** Ce chapitre montre comment l'architecture formelle SmartGov peut être instanciée sur un cas politique concret, et explique comment l'utilisation de la couche décisionnelle multi-agents autonome et adaptative assiste le décideur dans l'élaboration de politiques urbaines.

Quelques réflexions sur la preuve de concept développée sont enfin présentées en guise de conclusion. Nous aborderons également les perspectives d'amélioration envisagées pour le futur.

Chapitre 2

L'impact des *Smart Cities* sur le paysage politique

Sommaire

2.1 Introduction

Dans le cadre de cette thèse, une politique urbaine représente la volonté d'agir sur un système complexe pour proposer une solution à un problème identifié ou satisfaire les besoins d'acteurs, publics ou privés. La mise en place d'une telle politique structure la société en lui imprimant une direction particulière, répondant à des questions d'intérêt général soulevés par les décideurs politiques. Ce chapitre n'a pas vocation à présenter un état de l'art exhaustif sur la conception de politique mais de s'intéresser à un aspect particulier dans l'élaboration des politiques urbaines : le nouveau rôle de la Smart City, productrice de données, et son impact sur le processus même de conception des politiques. Nous invitons le lecteur intéressé par la conception des politiques à parcourir des ouvrages comme ceux de Bardach [14] ou Birkland [25] qui décrivent les différentes étapes du processus politique. Dans notre discours, le *décideur* ou *décideur politique* est une entité publique ou privée cherchant à proposer une politique urbaine. Les parties prenantes, appelés *stakeholders* en anglais, représentent à la fois les citoyens et les différents acteurs concernés par la politique. Les décideurs peuvent faire partie de ces parties prenantes.

L'émergence de la Smart City implique de nouvelles manières de concevoir des politiques. Sur la prise de décision notamment, l'intégration des parties prenantes et du citoyen dans la boucle de conception est facilitée, ainsi que de la collecte automatisée des données. Nous considérons que la formalisation du problème intervient en entrée de l'outil pour structurer la direction que prendra la politique urbaine construite. Pour notre outil d'aide à la décision, nous nous positionnons donc uniquement sur l'aspect élaboration d'une politique correspondant à l'identification des actions politiques clés, qui peuvent être mises en places. Avec l'émergence de la Smart City et des citoyens connectés, l'étape de consultation est d'autant plus intéressante puisque l'on observe, depuis maintenant quelques années, que le paysage politique se transforme pour intégrer davantage le citoyen dans le processus de conception. Ainsi trois problématiques ont motivé les choix effectués pour les travaux de cette thèse :

— la difficulté de créer une politique dans un futur incertain ;
— l'intégration des données nécessaires pour construire une politique ;
— l'utilisation du retour des citoyens pour affiner la politique urbaine.

Dans un premier temps, nous nous intéressons à l'incertitude correspondant aux aspects d'identification et de formulation du problème dans le cycle d'élaboration d'une politique. Ensuite, nous proposons des politiques urbaines envisageables pour élaborer des politiques sous incertitude et nous abordons brièvement l'implémentation et l'évaluation des résultats. Enfin, l'essor des *Smart Cities* ainsi que la place croissante de l'usager dans la prise de décision politique sont présentés, qui dessinent le socle de notre outil. En conclusion, nous proposons une ouverture sur la prise de décision combinée à l'intelligence artificielle motivant notre travail pour la conception de politiques urbaines.

2.2 Une planification sous incertitude

Cette section s'intéresse à la prise de décision sous incertitude, c'est-à-dire lorsque les probabilités sur le résultat ne sont ni connues, ni même connaissables, par opposition au risque, où les probabilités sur le

résultat sont connues [134].

La planification sous incertitude est découpée en deux catégories : l'une liée à la nature humaine du décideur et ses limites cognitives et l'autre repose sur la difficulté de prédire l'avenir et donc d'envisager des politiques sur le long-terme.

2.2.1 Inhérente à la nature du décideur

Cette partie s'intéresse aux études effectuées sur la cognition et les limites connues de la prise de décision. Pour le décideur, les limitations généralement identifiées sont liées à l'évolution de l'aide à la décision [237].

John von Neumann et Oskar Morgenstern présentent, en 1944, l'utilité espérée comme fondement mathématique pour la prise de décision économique [249]. Dans leur approche ils considèrent, comme pour la majorité des travaux de cette époque, que les décideurs sont rationnels. Cependant, la rationalité limitée [222], introduite par Herbert Simon en 1947, considère que, confronté à un choix, le décideur politique adopte un comportement rationnel, mais que celui-ci est limité en termes de capacité cognitive et des informations disponibles sur le problème étudié. Ainsi le décideur cherche plutôt à produire une solution satisfaisante plutôt qu'une solution optimale.

L'accumulation d'informations présente deux problèmes. D'une part, c'est un obstacle pour aider le décideur dans sa prise de décision. En effet, certaines sources d'informations comme l'opinion publique, les débats ou encore les sondages sont délicates à exploiter puisqu'elles sont subjectives et exprimées par des acteurs non experts. D'autres part ces informations peuvent être difficiles à recueillir et réduisent le temps dont dispose le décideur politique pour prendre sa décision ou déterminer d'autres options possibles [223, 121]. Il est par conséquent impossible de considérer que les décideurs disposent de la totalité des informations pour prendre des décisions dans un monde complexe [156].

En 1950, le théorème d'impossibilité d'Arrow se place dans un contexte où plusieurs décideurs doivent ensemble choisir une politique. Il suppose que chaque décideur ne pourra exprimer ses préférences que par la comparaison deux à deux des différentes solutions. Le théorème indique qu'il n'existe pas de processus de choix qui permette de créer une hiérarchie des préférences cohérente à partir de l'agrégation des préférences individuelles exprimées par chacun des membres d'une même collectivité [9]. Ce résultat justifie la difficulté rencontrée de chercher à satisfaire tous les acteurs engagés dans le processus de décision.

Un autre problème de planification repose sur l'agenda politiques des décideurs. En effet, ils ont tendance à attribuer généralement plus d'importance aux actions à court terme qu'aux bénéfices à long-terme du fait de considérations électorales [146].

De plus, même en planifiant correctement une politique, le décideur est limité par sa propre vision du monde. La myopie cognitive intervient lorsque les décideurs politiques ignorent certains aspects du problème, de manière involontaire, en se concentrant uniquement sur un nombre limité d'alternatives ou sur une vision restreinte et agrégée de l'optimalité du problème [103]. À différencier de l'hystérésis cognitive comme la tendance à rester sur sa décision [178] par exemple avec le besoin de se renseigner uniquement pour confirmer sa décision.

Certaines limites cognitives ne peuvent pas être résolues avec notre outil d'aide à la décision. Cependant,

dans le cadre de notre thèse, nous cherchons à atténuer les problèmes liés à l'exploitation des données pour le décideur grâce à un outil autonome et l'apport de la Smart City.

2.2.2 Inhérente à la nature incertaine du futur

La nature incertaine du futur, comme les évènements exogènes, complique la prise de décision pertinente pour concevoir des politiques urbaines. L'anticipation de ces évènements exogènes possibles :

- économiques : coûts et capacité d'interventions sur les politiques ;
- environnementaux : impact de la pollution, changement de direction dans les lois sur le climat et le réchauffement climatique ;
- démographiques : direction de son l'évolution ;
- nouvelle génération de décideurs politiques : accès à l'information, nouveaux outils.

est requise pour la planification sur le long-terme [146]. De plus, les décideurs éprouvent des difficultés à traduire leurs besoins en actions pertinentes lorsqu'ils en évaluent les effets à long terme.

Dans leur ouvrage, Lempert *et al.* [146] s'intéressent à la planification sur le long-terme (*long-term policy analysis*, LTPA), afin de produire des politiques sur plusieurs dizaines d'années. Cela consiste à étudier l'élaboration de politiques en considérant d'une part que les informations dont dispose le décideur politique ne sont pas exhaustives mais d'autre part, que le futur est incertain. Avec les points abordés précédemment, notamment l'apparition d'évènements exogènes, il semble toutefois difficile de garantir que les politiques seront toujours pertinentes à $t + \delta t$. Bien que l'objectif de la planification sous incertitude soit de capter les tendances qui peuvent survenir dans le futur pour envisager un ensemble de scénarios plausibles, il est complexe de simuler l'imprévisible.

2.2.3 Anticipation du futur

Nous faisons le choix de prendre en compte l'incertitude avec une construction "continue" de la politique, où l'analyse des données disponibles peut faire apparaitre les évènements exogènes que l'on peut du coup intégrer, ce qui nous permet ensuite d'ajuster la politique mise en place. Notre hypothèse est que l'accessibilité aux données offerte par la Smart City permet d'identifier plus facilement des déviations autour de la politique actuelle et qu'elles seront naturellement captées par la métropole. Ainsi, il sera possible de suivre une approche dynamique des politiques, avec des scénarios d'actions plus pertinents, capables de s'adapter aux problèmes susceptibles d'émerger dans le futur. La construction de politiques de manière continue est possible si l'outil d'aide à la décision est suffisamment autonome et adaptatif pour intégrer des changements dans la description du monde. Si c'est le cas, il est alors envisageable de faire des projections de politiques en continu et de les ajuster au fur et à mesure que les nouvelles données arrivent. On peut par exemple imaginer que les données de la Smart City nourriront des algorithmes de prédiction plus performants qui permettront de mieux anticiper les événements. Nous considérons d'autre part que l'ajustement fréquent et régulier de politiques fait que l'impact des futurs évènements exogènes seront réduits dans la mesure où les problèmes seront identifiés et corrigés plus rapidement.

2.3 Types de politiques

Cette section introduit les différents types de politiques qui peuvent nous intéresser dans la construction d'un outil d'aide à la décision.

2.3.1 Politique incitative

Une politique est incitative lorsque son objectif est d'orienter le comportement des individus. Schneider et Ingram [214] considèrent même qu'une politique cherche presque toujours à modifier le comportement des usagers en leur permettant de faire des choses qu'ils n'auraient pas fait autrement. Les travaux de Kahneman *et al.* [124, 123] sur la *peur de la perte* (*loss aversion*) montrent que les gens accordent plus d'importance à la perte monétaire qu'à un profit de valeur égale. Tyler a une vision optimiste du comportement de l'usager puisqu'il considère, dans son ouvrage sur l'obéissance aux règles et à la politique, que si la loi fait sens aux yeux des usagers et que ceux-ci l'acceptent, alors ils ne vont pas chercher à enfreindre la loi, même si cela est possible sans avoir de conséquence [241]. Les travaux de Holst [107] concluent que l'utilisation de pénalités est généralement d'avantage considérée pour faire appliquer une politique, par exemple avec l'augmentation des taxes sur la cigarette ou l'essence pour que leurs consommations diminuent.

Il nous apparait difficile d'éprouver des politiques incitatives directement dans le cadre de cette thèse puisque cela nécessite de calibrer des éléments tels que la tendance à l'obéissance, l'aversion de la perte, ou la réaction à l'augmentation des taxes pour les différents acteurs. En effet, même si ceux-ci expriment plus aisément leurs avis, leurs retours sur une politique incitative seront rarement positifs puisqu'elle a plutôt tendance à forcer un changement de comportements (l'augmentation d'une taxe sur la tabac pour inciter à diminuer sa consommation est généralement peu appréciée des fumeurs). Il semble irréalisable de disposer de sondages exhaustifs capables de comprendre la réaction des acteurs par rapport à ces problématiques. Par conséquent, nous nous intéressons aux politiques qui s'adaptent aux modifications identifiées dans l'environnement sans contraindre directement les acteurs.

2.3.2 Politiques robustes et adaptatives

Une politique proposant une séquence d'actions dans un environnement incertain et proposant une solution marchant dans la majorité d'entre eux est appelée *politique robuste* statique [157]. Une politique robuste n'est donc pas nécessairement optimale pour le futur considéré le plus probable, dans la mesure où elle doit être performante pour un ensemble de futurs envisagés pour la prise de décision.

Bien que le terme de prise de décision robuste soit régulièrement utilisé pour aborder la prise de décision sous incertitude [146], peu d'experts utilisent directement la notion de politique robuste, lui préférant la notion de *politique adaptative* [251, 13]. Une politique adaptative est décrite comme une politique intégrant plusieurs combinaisons d'actions politiques ainsi que la temporalité pour savoir quand effectuer une action. Certaines actions doivent être implémentées directement, d'autres le seront à un moment indéterminé dans le futur. Il est également possible que certaines actions ne soient jamais implémentées si les conditions ne sont pas réunies. La politique contient également des plans d'urgences ainsi que des spécifications d'applications

dans le cas où toute la politique doit être reconsidérée. Ainsi la prise en compte d'un futur difficile à prévoir nécessite de reconsidérer la politique comme une entité à part entière. La politique comme entité implique de détailler ses objectifs, ses paramètres de réglages, son horizon de temps, soit autant d'informations pertinentes pour simuler la politique comme un objet concret pouvant être régulé et ajusté. Les approches adaptatives permettent ainsi aux décideurs politiques de mieux gérer les incertitudes auxquelles ils sont confrontés lors de la création de politiques, ces dernières pouvant être ajustées avec le temps.

2.3.3 Vers la politique continue et évolutive ?

Un verrou de la conception d'une politique urbaine est la prise en compte des changements de la société et de l'évolution de la zone urbaine [202]. La politique adaptative crée une politique à un instant t en proposant des variations pour les futurs envisageables (voir figure 2.1). Dans cette thèse, nous considérerons la politique continue et évolutive, qui utilise la disponibilité accrue des données pour s'adapter en permanence aux modifications observées et aux évènements exogènes émergeant dans l'environnement.

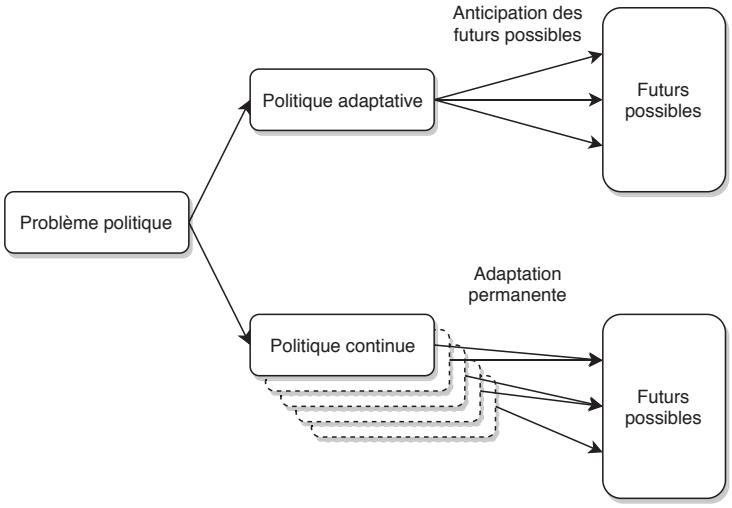


FIGURE 2.1 – Représentation de la politique adaptative par rapport à la politique continue qui s'adapte en permanence aux modifications observées.

Notre outil d'aide à la décision, plutôt que d'envisager les futurs possibles, construit une politique avec les données actuelles dont il dispose. Si cette politique est acceptée par le décideur et mise en place, alors le futur lié à cette politique sera capté avec les nouvelles données acquises.

Nous avons abordé différents types de politiques dans cette section, notamment des travaux sur les politiques robustes et la planification sur le long-terme qui semble être difficilement applicable en l'état. En effet, le futur est par nature incertain et prévoir une politique robuste prenant en compte un nombre conséquent de paramètres semble plus coûteux que d'affiner une politique au fur et à mesure que des nouveautés

apparaissent. La politique robuste adopte à notre avis un comportement trop conservateur dans la solution fournie où la priorité est de limiter les dégâts liés aux évènements du futur. Par conséquent, nous souhaitons privilégier une politique adaptative à une politique robuste. Cependant, quelles sont les limites actuelles rencontrées dans la conception de politiques adaptatives ? La prochaine section s'intéresse aux méthodes de conception actuellement utilisées par les décideurs politiques et s'interroge sur les outils requis permettant de prendre en compte les données disponibles pour produire des politiques adaptatives.

2.4 Une tâche coûteuse pour le concepteur

Quel que soit le choix du type de politique que le décideur souhaite élaborer, le processus d'implémentation et d'évaluation de la politique requiert un investissement conséquent de la part de celui-ci. À travers l'observation de la littérature, nous nous intéressons à l'implémentation d'une politique urbaine et aux éléments clés nécessaires pour permettre de convenablement établir la politique. Une fois cette politique implémentée, correspondant aux choix des actions politiques proposées pour modifier la réalité, celle-ci est mise en place et évaluée par les parties prenantes afin de déterminer la réussite ou l'échec de la politique. Nous aborderons donc ce qui signifie un échec ou une réussite et comment l'évaluer. Enfin, nous traiterons brièvement ce qu'une politique, qu'elle soit pertinente ou non, peut offrir comme perspectives au décideur pour les politiques futures.

2.4.1 Conception et validation d'une politique urbaine

L'ouvrage de référence de Birkland [25] utilise la notion de *policy design* pour désigner le processus par lequel une politique est conçue pour atteindre un but spécifié par le décideur. Ce processus politique requiert de collecter des données pour construire ce qu'il appelle un énvironnementṕour la prise de décision. Ces données sont issues de sources variées mais l'une des plus importantes selon lui est l'opinion publique. Le sondage, par exemple, est une manière classique de collecter des informations sur les préférences du public. Les acteurs publics et privés sont donc considérés comme les acteurs de demain, par opposition aux acteurs actuels comme les organismes et les décideurs politiques. L'époque où les citoyens étaient considérés comme des individus à satisfaire politiquement pour s'assurer de meilleurs résultats aux prochaines élections est maintenant révolue [260].

Ainsi, les futurs acteurs ne sont pas considérés comme d'importance capitale dans le processus de conception politique, notamment dû à la difficulté d'identifier leurs attentes particulières. Une question nous semble donc importante pour la suite de ce chapitre : comment décrire de manière exhaustive les éléments propres à un environnement pour éprouver des politiques si les principales cibles de ces politiques ne sont pas bien considérées ?

Il faut observer de plus que l'environnement pour la décision politique doit d'une part considérer le comportement des différentes parties prenantes pour déterminer comment les propositions d'actions politiques vont permettre d'améliorer le sujet d'étude, et d'autre part prendre en compte une description réaliste de

l'environnement. Cette tâche d'accumulation et de collecte des données d'usages est généralement longue et délicate, et s'avère donc coûteuse pour le décideur.

Une fois que le décideur politique a complété son environnement de prise de décision, il peut construire la politique urbaine et la proposer à sa communauté politique. Nous parlons alors de la phase d'évaluation. Celle-ci est vue dans un premier temps comme la fin du processus de conception où la politique est satisfaisante et acceptée par les décideurs politiques et les gens concernés, ou au contraire soit annulée, soit retravaillée. Albæk propose un historique du processus d'évaluation des politiques, de son émergence en 1960 jusqu'au début des années 90 [3]. La conclusion de son travail est que différentes parties prenantes peuvent réagir différemment à une même politique puisqu'ils possèdent chacun leurs intérêts propres. Cela rejoint notre réflexion précédente, appuyant le besoin de disposer d'un outil permettant de centraliser les retours des différentes parties prenantes et ce qu'ils apprécient ou non dans la politique proposée. Cela réduit aussi les risques de l'appréciation après-coup de la politique dans la réalité.

2.4.2 Résultats des politiques et apprentissage des erreurs

Qualifier une politique de réussite est sujet à discussion. En effet, un décideur politique pourrait affirmer que la politique qu'il propose est un succès alors qu'un opposant certifiera le contraire, ou tous deux peuvent se prêter à un *blame game* où chaque décideur politique rejette la faute des difficultés rencontrées sur son prédécesseur [108]. Certains auteurs affirment ainsi que, pour éviter d'être tenus pour responsables de l'échec des actions politiques, les gouvernements et décideurs politiques sont encouragés à éviter la définition précise d'objectifs politiques et donc l'évaluation des politiques [115]. Dans le cas où la politique urbaine est une instance complexe traitant généralement plusieurs problèmes, il est possible que la réussite de la politique porte sur un nombre restreint d'aspects. Quels sont donc les outils dont disposent les décideurs pour d'une part affirmer qu'une politique est pertinente ou non, et d'autre part anticiper son éventuel succès ?

McConnell propose la définition suivante pour établir le succès d'une politique [165] :

> Une politique est une réussite si elle satisfait les objectifs des partisans qui l'ont envisagée et si elle ne déclenche aucune critique ou si est universellement acceptée.

Avec une telle définition, le succès d'une politique est rarement garanti. De plus, cette définition ne prend pas en compte d'échéance temporelle. Pendant combien de temps est ce que la politique ne doit recevoir aucune critique ? D'autant plus qu'avec la Smart City, les parties prenantes ont la possibilité d'exprimer de plus en plus facilement leurs avis, de même qu'il est possible de capturer automatiquement les données permettant de qualifier les usages (ligne de bus peu utilisée, navette de tram surchargée en heure de pointe, etc.). Ainsi, le succès d'une politique n'est pas binaire mais nuancé.

Il propose également la définition de l'échec d'une politique [165] :

> Une politique échoue, même si elle est une réussite sous certains aspects, si elle ne répond pas aux objectifs des partisans qui l'ont envisagée et si l'opposition est forte ou si elle ne dispose d'aucun support.

Nous pouvons également identifier les sources d'échec suivantes [112, 211] :

— les bénéfices de la politique sont moins élevés que les coûts ;
— pas d'amélioration de l'existant ;
— le résultat est strictement moins bon que ceux obtenus par d'autres gérant les mêmes problématiques.
Ces échecs, identifiés dans les politiques, permettent d'extraire des connaissances et améliorer les nouvelles propositions de politiques.

L'échec d'une politique est l'occasion d'extraire des connaissances sur les décisions politiques effectuées [162, 25]. Plusieurs travaux proposent ainsi des modèles pour extraire des informations d'une politique, quel que soit son résultat, et d'apprendre sur ce qui a été pertinent lors de son cycle de conception. Un parallèle est établi entre l'évaluation d'une politique et la création d'un ensemble rigoureux de méthodes pour la conception de politiques, l'*evidence-based policy-making* [97, 211]. L'observation du résultat d'une politique, son échec comme sa réussite, fournit des indices sur la pertinence ou non des choix politiques effectués : objectifs réalistes, choix des actions politiques [162]. Des enseignements peuvent être tirés pour créer une nouvelle itération de la politique ou améliorer la construction de prochaines politiques [211].

2.4.3 Besoin de solutions pour une meilleure intégration des usagers

La politique est initialement envisagée comme *top-down* où les décideurs politiques identifient ensemble un problème et décident de construire des solutions [104, 200]. La conception *bottom-up* des politiques propose de s'intéresser aux problèmes identifiés, dans un premier temps au niveau plus local [149], puis dans un second temps directement par les citoyens, qui les font remonter à leurs représentants politiques [25]. L'intérêt de ces démarches est que le citoyen participe davantage au processus de conception des politiques et apportent un regard plus nuancé sur les objectifs envisagés par les décideurs [102]. L'objectif est louable mais les méthodes actuelles limitent la possibilité des citoyens de s'exprimer facilement et leurs interactions ne sont pas toujours représentatif de la réalité.

Pour faciliter l'intégration des données et des nouveaux acteurs dans le processus politique, nous envisageons d'agencer un environnement afin de construire une politique. Cependant, que représente cet environnement ? Est-ce un outil de développement ? Un cadre de réflexion utilisé par les décideurs ? Et surtout, comment construire cet environnement ? Comment intégrer le retour des usagers plus facilement qu'en les sollicitant lors de séances publiques ou via des sondages par exemple ? Nous avons mentionné en introduction que la Smart City modifie le paysage politique de différentes manières. L'une d'entre elles, que nous aborderons à la section suivante, propose justement de s'intéresser à ces questions.

2.5 Smart Cities : les villes de demain (et d'aujourd'hui ?)

Depuis l'émergence du terme de Smart City dans les années 90 [83], plusieurs termes ont été utilisés dont les plus fréquents sont la *ville intelligence* (*intelligent city*) [136], la *Smart City* (la communauté effectue une distinction entre ces termes mais celle-ci se perd avec la traduction française [52]) et la ville digitale (*digital city*) [212]. Le parcours de la littérature indique qu'il n'existe pas encore de consensus sur la définition de la Smart City [42, 105, 176] et qu'elle est encore sujet à débat [105, 106]. Des travaux proposent même une

synthèse des définitions rencontrées dans la littérature autour de la Smart City [4, 147, 170]. Parmi ce large choix, nous optons pour une définition combinant d'une part le besoin d'infrastructures de communications (ou *Information and Communications Technology* ICT) mais également l'inclusion d'outils permettant au citoyen de participer à l'évolution de sa ville. En effet, nous rejoignons la position d'Hollands [105] qui critique les villes se revendiquant de Smart Cities dès lors qu'elles disposent de réseaux de communication connectés, sans garantir l'existence des traitements et des analyses qui vont avec les données collectées ni l'ouverture aux citoyens. Les infrastructures technologiques doivent en effet servir de plateformes facilitant la communication entre les parties prenantes et favoriser l'innovation. Allwinkle évoque ainsi l'apprentissage réalisé par les intervenants au sein d'un processus de co-construction simplifiée des politiques urbaines envisagées [7].

Avec la synthèse des définitions de Smart City par Giffinger *et al.* [84], Nam et Pardo [176], Caragliu *et al.* [40] et Marsal-Llacuna *et al.* [158], nous considérons donc que :

> Une ville est intelligente, au sens de la Smart City, si elle dispose d'infrastructures connectées capable de fournir des données sur les usages et les usagers pour améliorer la qualité de vie dans la zone urbaine.

Nous considérerons d'autre part que le décideur d'une Smart City aura besoin d'un accès simplifié aux données nécessaires pour élaborer les politiques.

Dans cette section, nous abordons ce que peut apporter la Smart City à la conception de politiques en nous intéressant dans un premier temps à la collecte de données avec les données ouvertes et la fouille de données et dans un second temps, nous mentionnerons comment le citoyen devient partie intégrante du processus décisionnel. Enfin, cette section s'intéresse à l'usage que nous pouvons faire de ces nouvelles données avec l'apport de la modélisation et de l'informatique pour construire des outils d'aide à la décision adaptées.

2.5.1 Accessibilité simplifiée aux données

Avec l'émergence de la Smart City, de nouveaux outils se développent pour créer et acquérir des données. Dans cette section, nous abordons les initiatives effectuées par les organismes privés et surtout publics pour rendre accessible leurs données avec les démarches sur les données ouvertes (*Open Data*) et l'implication sur les données accessibles et leur traitement dans un contexte de masse de données (*Big Data*).

Données ouvertes

La Smart City se définit en partie par l'installation d'infrastructures de communications pour collecter automatiquement les données d'usages. Par exemple les capteurs pour calculer la vitesse moyenne des véhicules sur des tronçons routiers ou l'occupation des emplacements de stationnement par tranche horaire [116]. L'adoption des données ouvertes permet aux différents acteurs (privés comme publics) de les utiliser pour produire des services à la communauté [173]. La ville de Lyon propose de rendre disponible des informations d'usages et des informations géographiques [1] afin de réaliser des applications sur l'usage de vélos collaboratifs par exemple.

1. Site des données ouvertes de Lyon

Les données ouvertes permettent d'augmenter la transparence des entités et augmente l'engagement du public dans le processus politique [117]. Les données n'ont de la valeur qu'une fois exploitées par les organismes intéressés. Ainsi, les dispositifs permettant de rendre la donnée ouverte permettent d'accroitre les efforts sur l'usage de celle-ci [101]. Ces données sont à disposition du public et peuvent également être utilisées par celui-ci. Cependant, il ne dispose pas nécessairement des mêmes connaissances ou dispositifs techniques que des experts politiques.

Par conséquent, un outil permettant de simplifier et réduire les connaissances expertes requises pour les analyser serait un plus pour un renforcement à grande échelle de l'engagement dans le processus politique et d'accroitre les démarches de données ouvertes. Avec notre outil d'aide à la décision, il serait envisageable que les usagers expriment leurs avis sur les politiques testées. Ainsi la visualisation des données simplifie l'analyse, pour le décideur et éventuellement les usagers, et facilite les échanges et propositions sur les politiques suggérées.

Masse de données

De même que pour la Smart City, il n'existe pas une définition unique du Big Data, d'autant plus que ce qui correspondait au Big Data il y a une vingtaine d'années est différent de ce que l'on considère aujourd'hui comme appartenant à cette catégorie. De Mauro *et al.* proposent une comparaison des différentes définitions [62] mais nous utiliserons la définition de Manyika [155] :

> [Le Big Data sont] des ensembles de données dont la taille est supérieure aux capacités classiques des logiciels à collecter, enregistrer, organiser et analyser.

L'objectif de cette section n'est pas de décrire les techniques existantes pour collecter les Big Data (voir Chen *et al.* pour une revue [49]) ou de les analyser (on parle alors de Big Data Analytics, voir Tsai *et al.* [235]) mais plutôt de s'intéresser à leur apport dans le processus politique.

Le Big Data porte donc sur l'ensemble des données produites et la manière de les collecter et de les agréger [34]. Une première utilisation porte sur l'analyse des données sociales collectées sur les sites comme *Facebook* ou *Twitter* [127] permettant de collecter des retours sur la qualité des services par exemple. La multiplication des capteurs, des applications mobiles et la capacité de collecter ces informations fournissent donc aux parties prenantes une source d'information conséquente. Les données d'usages permettent d'identifier et d'anticiper la réaction des citoyens par rapport à l'application d'une action politique de manière indirecte. Il ne s'agit plus de créer des séances d'échanges dont, comme nous l'avons mentionné précédemment, les échanges peuvent être biaisés. Les décideurs disposent d'un ensemble d'informations brutes où les citoyens expriment leur avis. Certains auteurs se questionnent sur la pertinence de ce genre de retours sur la qualité de la conception politique [217] alors que d'autres y voient une opportunité de développer des approches bottom-up où les problématiques politiques sont identifiés par les citoyens [204].

Apport pour le citoyen

Nous pouvons conclure que ces démarches changent la manière dont la politique est conçue en fournissant aux citoyens des outils qui étaient autrefois les outils des décideurs. En rendant publique les données, le citoyen peut questionner et réfléchir sur les choix des actions politiques. D'un autre côté, les décideurs politiques possèdent désormais des outils pour collecter plus facilement des retours plus authentiques de la part des citoyens et pour utiliser toutes ces données pour affiner le choix politique et faciliter l'évaluation.

Bien que ces problématiques sortent du cadre de nos travaux, il est important de se questionner autour de l'éthique au niveau de la Smart City [132]. Pour que les données d'usages soient pertinentes, il est nécessaire de prendre en compte un volume important d'informations personnelles sur les préférences et la personnalité des citoyens. Afin de respecter la vie privée des usagers, il faut garantir aux usagers que leurs données sont privées et anonymisées [59]. Solove dresse une liste des infractions à la sécurité des données [227], nous considérons cependant deux points importants : l'utilisation des données collectées pour un but précis ne doit pas l'être pour un autre but sans avoir le consentement du sujet (utilisation secondaire) et le sujet doit connaitre et avoir accès aux données qui sont collectés sur lui (exclusion). Par conséquent, l'utilisation de ces techniques doit être encadré et la sécurité des données ainsi que leurs usages doivent être réfléchis [34, 132, 225].

2.5.2 Intégration des usagers dans le processus décisionnel

L'idée de faire prendre part à la décision toutes les parties prenantes remonte au moins à la fin des années 1970 [255]. Le *tournant participatif* a eu lieu dans les années 2000, où l'usage de la participation des parties prenantes a significativement augmenté [238]. Dans la littérature, nous trouvons les termes de modélisation participative (*participatory modelling*) [71] ou de participation à médiation(*mediated modelling*) [242] pour décrire l'intégration des parties prenantes dans le processus décisionnel ou encore de modélisation environnementale intégrée (*Integrated Environmental Modeling*) [139] pour décrire les modèles pour cette intégration. La modification de la gouvernance, et plus particulièrement de la gouvernance intelligente (*smart governance*) ou gouvernance participative est l'une des clés de la Smart City [18, 85] et requiert la mise en place d'infrastructures pertinentes permettant plus de transparence, de collaboration et d'échanges [120]. Les bénéfices apportés par la gouvernance participative sont les suivants [137] :

— la participation permet de capturer l'ensemble des connaissances et des perspectives et de produire de meilleurs résultats ;
— la participation permet une meilleure acceptation des résultats, une facilité d'implantation des politiques et une meilleure gestion des conflits.

Cette gouvernance participative s'inscrit dans la volonté de fournir au citoyen une voix dans le processus politique. Le *citizen empowerment* (donner le pouvoir aux citoyens) est la capacité du citoyen a agir lui-même sur les problèmes auquel il est confronté [128]. Cette notion émerge dans les années 1980 mais devient désormais possible dans le cadre de la gestion politique avec la Smart City [41, 63] et l'eGouvernance.

Dans cette section, nous nous intéressons plus particulièrement à la place des usagers dans le processus décisionnel et comment obtenir leur participation. Dans un premier temps, nous définissons l'acteur participant

au processus de décision et les avantages de cette participation. Dans un second temps, nous étudions les outils pour solliciter les parties prenantes et construire de nouveaux types de modèles. Enfin, nous nous intéressons plus spécifiquement aux notions de modélisation et à l'usage des jeux sérieux dans la littérature.

Quels sont les nouveaux acteurs sollicités ?

La Smart City est l'occasion de donner à la parole à de nouveaux acteurs de la vie politique même si les hommes politiques et les experts (urbanistes, architectes, etc.) restent très impliqués. Ces nouveaux acteurs sont donc les citoyens, parfois directement intégrés dans la boucle de conception des politiques, qui ont maintenant l'opportunité de discuter avec les parties prenantes. Ils peuvent d'autre part exprimer indirectement leur avis sur les réseaux sociaux par exemple ou directement en utilisant des applications sur smartphones dédiées [173]. Surowiecki avance que, même sans connaissances expertes moyennes, l'ensemble des individus d'une société peut contribuer à la construction d'une politique pertinente et que chacun d'entre eux peut évaluer ce qui est pertinent pour lui ou elle [228]. Il appelle ce phénomène la *sagesse des foules* (*Wisdom of Crowds*). Avec des outils adaptés, les décideurs et les citoyens peuvent présenter leurs idées et échanger. L'analyse de l'agrégation de ces retours permet (dans l'idéal) de produire une politique co-construite satisfaisante.

Engagement dans la vie politique L'engagement dans la tâche politique est un problème auxquels sont confrontés les décideurs politiques [29]. Un moyen envisagé est d'avoir recours à des annonces sur les réseaux sociaux ou des applications mobiles [133]. La ludification est un support régulièrement utilisé pour la simulation participative et comme moyen d'échanger autour d'un sujet [233]. Gong [88] insiste sur l'intérêt de disposer d'outils pour évaluer ces coûts en amont, notamment en concevant des expérimentations à l'aide d'applications dédiées. La ludification augmente l'engagement et les parties prenantes expriment leur opinion en réagissant à l'évolution des activités ludifiées [99, 198]. Les jeux sérieux sont utilisés dans la planification urbaine [198, 206] où ils stimulent l'engagement de l'utilisateur et le sensibilisent aux problèmes politiques [248]. L'usage du jeu sérieux dans l'élaboration de politiques est questionné par Mayer [163] qui s'intéresse à son impact sur la perception des politiques. En effet, la construction d'un jeu sérieux et sa proposition à des utilisateurs revient déjà à simplifier le problème. Avec les échanges autour du jeu, il est ainsi possible pour les décideurs et les utilisateurs de réfléchir à la complexité technique et sociale de leurs suggestions et comment proposer un nouveau discours pour présenter leurs résultats.

Cependant, les jeux sérieux ont également leurs limites, notamment le manque de réalisme et les hypothèses choisies peuvent être remises en question [198]. De plus, il existe peu d'exemples de politiques co-construites avec des jeux sérieux pour déterminer l'impact de l'interaction entre décideurs et utilisateurs sur la qualité de la politique [107, 35].

Usage de la gouvernance participative

Dans leurs travaux sur la modélisation environnementale intégrée, Laniak *et al.* proposent de nouvelles terminologies pour la modélisation [139] en faisant une distinction entre l'intégration pour construire des

systèmes holistiques et l'évaluation pour l'analyse de la pertinence des politiques. Les travaux de Voinov *et al.* présentent une liste d'outils de modélisation pour la participation des parties prenantes [248], disponible pour les décideurs politiques. Dans un premier temps, les décideurs et les parties prenantes définissent ensemble la vision et la portée de la politique à l'aide de réunions, brainstormings et la conception de scénarios. Les objectifs de la politique sont ensuite déterminés avec la sélection des paramètres et des variables que le modèle pourra prendre en compte. Cependant, le débat seul ne permet pas de capter toutes les opinions des usagers mais uniquement d'obtenir des retours de la part des gens intéressés par la démarche initiale. Des retours plus complets surviennent lorsque les usagers vivent au quotidien la politique mise en place, et commencent à s'exprimer par les moyens de concertation disponibles. De la même manière l'expression sur une politique introduit un autre biais. Si l'on exprime plus facilement un avis négatif que positif sur des plateformes de communications comme *Twitter*, ce n'est pas pour autant que quelqu'un qui ne s'exprime pas approuve la politique en place. Il est difficile d'estimer l'impact et l'importance des retours des usagers dans ce cadre-là. Les échanges constructifs sont complexes à obtenir dû à l'application actuelle des politiques. En effet, un décideur propose une politique pouvant donner lieu à un débat où les citoyens réagissent. Le décideur ajuste ensuite sa politique en conséquence. Il n'y a donc pas de véritable débat synchrone. En permettant au citoyen de s'exprimer pendant le processus de conception, de manière continue plutôt qu'après de manière ponctuelle, il est possible de rendre plus adaptatif le processus de conception et de réduire les erreurs commises. Il n'est alors plus nécessaire de chercher une conception initiale de la politique *parfaite* puisqu'il sera désormais possible de l'ajuster en continue.

En étudiant les travaux sur la gouvernance participative, il apparait que celle-ci peut être sollicitée aux trois grandes étapes de la conception :

1. L'identification d'un problème dans une approche bottom-up où les citoyens expriment leurs problèmes ;
2. La suggestion de modifications en collaboration avec les décideurs politiques pour affiner les actions et solutions proposées ;
3. Les critiques ou le support attribués à une politique donnée après que celle-ci soit appliquée en réalité et des suggestions pour évaluer les impacts et les effets, les conséquences sur le court et long terme.

Actuellement, le troisième point est fréquemment utilisé car les réactions apparaissent naturellement suite aux mauvaises politiques. Notre objectif est de proposer un outil permettant de permettre une gouvernance participative en intégrant ces trois aspects. Cependant, si nous considérons que la Smart City est déjà disponible et que le citoyen peut plus facilement exprimer son avis, il est encore nécessaire de l'engager dans le processus politique pour obtenir des retours qui seraient aussi bien positifs que négatifs plutôt que régulièrement négatifs.

2.6 Conclusion et perspectives sur le rôle de l'Intelligence Artificielle dans la prise de décision

Le nouveau paysage de l'élaboration des politiques urbaines passe par la Smart City avec l'accès à quantité de données (sur l'occupation de la voirie, les places de stationnement, la pollution, etc.) ainsi que

des données représentant l'avis des citoyens, favorisant le *citizen empowerment* en octroyant davantage de pouvoir aux habitants, pour participer à l'élaboration des politiques via des outils appropriés. Ainsi toutes les parties prenantes, experts, citoyens ou acteurs privés, peuvent être intégrées dans la prise de décision politique. À travers ce chapitre, nous avons montré l'intérêt et l'importance de fournir aux décideurs des outils spécifiques permettant de prendre en compte les données ouvertes de la métropole, des données collectées automatiquement sur les impacts de la politique et le retour des usagers pour ajuster les actions choisies, avec un protocole d'interactions adapté aux parties prenantes. Plusieurs travaux se sont développés dans ce sens et notamment la simulation multi-agent [142] : dès 2009, la communauté économique a suggéré l'utilisation massive de modèles multi-agents pour l'aide à la décision [74], en opposition aux systèmes experts, qui étaient légion dans les années 80.

Avec les limites des systèmes experts, notamment la difficulté de l'acquisition des connaissances, les problèmes de validation et d'explicabilité des résultats [164, 196], les travaux sur les agents intelligents et son corollaire, l'IA distribuée ou collective, se sont fortement développés [114, 119].

La simulation devient ainsi de plus en plus présente dans le processus décisionnel, et particulièrement pour la conception de politiques, elle est reconnue comme pouvant réduire les coûts. En effet la simulation permet d'une part d'étudier les impacts et les effets des politiques avant leur implémentation, et d'autre part de rendre plus efficace la consultation publique via par exemple des démonstrations, la simulation visuelle pouvant faciliter les explications relatives à la politique [129].

Nous estimons ainsi que la simulation multi-agent est un bon outil de conception de politiques, notamment grâce à la visualisation des impacts des actions politiques qui peut faciliter la prise de décisions. C'est également un bon support d'interactions pour co-construire les politiques, en multipliant et simplifiant les protocoles d'échanges possibles avec les différentes parties prenantes. Des politiques ont déjà été conçues en ayant recours à la simulation multi-agents : l'aménagement urbain [195, 161] ou les impacts sur la pollution [79]. Dans ces approches, la simulation multi-agents est souvent utilisée comme support d'interactions où certains acteurs sont humains et prennent des décisions, et d'autres acteurs sont des agents artificiels capables de proposer des adaptations en fonction des règles de gestion définies par le concepteur.

Notre objectif dans le cadre de cette thèse est de construire un outil d'aide à la décision à partir d'une simulation multi-agents de l'environnement et d'adopter une approche innovante où l'intelligence artificielle est également utilisée au niveau des agents responsables de la prise de décision politique. Le décideur pourra donc suggérer et tester différentes actions politiques et exprimer des suggestions relativement aux politiques simulées, en constatant visuellement leurs effets sur l'environnement virtuel. En couplant notre outil aux possibilités offertes par la Smart City, nous envisageons de construire des politiques continues, régulièrement mises à jour avec la collecte ininterrompue de données et le retours des citoyens, dans une boucle dynamique de conception des politiques. Une perspective sur le long terme est de s'intéresser à l'acceptation de notre démarche par la société, où le décideur et les parties prenantes restent maîtres de la prise de décision politique et où l'intelligence artificielle prend le rôle de conseiller, à partir d'une vision globale élaborée sur l'observation de l'environnement, pouvant être différente de celle envisagée par les décideurs.

Chapitre 3

L'Intelligence Artificielle Distribuée au service de la conception de politiques

Sommaire

3.1 Introduction

La mise en place de systèmes intelligents pour assister le décideur politique dans la conception de politiques est d'autant plus importante que l'avènement de la Smart City accroit le nombre de données à prendre en compte. S'il était possible de disposer d'une simulation parfaite de la réalité, un monde bac à sable virtuel où il serait possible de tester des modifications politiques et d'étudier leurs impacts sur le long terme, la conception politique serait simplifiée. Nous ne sommes pas encore à ce niveau de simulation mais nous pouvons envisager des outils s'en approchant. Dans un contexte où la zone urbaine est un système complexe où les usagers représentent des individus aux objectifs multiples, il apparait comme une évidence d'exploiter les simulations multi-agents pour les représenter. Dans le cadre de la construction de politiques, nous nous intéressons plus particulièrement à la représentation des comportements réalistes pour produire des politiques pertinentes. Cependant, avant de pouvoir tester des politiques dans un monde virtuel adapté, il est nécessaire de construire les entités de ce monde, à savoir l'environnement, support des interactions des utilisateurs, et les avatars des utilisateurs eux-mêmes.

Notre état de l'art se poursuit avec ce chapitre introduisant la simulation multi-agents et ses composantes : les agents et l'environnement. Nous décrirons ainsi brièvement ce qui définit l'agent et l'environnement. Nous parlerons de leur combinaison, formant généralement des systèmes complexes, et qui permet d'aborder la notion de simulation multi-agents ainsi que son fonctionnement. Nous indiquerons enfin comment la simulation peut s'inscrire dans un processus d'aide à la décision.

Ensuite, nous aborderons une propriété importante pour notre approche, avec l'émergence des comportements chez les agents. Compte-tenu de notre champ d'application, un intérêt particulier est porté à la modélisation réaliste des systèmes observés, ainsi qu'à la visualisation des données de simulation.

Enfin, les problématiques liées à la simulation multi-agents sont discutées et amènent à explorer le concept de simulation multi-niveaux et la mise en œuvre du couplage de dynamiques pour la représentation des systèmes complexes.

3.2 Représentation des systèmes complexes avec les systèmes multi-agent

Les systèmes multi-agents (SMA) sont composés d'un ensemble d'entités autonomes appelées agents. Ces agents interagissent avec un environnement dynamique à l'aide d'une boucle sensorimotrice décrivant les perceptions de l'état de l'environnement qu'ils reçoivent de leurs capteurs et les actions qu'ils effectuent par l'intermédiaire de leurs actionneurs.

Les systèmes multi-agents relèvent de l'intelligence artificielle distribuée (*Distributed Artificial Intelligence* ou DAI), même si Ferber, dans l'introduction de son ouvrage sur les systèmes multi-agents [75], s'interroge sur les raisons de distribuer l'intelligence. Son explication est que les systèmes deviennent de plus en plus complexes et qu'ils disposent d'un nombre important de sous-systèmes. On parle ainsi de systèmes complexes, qui sont des systèmes comportant un grand nombre d'éléments en interaction les uns avec les autres, sans ordre

prédéfini et de façon simultanée, ce qui rend difficile la prédiction de comportement du système global par des algorithmes classiques de modélisation. Ainsi les systèmes multi-agents sont perçus comme une solution à la modélisation de systèmes complexes, avec beaucoup d'entités ou de multiples perspectives d'interactions et d'évolution (figure 3.1).

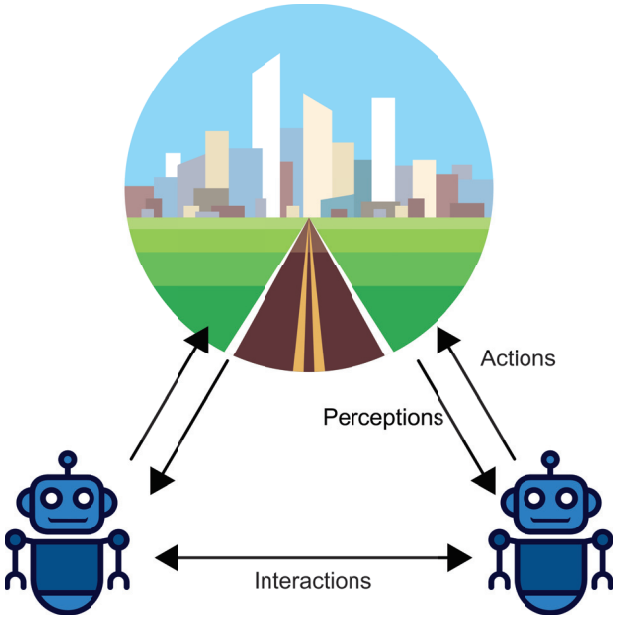


FIGURE 3.1 – Représentation de la boucle sensorimotrice entre deux agents dans un même environnement. Dans un système multi-agents, les agents vont également interagir entre eux et faire émerger des comportements.

La modélisation d'un système complexe par un système multi-agents conduit à identifier les différentes entités en jeu et à représenter leurs interactions.

3.2.1 L'agent comme entité atomique

Dans la littérature, différents types d'agents existent, avec notamment la distinction entre les agents physiques (*hardware agents*), pouvant par exemple être représentés par des robots qui évoluent dans le monde réel, et les agents logiciels (*software agents*) évoluant dans un environnement virtuel. Dans le cadre de cette thèse, seuls les agents logiciels sont considérés.

La définition d'un agent comme entité individuelle, proposée par Russel et Norvig [208], considère comme agent toute entité en mesure de percevoir son environnement à l'aide de capteurs et d'agir sur les éléments de l'environnement par l'intermédiaire d'actionneurs. Dans leur ouvrage, ils utilisent le terme d'agent *rationnel* emprunté à l'économie et où l'agent prend des décisions rationnelles, c'est-à-dire qu'il cherche à maximiser une mesure de performance interne en fonction de sa perception du monde et de ses connaissances. Wooldridge et Jennings ajoutent à la boucle sensorimotrice la notion d'autonomie qui leur permet d'agir sans intervention

directe extérieure (humaine ou autre) et ont le contrôle de leurs actions et de leur état interne [119, 258]. Ils ajoutent également les notions de réactivité, où les agents réagissent aux changements qui s'opèrent dans l'environnement et de pro-activité, où les agents anticipent certaines évolutions de l'environnement. Ces éléments dotent les agents de comportements complexes qui leur permettent d'atteindre les objectifs recherchés.

3.2.2 L'environnement, le support de l'interaction

La modélisation de l'environnement est cruciale pour l'agent puisqu'il correspond au support de l'interaction et permet aux agents d'avoir des boucles sensorimotrices pertinentes. La structure de l'environnement dépend des choix réalisés par les concepteurs et le type de tâches qu'ils souhaitent résoudre [254]. Il est par exemple possible de décrire les interactions entre les agents avec un support de communication direct via un protocole spécifique défini au niveau des agents [180] ou par des échanges indirects où l'agent modifie l'environnement pour transmettre l'information aux autres agents : la stigmergie chez des fourmis par exemple où le dépôt de phéromones permet de construire une piste pour la recherche de nourriture, ou encore par l'intermédiaire de messages laissés sur un tableau noir [73].

Russel et Norvig proposent un ensemble de propriétés clés adoptées par la majorité de la communauté [208] :

- Un environnement *accessible* signifie que l'agent dispose d'une représentation totale de l'environnement. À l'inverse, un environnement *inaccessible* ne fournit qu'une représentation partielle de celui-ci.
- Le changement d'un état dans un environnement *déterministe* dépend uniquement de l'état courant et des actions sélectionnées par les agents alors qu'un environnement *non-déterministe* ou *stochastique* aura une distribution de probabilité sur les états probables à partir de l'état courant et des actions sélectionnées. Par exemple, un agent ayant un actionneur défaillant n'effectuera pas l'action qu'il souhaite dans 100% des cas.
- Un environnement *statique* ne change pas pendant que l'agent prend sa décision et effectue son action. Un environnement *dynamique* peut évoluer entre le moment où l'agent reçoit ses perceptions et le moment où il effectue son action.
- Le type des perceptions et actions disponibles pour l'agent est déterminé par l'aspect *discret* ou *continu* de l'environnement.
- Un environnement est *inconnu* si l'agent doit apprendre le modèle de l'environnement, c'est-à-dire doit apprendre l'effet d'une action sur l'état courant de l'environnement. Un environnement est *connu* de l'agent quand il dispose du modèle de son environnement.

La description de l'environnement peut être faite en s'intéressant aux éléments composant l'environnement, les actionneurs disponibles pour les agents et leurs capteurs, ainsi que de leurs mesures de performance [208].

L'environnement joue un rôle important dans la simulation multi-agents [254] puisqu'il est le support des calculs de la simulation et l'espace où se créent de nouvelles entités (structures, agents). Il décrit donc un cadre structurel avec les définitions de Russel et Norvig ci-dessus, mais il peut également décrire un cadre organisationnel en considérant des structures sociales [75] que les agents utilisent pour agir.

3.2.3 Fonctionnement d'un système multi-agents

Une définition étendue de l'agent au domaine multi-agents est proposée par Ferber, qui dresse une liste d'attributs synthétisant ce que tout agent évoluant dans une simulation multi-agents doit posséder [75, 76] :

— L'agent peut communiquer avec d'autres agents ;
— L'agent est mû par un ensemble d'objectifs propres ;
— L'agent possède ses propres ressources ;
— L'agent ne dispose que d'une représentation partielle des autres agents ;
— L'agent possède des compétences qu'il peut offrir aux autres agents ;
— L'agent a un comportement tendant à satisfaire ses objectifs, en tenant compte d'une part des ressources et des compétences dont il dispose, de ses propres représentations du monde, et des communications qu'il perçoit.

Les agents d'un système multi-agents interagissent avec leur environnement et les autres agents pour atteindre leurs objectifs.

En combinant un environnement adapté à nos besoins avec un ensemble d'agents, une description formelle peut être élaborée pour les systèmes multi-agents [64, 76] à partir des éléments suivants :

— Un environnement décrit en fonction des besoins du concepteur [254] ;
— Un ensemble d'agents possédant des objectifs et un état interne ;
— Un ensemble de perceptions de l'environnement associé à chaque agent et qu'ils peuvent obtenir ;
— Un ensemble d'actions que les agents peuvent effectuer sur l'environnement ;
— Une fonction d'évolution du monde, construite à partir des actions proposées par les agents.

L'utilisation de l'approche orientée agent pour la modélisation de systèmes complexes est donc naturellement envisagée, la représentation par des agents dotés de capteurs et de leur propre prise de décision permettent de distribuer la complexité [118, 28]. Les approches orientées agents sont utilisés dans de nombreux domaines pouvant être qualifiés de système complexe comme l'écologie [90], l'industrie [185, 197], les transports [48] et nous verrons plus en détail les travaux sur la modélisation de la ville dans la section 3.3.

Les environnements que nous étudions dans le cadre de cette thèse seront partiellement observables, stochastiques et discrets.

Émergence de comportements

Une simulation multi-agents est utilisée pour modéliser des systèmes complexe hétérogènes et évolutifs. Dans ce cadre, l'utilisation d'agents exhibant des comportements simples peuvent faire émerger une intelligence supérieure à celle des agents qui composent la simulation. Reynolds propose une modélisation de comportements complexes comme le déplacement de nuées d'oiseaux ou des bancs de poissons avec des boids (agents) dont les interactions sont régies par de simples règles d'évitement de collision avec leurs voisins, permettant ainsi l'émergence de motifs de déplacement complexes [205]. Les exemples d'interactions que l'on peut rencontrer sont la *coopération*, où les agents travaillent ensemble vers un objectif commun [75] ; la *coordination*, où les agents s'organisent de manière à augmenter le nombre d'interactions positives ; la *négociation*, où les agents

se consultent pour parvenir à une décision commune ; la *compétition*, dont le but est d'atteindre un objectif avant les autres agents.

3.3 Modélisation d'une zone urbaine

Une zone urbaine est un système complexe représentant de nombreux acteurs aux objectifs et attentes différentes ainsi que l'ensemble des dynamiques existantes exploitées par ces acteurs. Il s'agit du réseau de transport urbain (bus, métro, voitures), des services publics (hôpitaux, écoles), de l'économie (lieux de travail) et des quartiers résidentiels. En fonction des objectifs du concepteur, une modélisation pertinente de la zone urbaine nécessite de disposer de ces dynamiques. Il est complexe d'avoir une vision globale et des hypothèses sont nécessaires pour avoir le modèle le plus simple possible mais comportant tous les éléments nécessaires à la bonne construction de l'environnement politique. La conception de politiques urbaines et d'aménagements du territoire doit prendre en compte toutes leurs particularités. La zone urbaine et sa modélisation sont aux cœurs de nos problématiques avec la conception de politiques urbaines. Ainsi, la modélisation géo-spatiale et les éléments à considérer sont abordés dans cette section.

En nous inscrivant dans la Smart City, la construction de nos simulations multi-agents sera alimentée de données représentant l'environnement, son fonctionnement et le comportement des usagers. Pour modéliser la ville, nous nous intéressons donc davantage aux approches élaborées à partir d'un grand volume de données disponibles plutôt qu'aux approches visant à collecter ces données. Dans un premier temps, nous abordons les techniques existantes utilisées pour la simulation de systèmes géographiques. Puis nous aborderons la notion de réalisme pour les agents et les populations synthétiques. Enfin, nous nous questionnerons sur la nécessité de modèles génériques pour la simulation multi-agents.

3.3.1 Approches existantes

Les premiers modèles urbains se basent sur les *automates cellulaires* (*Cellular Automata* CA) où les agents et l'environnement sont confondus. En effet, les approches à base de CA représentent l'environnement sous forme de cellules où la propriété agrégée de la cellule correspond à l'état d'une population par exemple [16, 152]. Cependant, même avec des modèles simples, la description de l'état des cellules comme différents niveaux d'utilisation du territoire permet de décrire un modèle urbain.

Les modèles à base d'automates cellulaires ont transité vers des approches utilisant la modélisation orientée-agents (*Agent-Based Modeling* ABM). Ces approches ont notamment permis de modéliser le déplacement des agents, qui n'était pas considéré dans le cas des CA. Les agents présents dans la simulation peuvent être représentés de manière hétérogène pour créer une dynamique d'échanges complexe entre eux et leur environnement. Ainsi, la principale différence entre les CA et les ABM est que le système est constitué d'un ensemble d'objets individuels avec un comportement unique et spécifique plutôt que d'avoir des modèles généraux où les comportements sont collectifs ou agrégés. La construction du premier modèle social d'agents est attribué à Thomas Schelling qui en 1971 présente comment les disparités entre populations font émerger des mouvements migratoires entre deux cultures distinctes [213].

Dans les années 1980, les modèles intégrés transports-urbanisme (*Land Use Transport Integrated* LUTI) voient le jour et sont conçus pour intégrer davantage les données d'informations géographiques (*Geographic Information System* ou GIS) et considérant tous l'éco-système du transport urbain pour construire une représentation plus poussée de l'existant [61, 111]. Les approches LUTI prennent en compte une zone urbaine plus conséquente que les ABM mais, de manière similaire aux CA, elles perdent le caractère d'individualité de l'agent, qui est de nouveau fusionné avec l'environnement.

Le choix de l'approche pour la modélisation urbaine dépend de la granularité souhaitée [56] : par exemple une échelle microscopique pour la gestion de feux de croisements [72] jusqu'à l'échelle macroscopique pour l'étalement urbain. Si le niveau d'abstraction est trop faible, la modélisation passe certainement à côté de détails clés, cependant trop de détails contraindra la modélisation qui devient trop complexe.

L'avantage, et la raison pour laquelle nous nous orientons vers la simulation multi-agents, est la nécessité de représenter des populations hétérogènes dont l'ensemble des interactions fournit un retour qui a du sens pour les décideurs politiques. L'approche multi-agents est utilisée pour gérer certains aspects des villes : simuler des piétons [45], gérer les emplacements de stationnement dans un quartier résidentiel [21], gérer les carrefours et les feux d'intersections [67, 257] ou encore mesurer l'étalement urbain [15, 161]. Dans le contexte de la modélisation de la ville, la recherche de réalisme est un impératif.

3.3.2 Réalisme de l'environnement et de l'agent

Les systèmes utilisant l'intelligence artificielle et par extension, les systèmes multi-agents, sont inspirés de l'existant ou plutôt de ce que nous comprenons de l'existant [172]. Cette réflexion est importante pour introduire la notion de *réalisme* d'une simulation. En effet, dans le cadre de la modélisation d'agents humains se pose le problème de l'évaluation du réalisme du comportement des agents et plus globalement du système complexe étudié, or il n'existe pas de métrique ni de méthodologie pour l'obtenir. Généralement le réalisme de la modélisation s'étudie de manière subjective par les concepteurs et les utilisateurs lors des expérimentations de simulation. Toute la difficulté réside dans le choix d'hypothèses de modélisation qui vont simplifier le modèle pour le rendre utilisable, tout en n'altérant pas le niveau de réalisme obtenu.

Le réalisme de l'environnement dépend notamment de la quantité et de la qualité des données disponibles. Par exemple, la description de l'emplacement des bâtiments dans une commune française est accessible avec les plans cadastres fournis par le gouvernement ; l'architecture d'un bâtiment est disponible si l'on dispose des plans ; l'itinéraire des bus est généralement disponible en open data dans les métropoles. Toutes ces informations, utilisées pour décrire le contexte géospatial d'une simulation urbaine sont autant d'éléments permettant d'en améliorer le caractère réaliste

Le réalisme des populations modélisées est tout aussi important, bien que traité de façon plus ou moins approfondie dans les approches évoquées précédemment, et de nombreux travaux sont disponibles sur la modélisation réaliste du comportement humain [126].

Ces dernières années, la disponibilité croissante de données sur les usagers permet de mieux modéliser les populations synthétiques. Introduites par Beckman *et al.* [17], ces dernières permettent de calibrer une large population d'individus pour la simulation. Les populations synthétiques sont régulièrement utilisées

dans les modèles LUTI pour augmenter le réalisme des populations en renseignant le domicile, le métier ou encore l'âge des agents dans la population. Il existe plusieurs méthodes pour générer de telles populations à partir des données collectées [175] comme les synthétiseurs de populations [201] qui génèrent automatique des données ou la distribution et l'ajustement de données existantes [92]. Nos travaux utilisent des populations synthétiques en considérant que la Smart City fournit et structure les données nécessaire pour la calibration, sans avoir à se soucier des outils nécessaires pour construire les populations.

Le niveau souhaité de la modélisation détermine également la qualité du réalisme attendu. Par exemple, une simulation microscopique de trafic routier va prendre en compte les actions individuelles, parfois irrationnelles, que peuvent effectuer les humains [96] alors qu'une simulation globale du trafic n'a pas besoin de la distinction de comportement entre les individus, seul le résultat émergent au niveau global considéré compte. Il est ainsi nécessaire de distinguer le réalisme du comportement de l'agent, qui dépend de la granularité de la simulation, du réalisme de la simulation de manière générale, c'est-à-dire la qualité de modélisation de l'environnement et des interactions entre ses composantes. Les populations synthétiques par exemple permettent de proposer des déplacements réalistes entre un lieu de travail et la résidence d'un agent en prenant en compte les trajectoires moyennes, issues des données démographiques sur les déplacements des usagers d'une zone particulière (enquêtes d'origine destination par exemple). La limite des approches à base de population synthétiques est qu'elles se basent sur des données agrégées pour définir les agents. Prenons l'exemple des trajectoires de déplacement précédent, nous aurons 20% des agents qui viendront du centre-ville et 80% qui viennent de la périphérie pour se rendre à leur lieu de travail en centre-ville. Ces données utilisées sont agrégées et représentent un comportement moyen alors que nous voulons disposer d'une simulation exhibant des comportements individuels. Il devient alors difficilement possible d'ajuster à la main le comportement de chaque agent individuel mais il est envisageable d'introduire des variances autour des moyennes dans les données démographiques.

Une solution pour accroître le réalisme est de piloter l'agent virtuel par un utilisateur humain. La communauté utilise la notion d'*avatar* pour décrire un agent contrôlé par un utilisateur humain et pouvant interagir avec d'autres agents, avatars ou non présent dans la simulation. Ainsi, en 2007, Ishida *et al.* proposent un design participatif [113] en 2 phases : la *simulation participative* où agents virtuels et avatars interagissent au sein de la simulation, et l'*expérience augmentée* consistant à réaliser l'expérience avec des humains puis de la simuler à l'aide d'une modélisation multi-agents. Dans la phase de simulation participative Ishida *et al.* font coopérer les usagers directement avec les agents présents et permet de disposer d'un environnement virtuel pour l'éducation et l'entrainement à des tâches comme l'évacuation de bâtiments. Dans notre approche, nous ne visons pas l'interaction directe avec les agents usagers lors de la simulation, mais plutôt la co-conception de politiques entre notre système et le décideur. L'interaction est ainsi envisagée lors de l'élaboration du problème, pendant la simulation en suggérant des modifications, et après la simulation pour évaluer les points forts et points faibles de la politique proposée par le système. Cette démarche relève de la modélisation participative [44, 96], qui consiste à solliciter les usagers pour concevoir ensemble, le processus de simulation. Elle est régulièrement utilisée dans la simulation multi-agents, car elle aide à structurer la pensée, à comprendre les dynamiques et permet aux différents intervenants (par exemple le système et l'utilisateur) de mieux se

comprendre mutuellement. Ainsi, le recours à la modélisation participative dans le cadre de nos travaux nous semble judicieux pour l'étape de co-construction des simulations. Lorsque l'on s'intéresse à la co-conception de politiques, on découvre une limite des approches de modélisation urbaine, même participative : l'interaction avec le décideur se résume le plus souvent à une interface permettant de modifier certains paramètres, rejouer un scénario ou changer le rythme de la simulation.

Les différentes approches exposées ici permettent de calibrer une population synthétique d'agents artificiels et d'augmenter le réalisme de la simulation à la fois au niveau de l'environnement et du comportement des agents. Couclelis, en 2001, précise que l'un des challenges de l'ABM est de modéliser des prises de décisions dynamiques et complexes pour interagir dans un environnement lui-même dynamique et complexe et avec les autres agents [57]. La notion de réalisme est importante pour garantir la pertinence d'une modélisation urbaine mais est encore un challenge de la communauté. Le réalisme dépend également des problèmes envisagés et peut conduire à des modélisations très spécifiques qu'il devient impossible de généraliser à d'autres configurations (représentation très microscopique du comportement des automobilistes à un carrefour pour disposer ensuite d'une simulation de trafic à l'échelle d'une ville).

3.3.3 Que penser des modèles génériques ?

Les contributions sur la modélisation urbaine multi-agents peuvent être scindées en deux grandes catégories : les approches proposant des modèles ou des structures génériques, et les approches modélisant le système urbain relativement à un problème spécifique. Chaque catégorie possède ses avantages et ses inconvénients. Construire des modèles génériques est plus complexe à élaborer, plus difficile à utiliser et à valider. Avec les modèles génériques il est par exemple souvent nécessaire de disposer de connaissances expertes pour créer une instance de ce modèle [57]. Un autre risque avec un niveau de généricité trop élevé est que certaines parties du modèle doivent être simplifiées pour pouvoir prendre en compte un grand nombre de situations. D'une façon générale en modélisation, un compromis est à trouver entre niveau de précision et généricité du modèle.

Dans le cadre de notre travail, nous nous sommes intéressés à l'aspect générique de la simulation afin de pouvoir envisager un grand nombre de situations de politiques urbaines. Dans notre travail, la généricité concerne deux aspects : le modèle de la zone urbaine étudiée et les politiques. Le modèle proposé pour représenter l'environnement urbain est générique dans le sens où il est construit à partir des données géographiques disponibles en open data (par exemple l'usage des données de Open Street Map). Pour les politiques urbaines, nous avons élaboré un formalisme générique qui prend en compte le périmètre, l'objectif, l'horizon de temps, etc. (décrit dans le chapitre 4), ce formalisme étant une des contributions de la thèse. En nous inscrivant en aval de la Smart City, notre objectif est de réaliser une preuve de concept d'un outil d'aide à la décision avec ce modèle générique pour assister le décideur dans la construction et la simulation d'un vaste échantillon de politiques urbaines. Avec la mise à disposition de nombreuses données, notre contribution permettra de simplifier l'intégration des flux d'information venant de la Smart City, qui seront ainsi collectés et traités pour alimenter notre modèle de manière continue avec l'objectif de permettre l'adaptation dynamique des politiques.

La prise en compte de la dynamique pose le problème du couplage entre les modèles. Pour notre système,

nous nous intéressons au couplage des dynamiques du simulateur et du modèle de prise de décision conduisant à modifier l'environnement de la simulation. Un système complexe met en jeu plusieurs dynamiques, agissant rétroactivement l'une sur l'autre, qui nécessitent d'être couplées pour pouvoir étudier le système en tant qu'entité holistique.

3.4 Interactions entre différentes dynamiques spatio-temporelles

La difficulté de modélisation des systèmes complexes découlent en partie de l'existence de différents niveaux d'organisation qui interagissent entre eux. Par conséquent, sa modélisation doit à la fois représenter ces dynamiques aux spécifiques temporelles et spatiales différentes mais doit également capturer les interactions qu'il existe entre ces dynamiques.

En 2012, Crooks *et al.*[58] proposent une synthèse sur les approches existantes utilisant le couplage (*coupling*) de simulations. Le couplage de modèles est né du constat que les outils GIS ont généralement des difficultés à manipuler de grandes bases de données ou d'itérations de simulations. Pour pallier ce problème, l'idée du couplage est de passer par un script ou logiciel externe qui permet de mieux contrôler l'exécution de la simulation [153]. Elles peuvent être définies par la combinaison de deux systèmes indépendants via l'échange de données.

Cette section aborde différentes stratégies adoptées pour combiner ensemble des environnements de structures variées, d'un point de vue organisationnel, temporel ou spatial en considérant les simulations multi-niveaux et le couplage de dynamiques.

3.4.1 Simulations multi-niveaux

On peut dire que les approches usuelles de simulation multi-agents reposent sur deux niveaux, un premier niveau microscopique et un second niveau macroscopique, mais des approches comme celle de Gaud *et al.* exploitent un nombre variable de niveaux d'abstractions dans la simulation [80, 81]. La simulation multi-niveaux ou approche hiérarchique apparaît dans les années 1980 pour répondre à l'aspect statique des simulations multi-agents de l'époque [82]. Ainsi, le premier niveau est constitué de plusieurs entités qui peuvent être regroupées pour former une entité de niveau supérieur. Dans le domaine multi-agents, les travaux portant sur la modélisation multi-niveaux sont principalement liés à l'étude des phénomènes d'émergences [166, 81]. Par exemple, les travaux de Servat [216] représente une simulation de fluides comme une simulation multi-niveaux où les molécules d'eaux (les agents) peuvent se regrouper ensemble pour former une mare ou un étang, pour ensuite se désolidariser formant des flaques lors d'une évaporation. On peut parler de la simulation multi-niveaux comme d'une interaction verticale où la simulation de niveau supérieur agrège et utilise les niveaux inférieurs. L'avantage d'un simulateur multi-niveaux est de pouvoir intégrer plusieurs niveaux d'agents et d'environnements [68] ou de pouvoir avoir des simulations de plusieurs niveaux dans un même environnement [82].

Les problématiques liées au cadre multi-niveaux portent sur le problème de structuration des entités ou d'emboîtement du premier niveau hiérarchique avec un niveau suivant [171].

3.4.2 Couplage de dynamiques

Un système complexe nécessite de considérer plusieurs aspects de dimensions variées. Nous sommes un lundi matin et un navetteur part en retard de chez lui et adopte une conduite plus agressive qu'à son habitude pour se rendre à son lieu de travail. Ce même matin, un second navetteur se rend également à son lieu de travail et conduit, comme à son habitude, un peu au-dessous de la vitesse autorisée. Nous pouvons imaginer une modélisation microscopique fine d'une ville où le comportement, les habitudes de conduites, le modèle de la voiture est renseigné pour chaque agent. La population hétérogène qui en résulterait interagit dans l'environnement et l'impacte de façon globale, notamment par l'émission de gaz lors de la circulation. Imaginons maintenant qu'un décideur politique s'intéresse à la pollution dans la zone urbaine où notre conducteur en retard s'énerve derrière le second navetteur. Nous avons ici une observation macroscopique de l'environnement. Le niveau macroscopique a une représentation agrégée du niveau microscopique, mais chaque élément individuel y participe, à l'image du comportement individuel du navetteur qui n'a que peu d'impact mais avec l'ensemble des navetteurs fournit des informations supplémentaires au niveau macroscopique.

Cette interaction entre le niveau microscopique et le niveau macroscopique représente donc deux modèles, donc deux simulations, avec des granularités et des objectifs différents. La temporalité des 2 mondes, micro et macro, est généralement différente. L'une des simulations évolue toutes les secondes alors que la modélisation macroscopique collecte les données de pollution par tronçon toutes les 10 minutes. Le couplage de dynamiques (ou simulation multi-modèles) consiste à combiner une simulation multi-agents à une autre simulation du même type ou représentant une dynamique différente (automates cellulaires, systèmes multi-agents, GIS, équations) [188, 47]. Il existe également l'approche de co-simulation qui s'intéresse à l'usage de différents simulateurs et à leurs communication entre eux [194].

Les premiers couplages entre les simulations apparaissent avec les automates cellulaires [55] et se limitent généralement à la mise en place d'un modèle microscopique générant des phénomènes observés au niveau macroscopique [37]. Plus tard, les travaux sur le couplage de dynamiques permettent un échange entre les dynamiques [60] en proposant des agents ayant une inter-influence sur les autres. Dans ce cadre, le terme d'*interaction horizontale*, par opposition à l'interaction verticale précédente, est utilisé car chaque dynamique est influencée par ces agents sans notion de hiérarchie. L'intérêt du couplage est de pouvoir ajouter des simulations à d'autres simulations en considérant qu'elles agissent comme des boites noires les unes pour les autres et fournissent un certain nombre d'entrées et de sorties qui sont leur moyen de communication [47]. Dans la cadre où le couplage est effectué entre deux simulations multi-agents, le couplage de dynamiques peut être temporel (la seconde contre la dizaine de minutes) ou spatial (perceptions locales de l'environnement pour nos navetteurs et données de capteurs au niveau d'un quartier pour le décideur). On y retrouve l'aspect du couplage abordé précédemment dans la mesure où le multi-modèle exhibe plusieurs propriétés, notamment celles de pouvoir être conçu comme une brique logicielle logique pouvant être interfacé avec un autre modèle et réutilisé [8, 39].

3.4.3 Identification des propriétés émergentes

Les précédentes approches mentionnées disposent de granularités microscopiques et macroscopiques. L'ensemble des comportements individuels et leurs interactions font émerger de nouveaux phénomènes [69, 216] (figure 3.2).

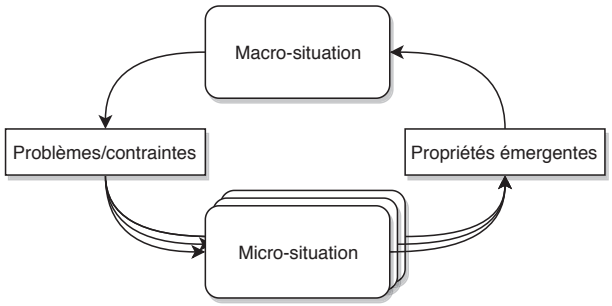


FIGURE 3.2 – L'interaction des agents entre eux et avec l'environnement déclenchent des propriétés émergentes au niveau microscopique. L'ensemble de leurs actions produit des macro-situations pouvant impacter l'agent au niveau microscopique. Figure inspirée de Ferber [75] (p.87)

C'est le cas par exemple de la génération de pollution qui va varier en fonction du départ des navetteurs (traduisant la notion d'heures de pointe). Par exemple, les heures de départ similaires de nombreux agents ayant pour conséquence l'apparition de bouchons faisant augmenter le niveau de pollution. Il est donc nécessaire d'identifier ces phénomènes émergents et leurs sources pour les capter au niveau macroscopique. Ensuite, le niveau macroscopique peut y répondre et tenter d'apporter des solutions (figure 3.3). Les propriétés émergentes peuvent être souhaitées par le système, au niveau microscopique, ou être une conséquence de l'interaction des agents avec leur environnement et créer des problèmes (congestion de trafic et sa résolution par exemple). Dans le cadre de la thèse, l'observation des propriétés émergentes négatives et l'identification des problèmes que cela génère sur la population microscopique, font partie des objectifs de notre étude.

La réponse à ces problèmes peut se traduire par une séquence d'actions à appliquer à l'environnement, qui traduit la politique urbaine sous-jacente. Pour reprendre notre exemple, les navetteurs en allant au travail vont créer une congestion et augmenter localement la pollution. Ce problème peut être identifié par l'analyse de la vitesse moyenne des usagers sur les tronçons, le modèle de véhicule, le type de conduite. À partir des capteurs de la dynamique macroscopique, un agent politique utilise des actionneurs adaptés pour résoudre ce problème (pénalisation du type de conduite, agrandissement de la voirie, possibilité de dépasser plus facile).

Ainsi, la figure 3.3 représente l'identification et la résolution de problèmes permise par la simulation multi-niveaux. Il est alors possible de traduire l'objectif du décideur comme l'identification de problèmes émergents au niveau macroscopique de la simulation.

3.4.4 Verrous de la communauté

Morvan propose trois questions liées à l'utilisation de la simulation multi-niveaux [171] :

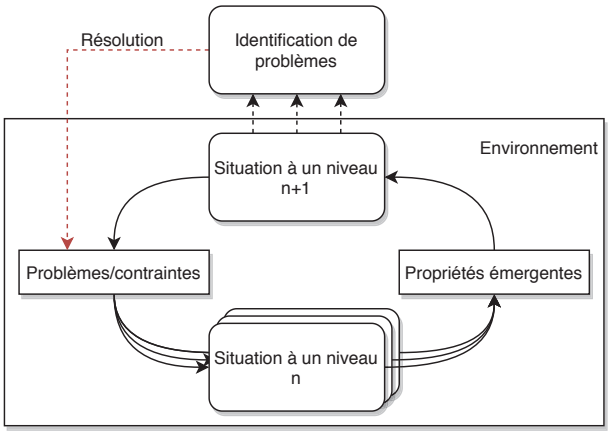


FIGURE 3.3 – Représentation générique du problème où les agents situés dans l'environnement, à un niveau quelconque (micro, macro), génèrent des propriétés émergentes observables à un niveau supérieur.

— l'établissement de méta-modèles génériques et de moteurs de simulations ;
— la détection et la réification de phénomènes émergents ;
— la définition de représentations génériques pour les entités agrégées.

Dans le cadre de cette thèse, nous nous intéressons aux deux premiers points dans le cadre de la simulation urbaine. Le troisième aspect, lié aux problématiques d'emboitement mentionnées précédemment ne nous concerne pas dans le cadre d'un couplage de dynamiques.

Le premier verrou que nous considérons est donc la détection de phénomènes émergents dans le modèle. Dans un cadre de conception politique, l'identification de phénomènes émergents à partir de données collectées de la ville est déjà une aide conséquente pour le décideur politique. La réification de ces phénomènes est actuellement l'étape de construction politique auquel les décideurs et les parties prenantes sont confrontés. La modélisation fournit un support pour d'une part faire émerger ces propriétés et d'autre part tester des actions politiques et observer leurs conséquences sur les problèmes identifiés. Afin de répondre au verrou, l'approche adoptée doit permettre à la couche macroscopique d'interagir avec la couche microscopique pour résoudre ces problèmes.

Le second verrou abordé dans le cadre de cette thèse est l'établissement d'un modèle générique pour concevoir des instances d'environnements et les politiques associées. La mise en place d'un outil autonome requiert de pouvoir structurer n'importe quelle zone urbaine dans un méta-modèle produisant une instance de simulation multi-agents avec un environnement cohérent permettant des interactions d'agents réalistes, et la mise en œuvre de politiques adaptatives pertinentes. Cette notion est d'autant plus importante dans le cadre de notre étude où le couplage de dynamiques requiert une autonomie aux deux niveaux de notre couplage micro/macro.

À notre connaissance, il n'existe pas de travaux dans le cadre de la modélisation urbaine qui représente une véritable interaction dynamique entre la modélisation de l'environnement et la construction d'une politique

satisfaisante.

3.5 Conclusion

La conception de politiques urbaines repose sur la modélisation d'un système complexe aux dynamiques multiples. À travers ce second chapitre d'état de l'art, nous nous sommes intéressés au recours de la simulation multi-agents pour construire ces politiques. Notamment avec l'intérêt de la simulation multi-agents pour la modélisation d'une zone urbaine et développer le besoin d'un modèle couplant plusieurs dynamiques pour mieux capter la complexité d'un système complexe comme une ville. Nous nous situons ici dans un contexte où le décideur politique cherche un outil l'aidant à faire sens des nouvelles données disponibles de la Smart City. En proposant un modèle générique pour la construction d'un moteur de simulation, notre contribution porte sur les verrous liés au couplage de dynamiques et aux approches multi-niveaux.

Dans le cadre de cette thèse, le couplage de dynamiques permet une temporalité différente des simulations et l'approche multi-niveaux consiste à combiner deux simulations représentant deux systèmes différents : le premier est une simulation individu-centrée avec la description des interactions des agents ; le second système prend en charge l'émergence des problèmes relatifs aux objectifs étudiés, et a pour objectif de répondre aux macro-problèmes identifiés en capitalisant les observations émergentes de l'environnement. L'intérêt dans l'approche multi-niveaux choisie est de pouvoir différencier des dynamiques de granularités spatiales et temporelles différentes. Ces deux notions sont essentielles pour aborder la simulation multi-agents comme approche distribuée pour la représentation et la modélisation de systèmes complexes, notamment les villes, pour la construction de politiques urbaines. Enfin, les approches de la littérature permettant de doter la simulation d'un moyen de contrôle sur les phénomènes émergents de la simulation avec la notion de multi-niveaux et de couplage de dynamiques ont été analysées. Ces différentes approches motivent le besoin de disposer d'un outil générique de conception de politiques permettant à la fois de représenter la dynamique microscopique des agents dans l'environnement et également de représenter les outils et actions permettant de proposer des solutions pertinentes dans l'environnement et d'en observer les conséquences.

À travers l'état de l'art sur les approches existantes pour la modélisation urbaine en utilisant des systèmes multi-agents, nous avons identifié les verrous liés à la mise en place du couplage de dynamiques multi-agents de modèles génériques. Une contribution à ces verrous pour les systèmes complexes modélisant la Smart City porte sur un choix d'architecture avec un couplage de dynamiques multi-agents, capable d'intégrer des temporalités différentes au niveau des dynamiques et permettant une vision globale de la zone urbaine. Notre objectif à travers cette preuve de concept est de réduire les connaissances expertes requises, que ce soit au niveau de la modélisation ou au niveau politique, pour établir des politiques urbaines co-construites à partir de simulateurs multi-agents.

Chapitre 4

Conception de politiques urbaines avec SmartGov

Sommaire

4.1 Introduction

Les deux précédents chapitres d'état de l'art permettent de voir l'intérêt de la simulation multi-agents et le couplage de dynamiques pour la co-construction de politiques. Une première dynamique doit représenter les comportements individuels des utilisateurs et leurs interactions et une seconde dynamique observe les conséquences à un niveau plus global pour déterminer les actions politiques qui peuvent être pertinentes à appliquer.

L'élaboration de politiques dans le cadre de la Smart City motive le besoin de disposer de nouveaux outils pour intégrer continuellement un vaste flot de données pour construire des politiques réactives et pertinentes. Nous utilisons la simulation multi-agents dont l'intérêt repose notamment sur les simulations individu-centrées pour représenter les objectifs individuels d'entités comme les acteurs de la politique pour construire un environnement réaliste dans lequel ils peuvent évoluer et ainsi disposer d'un espace sur lequel éprouver des politiques en étudiant le retour des agents présents. Ainsi, les deux précédents chapitres ont introduit les problématiques relatives au domaine d'élaboration de politiques de décisions et à la simulation multi-agent. L'objectif maintenant est de relier ces deux domaines afin de produire un outil d'aide à la décision pour le décideur politique. La particularité de cet outil est d'évaluer des politiques urbaines dans un environnement virtuel réaliste et de proposer de nouvelles politiques ou des ajustements sur les existantes.

Ce chapitre introduit la première contribution de la thèse avec la proposition d'une architecture générique pour la co-construction de politiques, *SmartGov*, basée sur le couplage de dynamiques entre un niveau microscopique (espace d'application de la politique) et d'un niveau macroscopique (espace d'élaboration de la politique). Les deux niveaux ont donc leur propre dynamique et disposent chacun de leur propre simulation multi-agents.

Notre contribution se décline ainsi en plusieurs composantes :

— l'architecture générique de SmartGov qui décrit :
  — l'environnement simulé de la zone urbaine avec la couche microscopique ;
  — la gestion de politiques définie dans la couche macroscopique ;
  — un modèle de politique structurant les interactions entre le décideur et SmartGov ainsi que la production de SmartGov ;

— le couplage entre deux simulateurs multi-agents pour permettre la conception de politiques urbaines ;
— la définition d'un formalisme pour l'expression des politiques ;
— la proposition d'un outil complet pour la co-conception de politiques.

Ce chapitre se découpe donc de la manière suivante. Dans un premier temps, nous apportons une description de l'architecture de référence avec le formalisme de chacune des briques qui la composent. Nous les décrirons de manière générique. La généricité de la couche microscopique a été particulièrement travaillée pour pouvoir prendre en compte un grand nombre d'environnements dont on a abstrait les principales caractéristiques. De la même façon, le gestionnaire d'agents politiques repose sur un cadre défini de façon très générique, permettant au décideur de calibrer précisément les éléments qui entrent en compte dans la décision politique.

Nous expliquons ensuite comment se déroulent les interactions pour permettre la co-conception de la politique. Le couplage entre les niveaux microscopiques et macroscopiques est alors décrit à partir d'exemples simples. Enfin nous détaillons les données requises pour définir les entrées du modèle générique, ainsi que les supports proposés pour la visualisation du paramétrage, du déroulement de la simulation et des résultats.

Soumissions L'architecture formelle SmartGov, présentée dans ce chapitre, a permis la publication :
— d'un article à la *IEEE Internationcal Conference on Tools with Artificial Intelligence* (ICTAI), présenté en 2017 à Boston aux États-Unis [189] ;
— d'un article aux *Journées Francophones de la Simulation Multi-Agent* (JFSMA), présenté en 2018 à Metabief en France [190].

4.2 Motivation

Nous proposons de décrire l'architecture SmartGov à l'aide d'un exemple applicatif suffisamment complet pour faire émerger les concepts nécessaires à la création d'une politique urbaine et d'en abstraire un formalisme cohérent. Nous construirons ensuite une instance de ce scénario pour montrer l'utilité de notre approche générique.

4.2.1 Introduction à la conception de politique

Dans le contexte de la Smart City, nous considérons un décideur souhaitant mettre en place une politique dans une zone urbaine spécifique. Il dispose ainsi de données de capteurs collectées automatiquement, des enquêtes sur le comportement et les motivations des usagers ainsi que des sondages relatifs au problème qu'il étudie. Cependant, le décideur n'est pas certain du résultat qu'apportera la nouvelle politique et décide donc peut-être de faire des essais avant d'appliquer la politique envisagée. Une méthode peut être de considérer une zone d'essai où appliquer la politique : par exemple sur un quartier donné, augmenter certains tarifs de stationnement et en baisser d'autres.

Le décideur définit les critères d'évaluations qu'il considère pour construire et évaluer sa politique. En fonction de ses objectifs, il cherchera à satisfaire les usagers ou d'autres intérêts. Il peut, par exemple,

s'intéresser au temps moyen qu'il faut aux conducteurs pour trouver un emplacement de stationnement dans le centre-ville ou encore étudier les revenus que les emplacements génèrent. Plusieurs méthodes sont disponibles pour évaluer la performance de la politique. L'une d'entre elles porte sur l'intégration des acteurs et usagers dans le processus de conception. Notamment, le décideur peut collecter l'avis des usagers de la zone de test, qu'ils soient directement disponibles (par exemple sur les réseaux sociaux), ou récupérables par des actions spécifiques (discussions dans les forums organisés par la ville ou auprès des conseils d'arrondissements).

Le décideur envisage ensuite d'étendre ou non la politique sur une plus grande zone urbaine. Même si un essai local est pertinent, il est possible que la politique appliquée à la zone complète ne convienne pas. Dans ce cas, le décideur risque d'être contraint de modifier voire d'annuler la politique initialement prévue. Cet échec est coûteux pour le décideur, à la fois en temps et en argent pour la ville comme pour les usagers. De plus, ces derniers pourront être plus réticents sur les nouvelles propositions de politiques.

Cette illustration montre l'utilité pour les décideurs politiques d'outils permettant d'éprouver les politiques urbaines sur des environnements artificiels réalistes. La simulation préalable leur permet de mesurer l'impact des politiques, de les ajuster et de tester différentes configurations avant de les implémenter dans le monde réel.

4.2.2 Les informations requises pour construire le monde virtuel

Prenons l'exemple d'un décideur politique souhaitant élaborer une politique tarifaire pour les *emplacements de stationnement sur voie* (en anglais *on-street parking*) dans plusieurs quartiers de sa ville. Par soucis de concision, les emplacements de stationnement sur voie seront appelés *emplacements*.

Le fait que les usagers passent un certain temps à chercher un emplacement génèrent de la pollution et de l'insatisfaction. Le décideur réalise ainsi que la motivation sous-jacente de sa politique tarifaire est d'assurer une bonne attribution des emplacements avec un juste prix, et ainsi réduire le trafic aux heures de pointes.

Le décideur politique doit donc, pour obtenir une simulation réaliste de la situation, être en mesure de fournir plusieurs éléments : une description du réseau routier de la ville ainsi que les informations sur la géo-localisation et la taille des différents emplacements ; les bâtiments environnant pour identifier les zones attractives pour les usagers (restaurants, commerces) ; et si possible les matrices origine-destination représentant les trajets entre le domicile et le lieu de travail des usagers.

Ces informations servent à définir l'environnement cible de la politique, ou *périmètre*. Il inclut les différentes *structures* représentant des éléments de l'environnement, comme les emplacements qui pourront être modifiés par la politique tarifaire construite. La description des *structures* comprend également des informations sur leur *type* (par exemple un bâtiment résidentiel ou un bâtiment de travail). Les emplacements ainsi que les bâtiments sont décrits par un ensemble d'informations qualitatives et quantitatives : les heures d'ouvertures, le tarif, le nombre de résidents ; dont les valeurs dépendent du type d'élément. Ces informations qualitatives et quantitatives font office de *perceptions* et renseignent des *types* spécifiques de *structures*.

Une fois le périmètre décrit, le décideur fourni une description réaliste des usagers, principales cibles de la politique tarifaire. Il va proposer une représentation des travailleurs pendulaires, à partir de collecte de données obtenues par des sondages, des questionnaires en ligne ou des applications spécifiques sur smartphone.

Le réalisme des agents composant la population est difficilement objectif et des hypothèses peuvent être faites à partir d'études sociales, définissant des profils types associés au domaine d'étude (par exemple dans notre cas, les profils de conducteur). Les données récoltées sont mises en correspondance avec les comportements types pour exprimer les *profils* d'usagers.

L'ensemble de ces informations, c'est-à-dire la description des structures, du réseau routier ainsi que celle des usagers, permet de construire un *environnement* virtuel dans lequel les *agents* vont 'évoluer'. La simulation multi-agents individu-centrée est donc particulièrement adaptée pour modéliser l'environnement et les usagers. Dans notre approche, les travailleurs pendulaires sont considérés comme des agents rationnels et utilitaristes. Chaque *profil* a ses propres préférences vis-à-vis de la politique considérée : par exemple l'usager *économe* qui acceptera de passer plus de temps qu'un usager *en retard* pour trouver un emplacement intéressant. Ces préférences peuvent être calibrées par le décideur en fonction du *périmètre* étudié. Une distribution de profils représente une *population* donnée.

À partir de ce point-là, le décideur politique dispose de la description d'un cadre sur lequel il va pouvoir appliquer une politique. Nous nous intéressons dans la suite de ce chapitre au formalisme proposé pour décrire et tester les politiques.

4.2.3 Support pour la mise en place d'actions politiques

Le décideur politique souhaite une politique satisfaisant dans la mesure du possible l'ensemble de ses *besoins* ou de ses *contraintes*. Le décideur politique exprime alors ses attentes à l'aide d'une *fonction objectif* mono-objectif unique, cette fonction intégrant différents critères estimés représentatifs afin d'obtenir une proposition de politique pertinente. Par exemple, le décideur peut avoir pour fonction objectif de minimiser la pollution et le nombre d'utilisateurs se rendant en centre-ville. Il va associer à chacun de ces deux critères des poids spécifiques agrégés pour produire une fonction mono-objectif.

Dans notre modélisation, chaque modification de l'environnement effectué par le décideur représente une *action politique*. Pour notre exemple, une *action politique* peut être de modifier la tarification des emplacements d'une zone donnée ou d'ajouter des emplacements de stationnement sur une zone très fréquentée. À partir de la fonction objectif, la réaction des usagers simulés permet au décideur d'évaluer l'impact d'*actions politiques* sur les *performances* de la politique. En analysant la réaction des usagers, le décideur politique évalue la pertinence des actions politiques qui ont été proposées. Il peut alors ajuster sa politique en proposant de nouvelles actions et ainsi construire la politique urbaine avec les nouvelles réactions observées. Nous envisageons également que les parties prenantes puissent réagir aux suggestions politiques. Dans le cas des emplacements, il est ainsi possible de choisir d'augmenter le tarif des emplacements dans certains quartiers et de le baisser dans d'autres (politique appliquée), dans le but de diminuer le temps de recherche d'un emplacement pour les usagers (fonction objectif à optimiser).

Une *politique urbaine* va donc se traduire par une *séquence d'actions politiques* portant des informations sur où et quand appliquer chaque action de la séquence. Comme une *politique urbaine* peut en final se révéler contre-productive, une aide au décideur est nécessaire pour continuellement ajuster les actions afin de s'assurer du succès de la politique. En utilisant des techniques d'apprentissage, notre modèle inclut l'évaluation

automatique d'actions politiques pour assister le décideur dans la recherche de politiques pertinentes.

FIGURE 4.1 – *Agents* microscopiques (triangles noirs) cherchant un emplacement dans la zone urbaine. Les emplacements disponibles (carrés gris clair) et occupés (carrés gris foncé) ainsi que les lieux de travail (gris foncé) et les résidences (gris clair) sont des *structures* composant le *périmètre*.

Un autre aspect de la modélisation réaliste et dynamique est celui du passage à l'échelle. Même en considérant une zone urbaine restreinte, par exemple celle de la figure 4.1, l'ensemble des politiques tarifaires à considérer pour chaque emplacement est trop conséquent pour évaluer toutes les alternatives possibles au niveau de cette zone. Passer à l'échelle d'une métropole paraît donc difficile sans un outil adapté. L'évaluation de chaque politique représente un coût non négligeable pour le décideur et des techniques d'optimisation doivent donc déjà être employées pour effectuer un choix parmi ces alternatives. Nous proposons un deuxième niveau superposé à la description des agents et de l'environnement, également constitué d'une simulation multi-agent agrégeant les structures en ensemble homogènes pour réduire le nombre de configurations à étudier. Celle-ci est couplée à la première et est composée d'agents politiques locaux, qui comme les agents microscopiques, interagissent sur une partie restreinte de l'environnement. Alors que les agents microscopiques évoluent sur le réseau en fonction de différentes contraintes, les agents politiques gèrent chacun un périmètre local de la ville, composé d'un ensemble de structures, et définissent les actions à mener en cohérence avec une politique donnée.

4.3 Architecture globale de SmartGov

L'exemple de motivation précédent nous permet de présenter les principaux concepts de SmartGov, l'architecture générique proposée pour la conception de politiques urbaines. L'objectif de SmartGov est de proposer un outil d'aide à la décision générique à destination des décideurs politiques. Cet outil d'aide à la décision a pour objectif de proposer les fonctionnalités suivantes :

- **Co-construction de la politique :** SmartGov doit faciliter la construction d'une politique urbaine en diminuant les compétences expertes requises pour le développement et la mise en place de simulations. Le décideur politique utilise les données à sa disposition pour décrire la zone urbaine qu'il souhaite étudier. Il paramètre ensuite l'outil pour que celui-ci propose des politiques.
- **Validation de la politique :** Avec SmartGov, le décideur est en mesure d'éprouver les actions politiques sur une zone urbaine et d'observer leur impact. Il permet ainsi de valider la pertinence de la politique dans la mesure où la couche microscopique est suffisamment réaliste. Le décideur politique utilise l'outil pour observer l'évolution de la zone urbaine avec la fonction objectif et les actions politiques qu'il mentionne. L'outil est également en mesure de produire par lui-même des politiques urbaines, soumises pour validation au décideur.
- **Modification de la politique :** Un point important dans notre architecture est que la co-construction permet notamment de construire une instance de la zone urbaine avec le décideur mais aussi au décideur politique de modifier directement une politique existante et d'en observer les impacts.
- **Visualisation de la politique :** SmartGov, dans sa version instanciée, doit permettre au décideur d'observer le déroulement d'une simulation de la zone urbaine et la conséquence des actions politiques.

L'outil est à destination du décideur politique, qui fait le choix de prendre en compte ou non de la satisfaction des usagers dans le processus de co-construction.

L'architecture générique de SmartGov (figure 4.2) se décompose en deux grandes briques conceptuelles.

La première, dite *couche microscopique*, porte sur la représentation d'un environnement réaliste et permettant de simuler l'évolution d'agents microscopiques. Ceux-ci, sensibles à leur environnement, vont interagir avec ses éléments et le modifier en conséquence. Ainsi, l'environnement contient donc les éléments nécessaires permettant la boucle sensorimotrice des agents microscopiques. L'interaction entre l'environnement et les parties prenantes concernées nous intéresse particulièrement pour concevoir les politiques urbaines. En effet, la réaction des citoyens à leur environnement permet d'évaluer la validité d'une politique urbaine. L'objectif est donc d'être en mesure de capter ces changements et de les quantifier afin de pouvoir établir si une politique est pertinente ou non.

La représentation du comportement d'un humain est complexe puisqu'elle nécessite de faire des hypothèses sur ses réactions et son modèle cognitif. Les paramètres qui entrent en compte lorsqu'il s'agit de prendre des décisions chez l'humain sont très nombreux (préférences individuelles, contexte, ressources financières, etc.). Afin de conserver une simulation crédible, nous avons fait des hypothèses pour représenter l'humain dans la simulation en considérant celui-ci comme utilitariste.

La seconde couche de notre architecture de référence est la couche dite macroscopique qui contient le cœur décisionnel de la politique dans SmartGov. L'objectif de SmartGov étant de produire des politiques

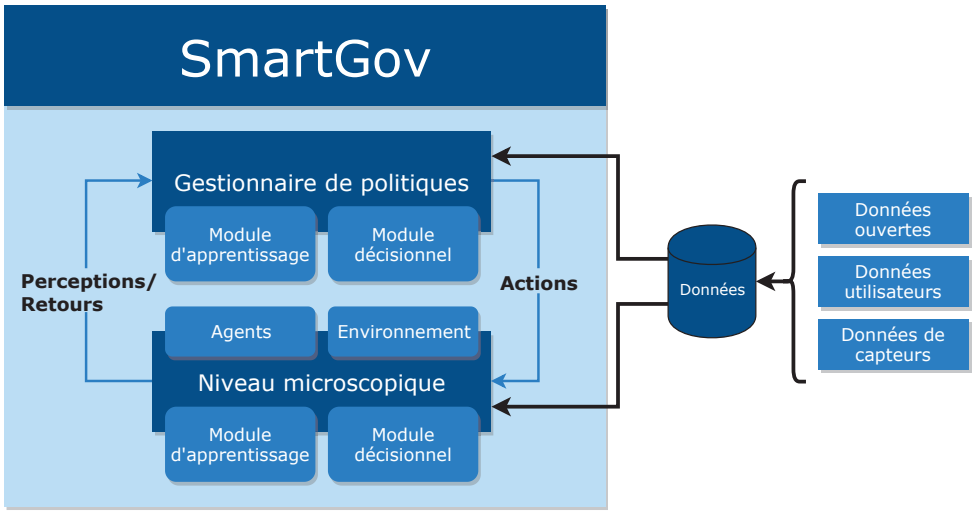


FIGURE 4.2 – Représentation de l'architecture générique SmartGov

urbaines, la couche macroscopique a pour but d'établir des règles de gestions à partir des éléments perçus dans l'environnement et les interactions entre l'agent humain et son environnement.

Nous avons élaboré un formalisme spécifique qui permet de décrire tous les éléments nécessaires pour créer une instance pour chaque couche. La suite de ce chapitre est une description de chaque couche et du formalisme proposé.

4.4 Couche microscopique

La **Couche Microscopique** ou **Couche Usagers**, abrégée par CU, correspondant à la simulation de l'espace urbain et des usagers de cet espace. La **Couche Usagers** est l'un des deux simulateurs multi-agents composant SmartGov et comprend un environnement $\mathcal{E}$ (section 4.4.1) ainsi qu'une population interne d'agents microscopiques N_h [1] (section 4.4.2). La figure 4.3 représente le fonctionnement des agents dans l'environnement de la couche microscopique.

Cette section présente le modèle théorique et générique de la CU en expliquant toutes ses composantes paramétrables, afin de concevoir différents types de simulations pertinentes.

1. Les agents microscopiques peuvent être des agents représentant des humains, on parlera alors spécifiquement d'agents humains pour simplifier.

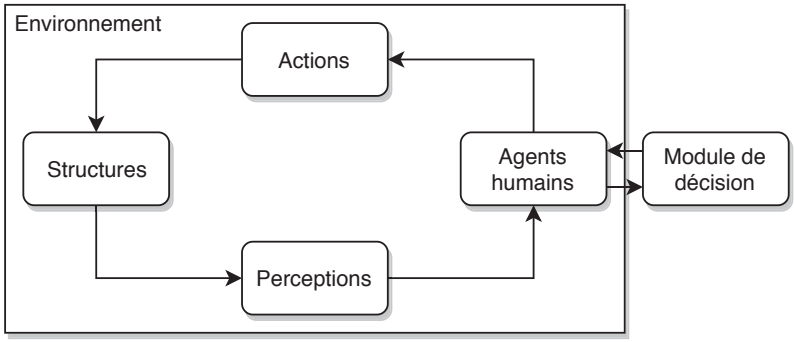


FIGURE 4.3 – Représentation de la Couche Usagers de SmartGov. L'environnement est le support de l'interaction entre ses composantes (*structures*) et les *agents humains*. Les *structures* sont perceptibles par les agents qui utilisent leur *module de décision* pour choisir des actions impactant l'environnement ou plus particulièrement des *structures*.

4.4.1 Représentation de l'environnement

L'environnement $\mathcal{E}$ est le support des boucles sensorimotrices des agents présents dans cet environnement. Il est défini par le tuple suivant :

$$\mathcal{E} = \langle \mathcal{S}, T, I, G, N_m \rangle \tag{4.1}$$

où :

- $\mathcal{S}$ est un ensemble fini de structures ;
- T représente un ensemble fini de types de structures ;
- I décrit un ensemble fini de perceptions ;
- G est le graphe défini par $G = \langle V, AR, \omega \rangle$ tel que ;
  - V est un ensemble fini de sommets ou nœuds (*vertices*) ;
  - $AR = V \times V$ est un ensemble fini d'arc (*arc*) pour le cas orienté, d'arêtes (*edges*) pour le cas non orienté ;
  - ω représente une fonction de poids soit sur les sommets, sur les arêtes ou sur les deux ;
- N_m représente un ensemble fini d'agents microscopiques.

La modélisation proposée pour l'environnement a été conçue de façon générique pour pouvoir représenter un assez grand nombre de zones urbaines, sous différentes formes. Ainsi, outre le réseau routier, il est possible de vouloir représenter la ville par son réseau fluide (canalisations d'eau, gaz) ou son réseau sec (électricité, fibre, etc.). Le modèle de graphe choisi permet toutes ces représentations.

Structures et types

La *structure* joue un rôle important dans SmartGov. Il s'agit de l'élément permettant l'échange et l'interaction entre les différentes couches du modèle. Tous les éléments constituant l'environnement de la Couche Usagers sont considérés comme des structures. Par conséquent, cela signifie que les agents

microscopiques sont également considérés comme des structures. Cette représentation a plusieurs avantages, d'un point de vue ingénierie et d'un point de vue du modèle formel. Une structure peut ainsi être représentée physiquement dans la simulation (exemple avec un emplacement de stationnement), ou également être abstraite (un ensemble d'emplacements formant un front de rue ou groupe de bâtiments) et ne sera pas rendue visible dans la simulation. Par exemple, si l'on considère un front de rue avec plusieurs emplacements de stationnement, la route ainsi que les emplacements de stationnements sont des structures.

Une structure $s_i \in \mathcal{S}$ est définie au minimum par son type $t_i \in T$. Les structures sont regroupées en fonction de leur type et ont un comportement commun. La structure s_i peut disposer d'un ensemble de perceptions disponibles $I_i \subseteq I$ que les agents microscopiques peuvent exploiter, on dit alors que cette structure est *perceptible*. De la même manière, certaines structures peuvent être *actionnables* par les agents microscopiques. La structure définit donc un ensemble fini d'actions A_i dont une partie ou la totalité peut être utilisée par l'agent en fonction de l'état de la structure. Par exemple, un centre commercial s_{centrer} dispose de l'ensemble d'action $A^h_{\text{centre}} = \{entrer, sortir\}$ mais l'agent ne perçoit toujours qu'une des deux actions en fonction de sa position par rapport au centre commercial. Les actions que l'agent peut effectuer sur une structure sont disponibles lorsque l'agent perçoit la structure.

Dans un premier temps, nous décrivons une structure $s_i \in \mathcal{S}$ comme un tuple $s_i = \langle t, I^h_i, A^h_i \rangle$ où I^h_i et A^h_i peuvent être vides. Nous reviendrons à la notion de structure dans la couche macroscopique (section 4.5.1).

Perceptions de l'environnement

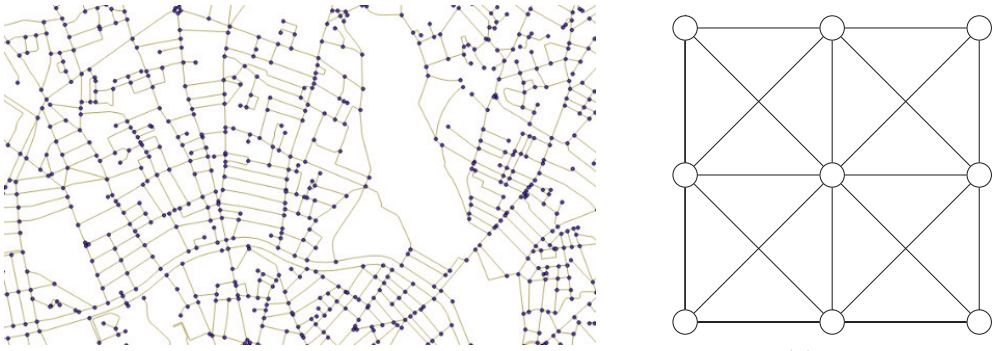
Une perception $i \in I$ quantifie une valeur perçue dans l'environnement. Dans le cadre d'une structure, celle-ci détermine comment les agents microscopiques vont interagir avec elle. L'ensemble des perceptions possibles est fixe. Les perceptions perçues sont choisies par les décideurs politiques, ou par le système lui-même.

Une perception i est caractérisée par sa valeur λ_i définie sur un espace algébrique $\Lambda_i \subseteq \mathbb{R}$. Une perception peut prendre un ensemble de valeurs, qu'il soit continu comme la distance entre un agent et sa destination, ou discret comme le prix d'un emplacement de stationnement dans notre exemple. Considérons l'occupation d'un emplacement de stationnement. Elle peut prendre deux valeurs : *occupé* ou *libre*. Par conséquent, la perception de l'occupation d'un emplacement de stationnement, notée $i_{\texttt{occupation}}$, a pour espace algébrique $\Lambda_{i_{\texttt{occupation}}} = \{\texttt{occupé}, \texttt{libre}\}$. Ainsi, un emplacement de stationnement libre aura la valeur $\lambda_{i_{\texttt{occupation}}} = \texttt{libre}$ pour l'usager.

Si nous considérons maintenant le tarif de cet emplacement, alors la perception du tarif, notée $i_{\texttt{tarif}}$, est définie par l'espace algébrique $\Lambda_{i_{\texttt{tarif}}} = \{0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4\}$. Un agent percevant le tarif d'un emplacement recevra une valeur $\lambda_{i_{\texttt{tarif}}} \in \Lambda_{i_{\texttt{tarif}}}$.

Dans notre modèle, la perception peut correspondre à la valeur d'une donnée d'un élément (comme dans les exemples précédents) ou s'exprimer comme une fonction des valeurs de ses éléments. Le système ne se limite donc pas à une représentation particulière de la donnée mais est capable de modéliser un grand nombre de situations.

Graphe de l'environnement



(a) Réseau routier (b) Grille

FIGURE 4.4 – Représentation d'un réseau routier (a) et d'une grille (b).

Dans notre exemple de motivation, les agents microscopiques sont représentés par des agents humains utilisant la voirie pour trouver un emplacement de stationnement satisfaisant. Dans SmartGov, seraient également possible des entités non humaines pouvant avoir un comportement spécifique comme des bâtiments, ou encore des entreprises. L'utilisation d'un graphe permet de définir une relation entre les différents sommets du graphe : un arc ou arrête entre deux sommets indique ce lien entre eux. Dans le cas d'un graphe représentant le réseau routier, si les croisements sont les sommets, les arcs pourront symboliser les routes qui les relient. Dans le cadre de SmartGov, le graphe G peut prendre plusieurs formes :

— **réseau routier** : les nœuds correspondent à des intersections et n'ont pas d'information autre que leur position géographique (figure 4.4a) ;

— **grille** : les nœuds sont davantage connectés et chaque nœud possède des informations spécifiques. Nous pouvons imaginer ici que la grille représente un réseau de discussion entre usagers et où le nœud représente un usager et les arcs ses connections avec d'autres usagers (figure 4.4b).

La fonction de poids ω permet d'affecter un vecteur de valeurs particulier à un arc ou un sommet. Par exemple, dans le cadre d'un réseau routier, nous pouvons considérer la distance entre deux nœuds, le niveau de congestion du tronçon avec le nombre de véhicules qui s'y trouvent, le temps nécessaire pour le parcourir à la vitesse maximale autorisée. Il est aussi possible de définir un poids agrégeant n critères différents.

Dans le cadre d'un réseau de canalisations alimentant des maisons, la fonction de poids ω pourrait décrire le débit de la section sur l'arc par exemple. Le nœud stockerait alors le nombre de foyers qui y sont rattachés.

4.4.2 Modèle de l'agent microscopique et son fonctionnement

L'agent microscopique a_m^n est décrit par un automate à états fini déterministe paramétrable avec le tuple suivant :

$$a_m^n = \langle \Sigma, S, e_0, F, \delta \rangle \tag{4.2}$$

dont la définition formelle sera donnée dans la partie suivante.

L'agent présent dans l'environnement est générique afin de décrire aussi bien une entité non humaine qu'une entité humaine.

Nous avons imaginé que pour de nombreuses simulations, le décideur aura besoin de représenter des usagers de la zone urbaine simulée afin d'évaluer l'impact de la politique sur leurs comportements. Une représentation crédible de ces usagers donnera plus de sens à la politique urbaine produite. Dans SmartGov, nous effectuons l'hypothèse forte que l'agent humain est rationnel et utilitariste. C'est pourquoi nous avons prévu une modélisation générique des agents humains comme une extension de l'agent microscopique où l'agent humain possède une personnalité propre que le distingue des autres.

Un agent humain a_h^n est donc défini par le tuple suivant :

$$a_h^n = \langle \Sigma, S, e_0, F, \delta, P^n \rangle \tag{4.3}$$

où P^n est la personnalité de l'agent humain. La suite de cette section décrit dans un premier temps comment nous exploitons un automate à états finis pour décrire le comportement de l'agent dans le cadre de SmartGov et comment sont représentés les agents ainsi que leur paramétrage. Puis, il sera enfin abordé l'utilisation et le fonctionnement de la personnalité de l'agent.

Automate à états finis

Un automate est constitué d'états et de transitions entre ses états. Un état spécifique de l'automate décrit donc les états qu'il peut atteindre en fonction de certaines conditions et ainsi qu'un comportement par défaut si aucune condition n'est respectée.

Formellement [224], un automate fini est décrit par le quintuplet $\mathcal{A} = \langle \Sigma, S, e_0, F, \delta \rangle$ où :

- Σ est un vecteur d'entrées décrivant les conditions des changements d'états ;
- S est un ensemble fini d'états ;
- $e_0 \in S$ représente l'état initial de l'automate ;
- $F \subseteq S$ représente l'ensemble des états finaux de l'automate ;
- δ est une fonction de transition entre les états où $\delta = I \times \Sigma \times A \to A$ (I l'ensemble des perceptions et A l'ensemble des actions) ;

Dans le cas de SmartGov, l'automate correspond à une partie du processus décisionnel de l'agent, l'autre partie étant définie par sa personnalité. Les transitions entre états décrivent des actions que l'agent effectue sur son environnement et les états décrivent un comportement spécifique de l'agent.

Les conditions portées sur les transitions sont paramétrables (figure 4.5) pour que le décideur puisse calibrer le comportement des agents présents dans sa simulation.

Considérons un emplacement de stationnement, noté `emp`. Sa réaction à l'environnement peut être décrit par un automate à états finis simple. Il possède deux états : {`libre`, `occupé`}. Dans ce cas, l'emplacement de stationnement réagit aux actions qu'un agent humain effectue sur l'emplacement par l'intermédiaire des actions `ENTRER` et `SORTIR` (figure 4.5). L'état de l'automate n'est pas modifié si aucune action n'est effectuée.

L'automate est décrit par l'ensemble de ses états, les conditions permettant le changement d'état, la fonction de transition, l'état initial et l'ensemble des états finaux. Par conséquent, l'automate de l'emplacement s'écrit :

$$\mathcal{A}_{\texttt{emp}} = (\{\texttt{libre}, \texttt{occupé}\}, \{\texttt{ENTRER}, \texttt{SORTIR}\}, \delta_{emp}, \texttt{libre}, \{\}) \tag{4.4}$$

où δ est décrit par :

$$\delta_{emp}(\texttt{libre}) = \begin{cases} \texttt{occupé} & \text{if } a_h^n \texttt{ ENTRER emp} \\ \texttt{libre} & \text{otherwise} \end{cases}$$

$$\delta_{emp}(\texttt{occupé}) = \begin{cases} \texttt{libre} & \text{if } a_h^n \texttt{ SORTIR emp} \\ \texttt{occupé} & \text{otherwise} \end{cases}$$

Par exemple, la transition $\delta(\texttt{libre})$ peut se lire comme l'emplacement passe de l'état `libre` à `occupé` si un agent a_h^n effectue l'action `ENTRER` sur cet emplacement.

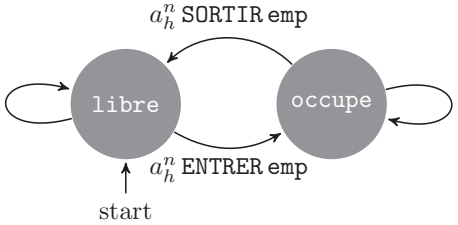


FIGURE 4.5 – Automate à états finis simple représentant le comportement d'un emplacement de stationnement.

Dans cet exemple, le décideur politique a donc la possibilité de paramétrer l'emplacement de stationnement en spécifiant des transitions différentes. L'automate de l'emplacement ne dispose pas d'états finaux.

Fonctionnement interne d'un état de l'automate Dans l'instance de SmartGov, chaque agent a son propre automate à états finis décrit de manière à être paramétrable par le décideur, ou à défaut un intermédiaire connaissant le principe des automates. Par conséquent, le vecteur d'entrées Σ, utilisé pour spécifier les conditions permettant le changement d'état, est décrit par le tuple :

$$\Sigma = \langle \mathcal{S}_\Sigma, I_\Sigma, A_\Sigma \rangle$$

où $\mathcal{S}_\Sigma$ correspond à l'ensemble des structures prises en compte, I_Σ l'ensemble des perceptions pertinentes et A_Σ l'ensemble des actions. Chaque état $s \in S$ de l'automate décrit :

— l'ensemble de structures pertinentes dans cet état $\mathcal{S}_s \subseteq \mathcal{S}_\Sigma$;
— l'ensemble des perceptions pertinentes $I_s \subseteq I_\Sigma$;
— l'ensemble des actions à considérer $A_s \subseteq A_\Sigma$.

Le décideur décrit également le fonctionnement des capteurs de l'agent (distance de vue, capacités perceptives).

Personnalité de l'agent

La personnalité correspond à la manière dont l'agent réagit aux perceptions de son environnement et l'importance qu'il attribue à chacune d'entre elles.

La personnalité de l'agent humain a_h^n, notée P^n, est défini par :

$$P^n = \{(w_i, u_i, c_i) | i \in I^n \subseteq I, w_i \in \mathbb{R}^+\} \tag{4.5}$$

où le tuple (w_i, u_i, c_i) représente l'*intérêt* d'une perception $i \in I^n$ avec :

- $w_i \in \mathbb{R}^+$ le poids attribué à cette perception i ;
- u_i: $\Lambda_i \rightarrow [v_1; v_2]$ la fonction d'utilité lié à la perception i ;
- $c_i \in$ $(\{\text{bonus}, \text{normal}\}, \{\text{normal}, \text{malus}\}, \{\text{bonus}, \text{malus}\}, \{\text{normal}, \text{normal}\})$ est une caractéristique associées à la perception i parmi l'ensemble des combinaisons *bonus/normal* et *malus/normal*.

Le poids w_i, associé à chaque perception $i \in I^k$, décrit l'importance que l'agent humain attribue à cette perception. Plus la valeur du poids est petite, moins la perception i a d'importance pour l'agent.

Chaque agent humain possède un ensemble fini de perceptions I^k associées à certaines structures de l'environnement. La fonction d'utilité $u_i(\lambda_i)$ définie la réponse de l'agent à la valeur de la perception i. Nous appelons *pire intervalle* (respectivement *meilleur intervalle*) l'intervalle dans lequel l'utilité est la plus faible (resp. la plus élevée) et égale à une valeur constante v_1 (resp. v_2) sur cet intervalle. Nous appelons *transition* l'évolution de l'utilité entre ces deux intervalles avec une équation définie dans la fonction d'utilité u.

L'évolution de l'utilité dans la *transition* peut être linéaire (voir figure 4.6). Dans ce cas, l'évolution de l'utilité est linéaire par morceaux. Les valeurs des meilleurs et pires intervalles détermine la direction, croissante ou décroissante, de la fonction. Il est également possible de représenter l'évolution de l'utilité par une fonction non-monotone en spécifiant des ensembles de meilleurs et pires intervalles. Dans ce cas, le décideur doit spécifier l'utilité associer à chaque intervalle (pour un ensemble de valeurs croissantes $a < b < \cdots < f$, il est possible d'avoir une utilité de 1 sur l'intervalle $[a, b]$, 0.2 sur l'intervalle $[c, d]$, 1 sur l'intervalle $[e, f]$ et 0 sur l'intervalle $[g, h]$ par exemple) Le choix de représentation dans ce chapitre présente un unique intervalle dans le pire et meilleur cas.

Il est également possible de faire évoluer l'utilité de manière polynomiale (voir figure 4.7).

Le décideur est libre dans sa représentation de l'utilité. Il peut par exemple considérer un *pire intervalle*, une transition vers le meilleur intervalle avec une fonction polynomiale suivi d'une nouvelle transition vers le pire intervalle avec une fonction linéaire.

Les bornes du *pire intervalle* et du *meilleur intervalle* ainsi que leurs valeurs v_1 et v_2 sont arbitraires et choisies par le décideur en fonction des données à sa disposition. La valeur dans le *pire intervalle* v_1 est toujours supérieure ou égale à 0. La valeur dans le *meilleur intervalle* v_2 est toujours inférieure ou égale à 1 et toujours supérieure ou égale à v_1. On vérifie donc les inéquations suivantes :

$$0 \geqslant v_1 \geqslant v_2 \geqslant 1$$

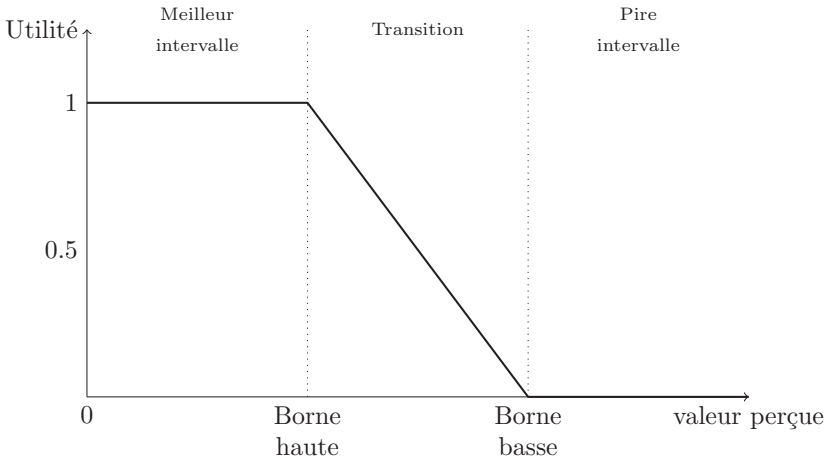


FIGURE 4.6 – Évolution linéaire de l'utilité pour un agent humain d'une valeur perçue. Dans le meilleur intervalle, entre 0 et a, l'utilité est de 1. Dans le pire intervalle, entre b et n'importe quelle valeur supérieure à b, l'utilité est de 0. Entre a et b, l'utilité de la structure décroit linéairement. Cela permet de représenter la réaction de l'agent à ses perceptions.

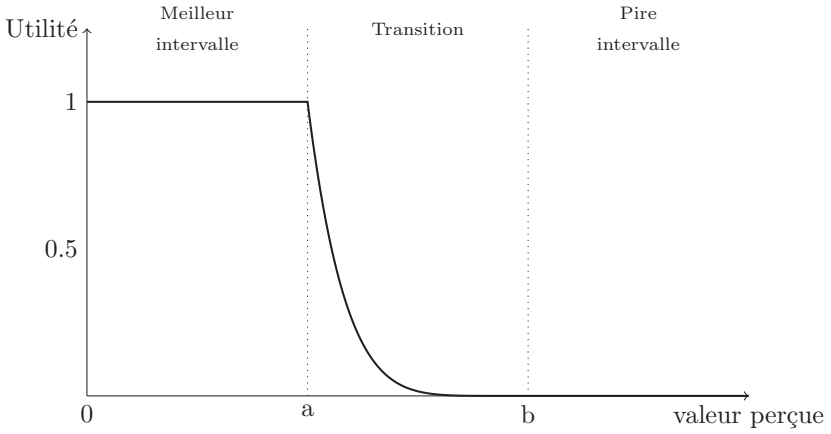


FIGURE 4.7 – Évolution polynomiale de l'utilité d'un agent humain pour une valeur perçue quelconque. Le polynôme est paramétrable pour avoir une courbe représentant les attentes du décideur.

Si v_1 est égale à v_2 alors l'agent considère que l'utilité est identique quelle que soit la valeur perçue λ_i.

La caractéristique c_i définie comment la perception est considéré lors du calcul du score et de la satisfaction que nous aborderons ensuite. Par exemple, si la caractéristique à la valeur $(bonus, normal)$ elle sera considérée comme bénéfique dans le calcul du score et comme normal pour le calcul de la satisfaction.

Le poids ainsi que la fonction d'utilité sont définis par l'utilisateur de SmartGov. Les personnalités de

différents groupes d'agents peuvent être paramétrées séparément pour concevoir une population d'agents humains hétérogènes et observer des comportements variés.

À titre d'illustration, considérons notre agent humain qui souhaite trouver un emplacement de stationnement dans son quartier. Dans notre modèle, il tient compte des trois perceptions suivantes :

— Le temps de recherche en secondes i_1 ;
— La distance entre l'emplacement et la position cible, c'est-à-dire la longueur du trajet à pied en mètres i_2 ;
— Le prix en euros i_3.

La personnalité pour cet agent est décrite par les fonctions d'utilité, poids et caractéristiques correspondants aux perceptions $\{i_1, i_2, i_3\}$. Elle est décrite par le décideur (tableau 4.1).

TABLE 4.1 – Description du tuple d'intérêt de l'agent pour les trois perceptions : temps de recherche, distance entre emplacement et position cible et prix.

Perception	Intervalles				Poids	Fonction d'utilité	Caractéristiques
	Bornes		Valeurs				
	Pire	Meilleur	Pire	Meilleur			
i_1	[600]	[0 :300]	0.2	1	0.75	affine	bonus, malus
i_2	[0 :250]	[450 :500]	0	1	1	affine	normal, normal
i_3	[0 :2]	[3.5 :4]	0	1	0.5	exponentielle	normal, normal

Le décideur choisit les valeurs à partir de données démographiques, de connaissances expertes ou autre. Ici, le choix indique que, pour l'agent, le temps de recherche est une perception de l'environnement qui a plus d'importance que le prix mais moins d'importance que la distance de marche. De plus, l'utilité la plus basse liée au temps de recherche est de 0.2 par exemple.

Le décideur peut modifier tous les attributs lié à la construction de la personnalité d'un agent et de l'intérêt pour chaque perception :

— Les bornes de chaque intervalle ;
— Les valeurs de chaque intervalle ;
— L'évolution de la transition entre ces intervalles ;
— Le poids associé ;
— La fonction d'utilité ;
— La caractéristique.

Ces valeurs pourraient être déterminées à partir d'algorithmes d'apprentissage automatique avec l'étude des trajets de volontaires par exemple en imaginant une application de mobilité et de parkings où l'utilisateur spécifie l'endroit où il stationne et son appréciation sur la disponibilité des emplacements par exemple.

Cette section introduit dans un premier temps la fonction de score σ déterminant les interactions de l'agent avec l'environnement, puis la fonction de satisfaction ς pour évaluer l'impact de l'environnement sur l'agent. Dans un second temps, nous verrons comment l'automate, les perceptions, la fonction de score et les actions sont utilisées pour décrire le comportement de l'agent.

Fonction de score σ Les agents présents dans SmartGov sont considérés comme utilitaristes et effectuent donc l'action qu'ils jugent être la plus pertinente à chaque instant. La fonction de score évalue un ensemble de perceptions liées à une action spécifique de l'environnement.

La fonction de score σ: $I^n \rightarrow [0;1]$ attribue un score à partir des perceptions obtenues et de l'intérêt de chacune d'entre elles, tel que :

$$\sigma(i_1, \ldots, i_n) = min \left(\frac{\sum\limits_{i;c_{i,0}=normal} w_i \times \lambda_{S_{Ii}}}{\sum\limits_{i;c_{i,0}=normal} w_i} + \frac{\sum\limits_{i;c_{i,0}=bonus} w_i \times \lambda_{S_{Ii}}}{\sum\limits_{i=1}^{n} w_i}, 1 \right) \tag{4.6}$$

où n est le nombre de perceptions considérés par l'agent humain. Les perceptions *bonus* signifient que leur utilité ne peuvent qu'améliorer le score. Nous avons une partie des perceptions qui sont statiques et prises en compte à chaque fois par l'agent pour le calcul du score et des perceptions dynamiques qui, lorsqu'elles sont présentes, améliore le score. Par exemple, le temps de recherche peut être considéré comme *bonus* dans la mesure où l'agent voit son score augmenter avec le temps de recherche qui augmente pour représenter l'urgence qu'il a de trouver un emplacement satisfaisant.

Considérons l'exemple suivant où un agent perçoit i et i' définis dans tableau 4.2.

TABLE 4.2 – Deux perceptions et leurs intérêts associés

Perception	Intervalles				Poids	Fonction d'utilité	Caractéristiques
	Bornes		Valeurs				
	Pire	Meilleur	Pire	Meilleur			
i	[0.75 :1]	[0 :0.25]	0	1	1	affine	normal, normal
i'	[0 :0.25]	[1]	0	1	0.75	affine	bonus, malus

Fonction de satisfaction ς À l'utilité, nous ajoutons la notion de *satisfaction* représentant le gain personnel de l'agent humain à effectuer une action particulière.

La fonction de satisfaction ς: $I^n \rightarrow [0;1]$ est construite de manière similaire à la fonction d'utilité et est décrite de la manière suivante :

$$\varsigma(i_1, \ldots, i_n) = max \left(\frac{\sum\limits_{i;c_{i,1}=normal} w_i \times \lambda_{S_{Ii}}}{\sum\limits_{i;c_{i,1}=normal} w_i} - \frac{\sum\limits_{i;c_{i,1}=bonus} w_i \times \lambda_{S_{Ii}}}{\sum\limits_{i=1}^{n} w_i}, 0 \right) \tag{4.7}$$

où n est le nombre de perceptions considérés par l'agent humain. Notons que les perceptions *malus* sont soustraites au lieu d'être additionnées lors du calcul de la satisfaction et impactent négativement la satisfaction. Par exemple, le temps de recherche est considéré comme *malus*. Si nous reprenons les perceptions précédentes,

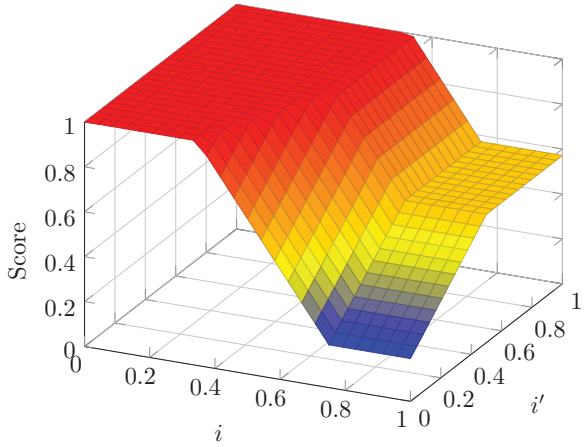


FIGURE 4.8 – Représentation du score à partir de équation (4.6) en fonction de deux perceptions i et i' d'après la personnalité de l'agent (tableau 4.2)

nous pouvons quantifier l'urgence de l'agent humain à trouver un emplacement de stationnement avec le temps qui passe. Cependant, cela signifie également que l'agent rencontre des difficultés pour trouver un emplacement de stationnement pertinent dans le quartier cible, ce qui conduit à une insatisfaction notamment due au temps de recherche conséquent. Une satisfaction élevée indique que l'agent réagit positivement à son environnement.

Ces deux notions, l'utilité et la satisfaction, sont les composantes de la personnalité de l'agent humain (l'agent microscopique n'ayant pas de personnalité). Elles permettent de produire des populations hétérogènes et propose les outils nécessaires pour que le décideur politique puisse facilement ajuster les agents humains et observer des comportements spécifiques dans la simulation.

Reprenons l'exemple introduit dans la partie précédente avec un agent humain et deux intérêts i et i' définis dans tableau 4.2.

La satisfaction évolue différemment de l'utilité figure 4.9.

L'usage de la satisfaction dépend des besoins du décideur et peut être utilisé pour modifier le comportement de l'agent ou encore d'exprimer son retour sur l'environnement.

Impact du score et de la satisfaction sur l'agent La personnalité de l'agent est combinée avec l'automate à états finis pour produire un agent paramétrable par le décideur. Dans ce paragraphe, nous nous intéressons à l'utilisation de la fonction de score σ avec les perceptions de l'agent en fonction de l'état de son automate. Lors de la description de l'automate à états finis, le décideur spécifie les actions $A_s \in A_\Sigma$ disponibles par état de l'agent. Chaque état de l'automate décrit un ensemble de structures à prendre en considération. Certaines de ces actions sont liées aux structures *actionnables* tel que l'ensemble des actions que peut effectuer l'agent dans l'état s est l'ensemble joint $\boldsymbol{a}_s = A^h_{s_0} \times \cdots \times A^h_{s_p}$ où $A^h_{s_i}$ est l'ensemble des actions possibles pour la structure actionnable s_i, tel que $\boldsymbol{a}_s = \langle a^h_{s_1,1}, a^h_{s_1,2}, \ldots, a^h_{s_n,p-1}, a^h_{s_n,p} \rangle$. Pour chacune

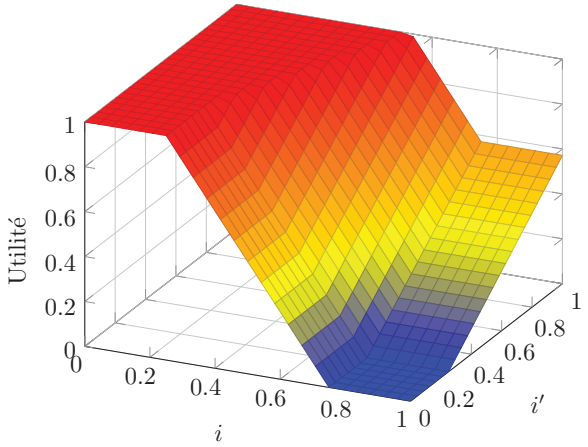


FIGURE 4.9 – Représentation de la satisfaction à partir de équation (4.7) en fonction de deux perceptions i et i' d'après la personnalité de l'agent (tableau 4.2)

de ces actions, l'ensemble des perceptions pertinentes pour évaluer une action est renseigné par le décideur. Ainsi, pour une action quelconque $a^h_{s_i,j}$, l'ensemble des perceptions à prendre en compte est $I^h_{s_i}$. L'agent calcule donc le score $\sigma(I^h_{s_i})$ de l'action $a^h_{s_i,j}$ pour chacune des actions possibles dans cet état. Il hiérarchise ensuite les actions à partir de leur score et effectue l'action avec le plus haut score avec une probabilité égale au score calculé pour cette action. Il a une probabilité d'effectuer la première action, sinon il a une probabilité d'effectuer une seconde action. Dans le cas où il n'effectue aucune des actions pour lesquelles l'agent a calculé un score, il effectue l'action par défaut défini dans son automate.

Considérons un agent dans un état 1 et la possibilité de transiter vers les 2, 3 ou 1 (action par défaut). Lorsqu'un état a plus de deux transitions, celles-ci sont hiérarchisées en fonction de leur importance pour l'agent. Considérons trois actions a_1, a_2, a_3 dans l'état courant de l'agent avec a_1 et a_2 deux actions liées à des structures actionnables et une action par défaut a_3. Il calcule le score de a_1 avec les perceptions I^h_1 tel que $\sigma(I^h_1) = 0.8$ et le score de a_2 avec les perceptions I^h_2, $\sigma(I^h_2) = 0.85$. L'action a_2 à un score plus élevé que a_1 et est considérée en premier. L'agent effectue donc l'action a_2 avec une probabilité de $\mathbb{P}(a_2) = \sigma(I^h_2)$ et une probabilité de faire l'action a_1 $\mathbb{P}(a_1) = (1 - \mathbb{P}(a_2))\sigma(I^h_1)$. Enfin, il effectue l'action par défaut a_3 avec une probabilité de $\mathbb{P}(a_3) = (1 - \sigma(I^h_1))(1 - \sigma(I^h_2)$. Soit pour un nombre quelconque d'actions hiérarchisées et ordonnées $\mathbb{P}(a_n) = \Pi_{i<n}(1 - \sigma(I^h_i))$.

La fonction de satisfaction est utilisée pour déterminer si l'agent est globalement satisfait de son environnement et, par conséquent, de la politique actuellement éprouvée. La satisfaction peut être utilisée pour représenter l'usure de l'agent humain face aux décisions politiques. Si cette satisfaction est faible pendant un certain temps, l'agent humain envisage de nouvelles alternatives (par exemple chercher du covoiturage plutôt que venir payer un emplacement de stationnement trop cher tous les jours). La satisfaction est un outil à disposition du décideur pour structurer l'évolution de l'agent par rapport à son environnement.

Objectifs de l'agent La description du ou des objectifs de l'agent microscopique est associée à l'ensemble des états finaux de l'automate. Dans le cas où l'ensemble des états finaux est vide, l'agent suit son automate pendant toute la durée de la simulation. Dans le cas où l'agent possède plusieurs états finaux, il suit les transitions de son automate jusqu'à atteindre l'un d'entre eux en fonction de ses interactions avec l'environnement.

La personnalité de l'agent sert également à déterminer son comportement vis-à-vis de sa recherche de l'ensemble des états finaux. Dans notre exemple, l'objectif de l'agent est de trouver un emplacement de stationnement pertinent pour lui. Cependant, si sa probabilité de se garer est toujours nulle, dû, par exemple, à des tarifs trop élevés, alors l'agent ne sera pas en mesure d'atteindre son état objectif.

4.4.3 Co-construction de la boucle sensorimotrice de l'agent microscopique

Cette section introduit la représentation générique de la boucle sensorimotrice de l'agent microscopique dans SmartGov. Nous verrons ainsi en détail comment le décideur dote l'agent de capteurs et d'actionneurs et comment ceux-ci sont utilisés avec la description de l'automate à états finis pour évoluer dans l'environnement.

Gestion des perceptions pour les agents microscopiques

Le décideur décrit et paramètre les capteurs de l'agent microscopique de la manière suivante :

— à travers la description des structures à considérer pour chaque état de son l'automate ;

— avec l'ajout de capteurs spécifiques.

De plus, chaque capteur définit un ensemble de caractéristiques liées à la manière de percevoir (distance de vue, cône de vue, perception globale ou locale).

Chaque état $s \in S$ de l'automate dispose d'un ensemble de capteurs, pouvant être les mêmes pour tous les états ou différents. Par défaut, les capteurs de l'agent perçoivent uniquement les informations liées aux structures mentionnées pour chaque état. Le décideur peut choisir de retirer un capteur, dans ce cas l'agent ne perçoit plus une partie ou la totalité d'une structure dans un état donné.

Considérons que l'automate de l'agent i soit constitué de deux états s_1 et s_2 et que l'environnement est décrit par l'ensemble des structures, $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2\}$. L'état s_1 considère les structures de l'ensemble $\mathcal{S}_1$ et l'agent i dispose de capteurs sur avec un ensemble $I_{\mathcal{S}_1}$ de perceptions sur ces structures. À chaque pas de temps où l'agent i est dans l'état s_1, les perceptions pertinentes sont l'ensemble des perceptions liées aux structures de $\mathcal{S}_1$. Si l'ensemble des structures pour un état donné est vide alors l'agent utilise son état interne pour déterminer les transitions de changement d'état.

Ainsi, l'automate détermine le type de structures que l'agent doit percevoir dans l'environnement mais sa position dans le graphe détermine un sous-ensemble de perceptions accessibles sur l'ensemble des structures qu'il doit observer. Dans notre cas, l'agent perçoit la totalité des emplacements sur le tronçon de route où il est présent (dans la mesure où ces tronçons représentent des lignes droites de 250 mètres, l'hypothèse que l'agent perçoit la totalité des emplacements associé au tronçon de route est réaliste).

Fonctionnement des actions pour les agents microscopiques

À partir de l'ensemble des perceptions obtenues, l'agent va pouvoir choisir des actions à effectuer en fonction de sa personnalité, de l'état de son automate. Les actions possibles sont déterminées par les actionneurs que le décideur lui ajoute.

L'ensemble d'actionneurs est :

— propre à l'agent ;

— lié aux structures actionnables mentionnées dans son état.

Le nombre d'actions que l'agent peut effectuer à chaque pas de temps est fini.

Une structure actionnable par un agent microscopique décrit les actions disponibles et leurs conséquences lorsque l'agent les effectue. L'emplacement est un exemple de structure actionnable où lorsque l'agent effectue l'action `ENTRER` (figure 4.5), l'emplacement modifie sa structure interne pour renvoyer la perception *occupé* aux autres agents microscopiques.

Par défaut, un agent microscopique dispose d'un actionneur pour chaque action d'une structure actionnable mentionnée dans son automate à états finis.

Les structures actionnables peuvent donc ajouter des actions supplémentaires par rapport à l'automate mais celles-ci ne sont connues que par le décideur. Le décideur choisit la manière dont les structures actionnables sont utilisées par les agents. En effet, il peut choisir d'interdire que les agents microscopiques utilisent une certaine action sur cette structure par exemple. Ou qu'une personnalité spécifique ne puisse jamais réaliser une action particulière sur la structure.

Considérons notre emplacement *on-street* (figure 4.5). Deux actions peuvent être effectuées sur cet emplacement : se garer, ou quitter l'emplacement (action `SORTIR`). En fonction de l'état interne de l'emplacement, c'est-à-dire s'il est occupé ou libre, une des deux actions est disponible à un moment donné. La possibilité d'effectuer une action est également soumise à des contraintes spécifiques. Par exemple, un emplacement occupé propose uniquement l'action `SORTIR`. Cependant, seul l'agent actuellement stationné sur la place à la possibilité d'effectuer l'action `SORTIR`, les autres agents percevant cet emplacement ne pouvant pas effectuer d'action dessus.

4.4.4 Utilisation de l'exemple de motivation pour construire la couche microscopique

Les trois précédentes sections ont permis de décrire formellement la couche microscopique avec l'environnement et les agents évoluant dans cet environnement. Avec la description de la fonction de score σ et la fonction de satisfaction ς, nous avons décrit en détail le comportement générique d'un agent humain a_h^n. L'agent microscopique, qui ne dispose pas de personnalité, utilise uniquement la description de son automate pour réaliser ses changements d'états. À titre d'illustration, ces descriptions formelles sont instanciées ici pour l'exemple introduit au début de ce chapitre (section 4.2.3), ce qui nous permet de mieux détailler certains aspects de la couche microscopique associée, et d'en donner une image la plus complète possible. L'exemple suivant représente donc des agents humains évoluant dans une zone urbaine avec un automate décrivant une

journée type d'un travailleur se rendant à son lieu de travail (navetteur).

Instance de l'environnement

Dans un premier temps, il est nécessaire de représenter les structures qui composent l'environnement et visible humainement : des emplacements de stationnements et des bâtiments. L'ensemble $\mathcal{S}$ des structures de l'environnement à considérer est $\mathcal{S} = \{\mathcal{S}_{emp}, \mathcal{S}_{bat}\}$ où $\mathcal{S}_{emp}$ est l'ensemble fini des emplacements de stationnements considérés et $\mathcal{S}_{bat}$ est l'ensemble fini des bâtiments. Les différentes structures sont décrites par un ensemble de caractéristiques, par exemple leur type : type des emplacements de stationnements, noté $T_{\mathcal{S}_{emp}} = \{$*off-street, on-street*$\}$ et type de bâtiments $T_{\mathcal{S}_{bat}} = \{residential, leisure, work\text{-}office\}$.

Les capteurs disponibles pour les agents humains sont définis par le décideur politique pour chaque état de l'automate. Les perceptions associées sont utilisées avec la personnalité de l'agent pour calculer le score des structures qu'il perçoit dans l'environnement, ce qui définit la probabilité d'effectuer des actions et guide donc le comportement de l'agent.

Dans cet exemple, le décideur considère les informations suivantes :

— la distance à vol d'oiseau entre l'emplacement de stationnement et son objectif (ici représenté par son lieu de travail), exprimé en mètres ;

— le prix par heure de l'emplacement de stationnement, exprimé en euros ;

— le temps de recherche d'une place, exprimé en secondes.

Il est également possible d'utiliser les vraies distances à pied, cependant nous utilisons l'approximation par la distance à vol d'oiseau pour gagner du temps sur le calcul. Cela fournit l'ensemble des perceptions $I = \{i_{d(s,s')}, i_{p(s)}, i_{(r)}\}$. Pour cet exemple, l'objectif de l'agent humain est de suivre une journée où il part de chez lui, se rend à son lieu de travail, travaille et revient chez lui. Pour se rendre à son lieu de travail, l'agent doit trouver un emplacement de stationnement $s' \in \mathcal{S}_{emp}$ satisfaisant, c'est-à-dire un emplacement dont le score renvoie une probabilité suffisamment élevée pour qu'il puisse se stationner.

Le réseau de l'environnement représente un réseau routier où les nœuds correspondent à des intersections, et les arcs aux tronçons routiers reliant chaque intersection. La fonction de poids ω est définie ici comme la longueur en mètres du tronçon routier. L'agent, lorsqu'il circule sur un tronçon, est en mesure de percevoir les informations qui lui sont associées (emplacements et les lieux de travail).

Le décideur créé une instance d'environnement à partir de l'ensemble des structures, de la description du réseau et des perceptions disponibles. Les agents microscopiques présents dans l'environnement vont interagir avec celui-ci avec leur automate à états finis décrivant leur boucle sensorimotrice.

Génération d'une population humaine

Pour produire des agents microscopiques satisfaisants pour l'instance d'environnement ci-dessus, le décideur propose un automate à états finis (figure 4.10) de manière à décrire le comportement d'usagers navetteurs dans une ville.

Le pré-requis pour concevoir l'agent dans SmartGov est donc de connaitre le principe des automates à états finis et d'être capable d'injecter les compétences métiers nécessaires à l'étude d'une politique particulière, pour

produire des agents représentatifs pertinents. La conception d'un tel automate nécessite soit de disposer de données réelles, soit de s'inspirer de formalismes existant dans la littérature (par exemple : COSMODRIVE [2] pour modéliser l'activté de conduite automobile ou encore les modèles exploitant la fouille de *workflow* comme le *workflow mining* ou le *process mining* pour représenter des comportements). L'expertise humaine est nécessaire pour construire et interpréter les modèles et/ou les données à disposition du décideur. Il adapte ensuite les connaissances extraites avec le modèle de l'automate à états finis. En fonction de ses besoins, l'automate peut évidemment être plus complet. Le décideur peut choisir d'ajouter un état entre le moment où l'agent stationne son véhicule et prend des transports en commun par exemple.

Dans le cadre de notre étude de politique tarifaire, l'automate utilisé dans SmartGov pour représenter la recherche d'emplacements de stationnement d'un actif de la ville un jour de travail est le suivant :

$$\mathcal{A}_{a_h^n} = (\{Repos, NavetteRT, NavetteTR, Recherche, Travail\}, \{h(t), v_{tr}, v_{re}, spot\}, \delta, Repos, \{\})$$

où le vecteur d'entrées contient :

- $h(t)$ l'heure du système ;
- *spot* un emplacement de stationnement ;
- v_{tr} distance au lieu de travail ;
- v_{re} distance à la résidence.

et tel que δ est défini par :

$$\delta(\text{Repos}) = \begin{cases} \text{NavetteRT} & \text{si } h(t) \geq \text{8h00 est vraie,} \\ \text{Repos} & \text{sinon} \end{cases}$$

$$\delta(\text{NavetteRT}) = \begin{cases} \text{Recherche} & \text{si } v_p \leq 250 \text{ est vraie,} \\ \text{NavetteRT} & \text{sinon} \end{cases}$$

$$\delta(\text{NavetteTR}) = \begin{cases} \text{Repos} & \text{si } re = 0 \text{ est vraie,} \\ \text{NavetteTR} & \text{sinon} \end{cases}$$

$$\delta(\text{Recherche}) = \begin{cases} \text{Travail} & \text{si } spot = satisfaisant \text{ est vraie,} \\ \text{Recherche} & \text{sinon} \end{cases}$$

$$\delta(\text{Travail}) = \begin{cases} \text{NavetteTR} & \text{si } h(t) \geq \text{18h00 est vraie,} \\ \text{Travail} & \text{sinon} \end{cases}$$

L'état initial e_0 est *Repos* où l'on considère que l'agent est chez lui au début de la simulation. Il reste dans cet état tant qu'il n'est pas 8h et se rend à son lieu de travail après cette heure (état $NavetteRT$). Lorsqu'il est à moins de 250 mètres de son lieu de travail (le calcul du plus court chemin entre l'entrée de l'agent

2. Lien vers le cosmodrive au LESCOT

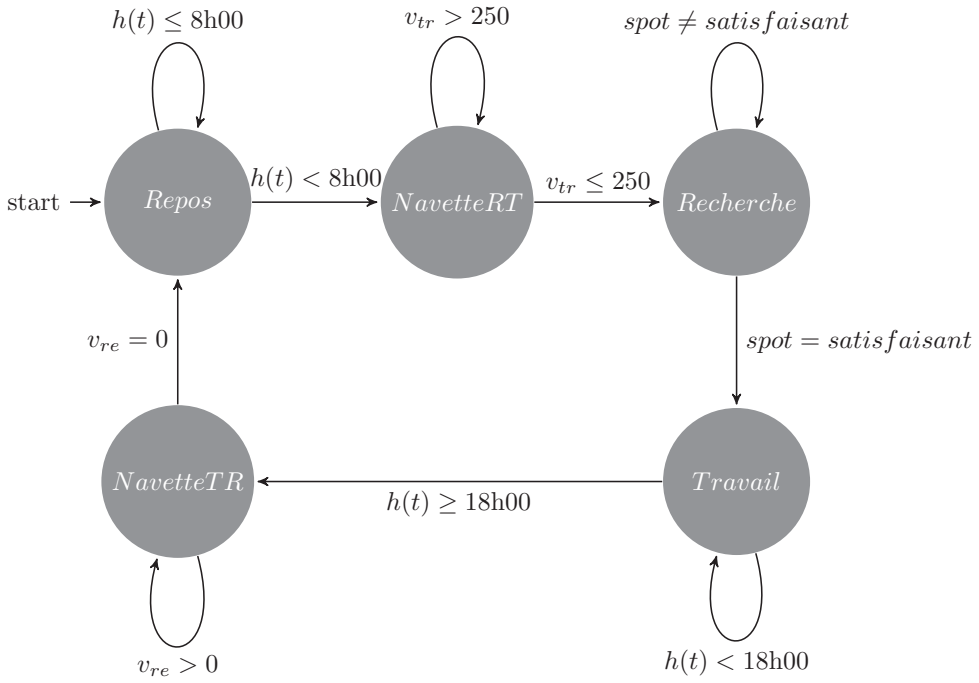


FIGURE 4.10 – Cet automate à états finis représente donc la routine quotidienne d'un navetteur les jours de semaine. L'état initial est *Repos* et il n'existe pas d'état final, le comportement boucle en fonction de l'heure de la simulation.

dans la simulation par son point d'entrée et une intersection la plus proche possible de son lieu de travail), il *Recherche* et évalue les emplacements de stationnement qu'il perçoit. La transition ($spot = satisfaisant$) correspond à un emplacement pour lequel l'utilité a permis à l'agent de considérer cet emplacement comme pertinent. Sinon, tant que l'agent ne trouve pas d'emplacement pertinent, il reste dans l'état *Recherche*. Une particularité ici, est que le comportement interne dans l'état *Recherche* indique que s'il passe plus de 20 minutes à chercher un emplacement $on - street$, il stationne dans un emplacement $off - street$ et à une satisfaction très faible. Une fois un emplacement trouvé, il entre dans l'état $Travail$ jusqu'à 18h où il rentre chez lui avec l'état $NavetteTR$.

Pour déterminer la manière dont un agent humain évalue les emplacements de stationnement perçus, le décideur politique paramètre les personnalités des agents. Il peut créer des profils d'agents humains en définissant des agents ayant différentes fonctions d'utilité. Dans notre exemple, on peut paramétrer les poids de la fonction d'utilité en créant différents ensembles d'agents (tableau 4.3). Par exemple, le profil normal considère qu'un emplacement situé à moins de 250 mètres de son lieu de travail à une utilité de 1 et de 0 s'il est situé à plus de 500 mètres. Après le choix d'une caractéristique $\{bonus, malus\}$, il considère également que plus le temps de recherche augmente, plus le besoin de se garer est fort. Cependant, il est moins satisfait de la disponibilité des emplacements en ville.

Ces profils sont ensuite distribués suivant une distribution gaussienne en fonction des répartitions de profils dans une population (tableau 4.4). Ces profils sont un exemple de ce qui peut être collecté automatiquement avec la Smart City, via des applications dédiées par exemple. Ici, ces données sont construites de manière arbitraire en s'intéressant à des profils grossiers [1] pour exprimer des populations hétérogènes sensiblement différentes. La distribution de ces profils, par exemple la population p_{NR}correspond à un fragment de la zone urbaine où il existe une majorité de profils normaux (60%), une partie de gens impatients (20%) et quelques profils écologiques et économes (10% chacun). Ces données peuvent, par exemple, être collectées par des enquêtes et des sondages [245].

TABLE 4.3 – Les quatres profils d'agents humains et les poids de leur fonction d'utilité

Perception	Profil	Intervalles				Poids	Fonction d'utilité	Caractéristiques
		Bornes		Valeurs				
		Pire	Meilleur	Pire	Meilleur			
$i_{d(s,s')}$	Normal	[500 :1000]	[0 :250]	0	1	1	affine	normal, normal
$i_{d(s,s')}$	Économe	[500 :1000]	[0 :250]	0	1	0.5	affine	normal, normal
$i_{d(s,s')}$	Écologiste	[1000]	[0 :500]	0	1	1	affine	normal, normal
$i_{d(s,s')}$	Impatient	[350 :1000]	[0 :250]	0	1	0.2	affine	normal, normal
$i_{p(s)}$	Normal	[3.5 :4]	[0 :2.5]	0	1	1	affine	normal, normal
$i_{p(s)}$	Économe	[2.5 :4]	[0 :1.5]	0	1	1	affine	normal, normal
$i_{p(s)}$	Écologiste	[4]	[0 :2.5]	0	1	0.5	affine	normal, normal
$i_{p(s)}$	Impatient	[4]	[0 :2.5]	0	1	0.2	affine	normal, normal
$i_{(r)}$	Normal	[600]	[0 :300]	0	1	1	affine	bonus, malus
$i_{(r)}$	Économe	[600]	[0 :300]	0	1	0.5	affine	bonus, malus
$i_{(r)}$	Écologiste	[600]	[0 :300]	0	1	0.5	affine	bonus, malus
$i_{(r)}$	Impatient	[150 :600]	[0 :100]	0	1	1	affine	bonus, malus

TABLE 4.4 – Trois populations créées à l'aide des profils de tableau 4.3 pour avoir des comportements hétérogènes

	Profils (%)			
Pop	Nor	Écol	Écon	Imp
p_{NR}	60	20	20	0
p_{NCE}	50	20	20	10
p_N	100	0	0	0

À partir de l'ensemble de ces caractéristiques, les agents humains ont désormais un comportement et une personnalité. Les perceptions disponibles pour l'agent humain décrivent sa boucle sensorimotrice ainsi que des objectifs à satisfaire dans l'environnement avec une description paramétrable de leur fonctionnement. À partir de cet exemple, la boucle microscopique du modèle est construite et les agents humains peuvent évoluer sur une instance spécifique d'environnement (figure 4.11). Il est maintenant nécessaire d'introduire

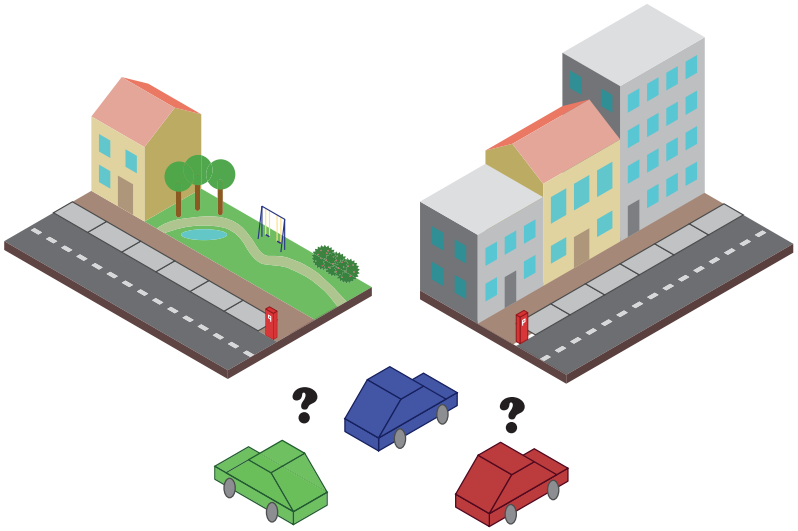


FIGURE 4.11 – Plusieurs agents cherchent un emplacement de stationnement satisfaisant en fonction de leurs perceptions pour se rendre à leur lieu de travail.

une autre dimension permettant de gérer les actions politiques appliquées sur la ville.

Modèle de déplacement

Notre scénario d'étude évalue une problématique de mobilité et nécessite des agents capables de se déplacer. Cependant, il est possible d'envisager des scénarios où la mobilité des agents n'est pas requise.

Par exemple, si le graphe G représente un ensemble de canalisations alimentant des maisons, les agents représentant des maisons sont immobiles et agissent sur l'arrivée d'eau. Il est également possible d'envisager que les agents représentent les fluides et les maisons des structures par exemple. Les choix de modélisation des agents et des déplacements reposent sur les besoins spécifiques du décideur qui n'est pas contraint en utilisant SmartGov.

Pour l'instance choisie, nous représentons le graphe G comme un réseau routier où ω représente la longueur d'un tronçon routier entre deux nœuds. Les agents vont ainsi chercher à trouver le chemin entre deux positions distantes (par exemple la distance entre la résidence et le lieu de travail) qui minimise la distance à parcourir. Il est possible d'envisager que ω corresponde à d'autres critères, comme la durée de parcours du tronçon auquel cas la recherche concernerait le trajet le plus rapide, ou la densité du tronçon et la durée du parcours, où l'agent cherchera un trajet qui minimise le trafic.

Nos agents utilisent l'algorithme de parcours de plus court chemin A* [93], une extension de l'algorithme de Dijkstra [66].

Afin de représenter un déplacement réaliste, la vitesse de déplacement est calculée avec le modèle de Gipps [86], permettant une adaptation dynamique de la vitesse du véhicule en fonction de son voisinage.

4.4.5 Conclusion

Les sections précédentes ont présenté de manière générique la couche microscopique de l'architecture SmartGov. Cette couche décrit formellement l'environnement et les agents microscopiques qui interagissent avec. Une instance de la Couche Usagers est une simulation multi-agents sur laquelle les agents humains vont agir en fonction de leur description : l'automate à états finis définissant leur comportement par rapport au cadre d'étude choisi, et leur personnalité. À partir de ces éléments, le décideur peut déjà observer la simulation et tester une politique spécifique, en appliquant des actions modifiant les structures de l'environnement et impactant les comportements individuels.

Les agents impactent l'environnement via leurs interactions avec les structures. Il est nécessaire d'effectuer plusieurs simulations afin de pouvoir en donner une interprétation moyennée satisfaisante et avoir des résultats pertinents.

Il semble que dans de nombreux cas, le décideur politique aimerait pouvoir observer l'impact d'actions particulières sur le comportement des agents et donc sur l'environnement. Pour observer cet impact, une approche consiste à adopter un processus itératif par essais et erreurs autour d'actions politiques. Par conséquent, nous proposons une couche supplémentaire ayant pour objectif de simuler le processus même d'élaboration de la politique. Nous allons cette fois représenter et gérer la politique c'est-à-dire le choix des actions, leur calibrage, leur application ensuite sur la couche microscopique, et observer l'impact sur l'environnement et les agents humains. Le décideur a ainsi la possibilité de répéter ce processus pour progressivement affiner les actions effectuées.

4.5 Couche macroscopique

La couche macroscopique correspond à la couche décisionnelle de SmartGov, responsable de l'élaboration et de la validation des politiques urbaines. La couche microscopique repose sur la description du réel avec l'environnement et les agents, alors que la couche macroscopique s'occupe de l'élaboration d'une politique et des actions à effectuer.

Cette partie introduit le *Gestionnaire de Politiques* (GDP) : il s'agit du deuxième simulateur multi-agents de SmartGov qui interagit avec le simulateur de la Couche Usagers. Il comprend un environnement $\mathcal{H}$ ainsi qu'une population d'agents appelés *agents politiques* N_p, qui interagissent avec l'environnement de la couche microscopique. L'objectif de la couche macroscopique est de permettre à une population d'agents politiques de décider des meilleures actions à mener sur les structures locales de l'environnement simulé. Ainsi, il est possible de faire émerger, au niveau global, des comportements des agents microscopiques s'adaptant et permettant d'affiner ces actions proposées afin de satisfaire l'objectif recherché par le décideur. Selon la durée de la simulation, différentes actions politiques vont s'enchaîner pour faire évoluer la politique urbaine initiale. Nous appellerons désormais *action politique* toute action menée par l'agent politique dans le cadre d'une politique donnée.

L'environnement de la couche du gestionnaire de politiques est représenté par l'ensemble des structures de la couche microscopique. La figure 4.12 montre comment la couche macroscopique exploite les structures

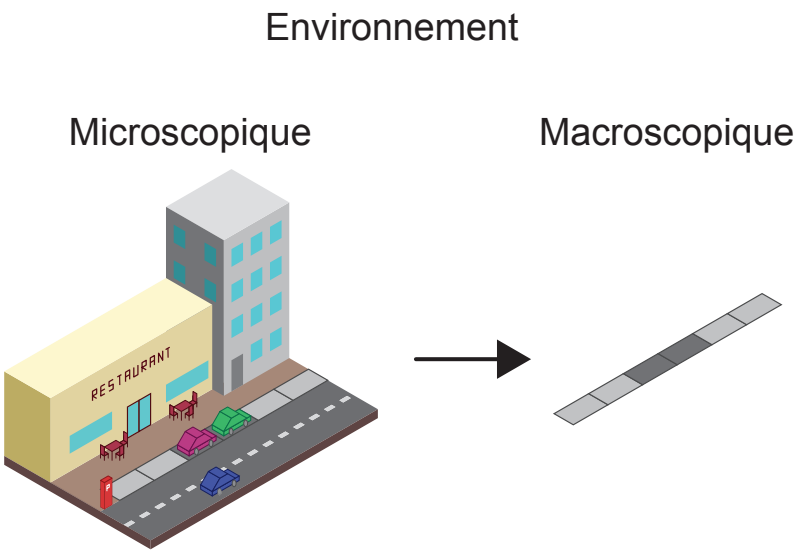


FIGURE 4.12 – L'environnement de la couche microscopique représente une instance simulée de la zone urbaine. Le gestionnaire de politiques construit son environnement à partir des éléments de la couche microscopique.

pour créer son propre environnement. Ici, la structure concernée est un front de rue, composé d'un ensemble d'emplacements. Le gestionnaire de politiques ne dispose d'aucune information à part l'état des emplacements (libres ou occupés).

On parle donc ici du couplage de deux simulateurs : l'un simulant les usagers évoluant dans l'environnement, l'autre simulant des actions sur l'environnement qui vont impacter les comportements des usagers. Dans la section suivante 4.5.1, nous présentons le modèle formel du gestionnaire de politiques. La section 4.6 présente ensuite plus en détails le formalisme de politique élaboré, puis nous expliquons comment se fait le couplage entre les 2 simulateurs (section 4.7).

4.5.1 Gestionnaire de politiques

L'environnement de la couche macroscopique, appelé *gestionnaire de politiques* $\mathcal{H}$, est défini par le tuple suivant :

$$\mathcal{H} = \langle I_{\mathcal{H}}, R_{\mathcal{H}}, A, N_p \rangle \tag{4.8}$$

où :

- $I_{\mathcal{H}} \subseteq I$ est un ensemble fini de perceptions de l'environnement de la Couche Usagers ;
- $R_{\mathcal{H}}$ un ensemble fini de représentations d'environnement construit à partir de l'agrégation de perceptions de $I_{\mathcal{H}}$;
- $A_{\mathcal{H}}$ un ensemble fini d'actions politiques disponibles pour le gestionnaire de politiques ;
- N_p un ensemble fini d'agents politiques.

Toutes les perceptions I sont initialement décrites lorsque l'environnement est construit. Ainsi, l'ensemble fini des perceptions $I_{\mathcal{H}}$ est un sous-ensemble des perceptions déjà présentes dans l'environnement et est utilisé pour créer les états du gestionnaire de politiques.

Extension des structures

Nous avons introduit la structure comme toute entité présente dans la couche microscopique. Les structures sont utilisées dans la couche macroscopique pour décrire l'ensemble fini de représentations d'environnements $R_{\mathcal{H}}$. En effet, nous augmentons la définition d'une structure en ajoutant deux ensembles :

— l'ensemble des perceptions $I_i^p \subseteq I$ que les agents politiques peuvent percevoir sur la structure $s_i \in \mathcal{S}$;

— l'ensemble des actions politiques $A^p \subseteq A_{\mathcal{H}}$ disponibles pour les agents politiques.

Ces ensembles peuvent être vides. Une structure possédant un ensemble d'actions politiques est dite *actionnable* par l'agent politique.

La structure est désormais décrite par le tuple suivant :

$$s_i = \langle t, I_i^h, A_i^h, I_i^p, A_i^p \rangle \tag{4.9}$$

L'intérêt de cette représentation repose sur une unique description des structures composant la couche microscopique et la couche macroscopique vient s'interfacer avec. Les structures font donc le lien entre une description des perceptions et des actions pour les agents microscopiques et permet à la couche du gestionnaire de politiques de décrire l'ensemble des représentations de son propre environnement.

4.5.2 Introduction au modèle de l'agent politique

L'ensemble des agents politiques est $N_p = \{a_p^1, a_p^2, \ldots, a_p^M\}$, $M \in \mathbb{N}$.

Un agent politique a_p^m est défini par le tuple suivant :

$$a_p^m = \langle \mathcal{S}_p^m, I_p^m, A_p^m, \rho_p^m \rangle \tag{4.10}$$

où :

— $\mathcal{S}_p^m \subseteq \mathcal{S}$ décrit un ensemble fini de structures pour lesquelles l'agent est en mesure de percevoir $I_p^m \subseteq I_{\mathcal{H}}$ perceptions ;

— $A_p^m \subseteq A$ décrit l'ensemble fini des actions disponibles sur les structures $\mathcal{S}_p^m$;

— $\rho_p^m : I_p^m \times A_p^m \to A_p^m$ est une fonction de décision pour choisir l'action correspondant au vecteur de perceptions fourni par l'environnement.

Les agents politiques sont distribués sur l'environnement : ils sont responsables d'une zone urbaine restreinte et appliquent des actions locales sur les structures pour augmenter les performances locales et globales. Une problématique dans le cadre de la couche macroscopique est la répartition pertinente des agents politiques sur l'environnement. Un autre point est la modélisation pertinente de l'agent politique pour qu'il

soit capable de s'adapter à n'importe quelle couche microscopique. C'est une autre contribution de la thèse, détaillée dans le chapitre 6.

4.5.3 Fonctionnement

Par analogie avec les agents humains, les agents politiques ont également une boucle sensorimotrice.

L'ensemble des perceptions $I_{\mathcal{H}}$ représente la totalité des capteurs d'un gestionnaire de politique et $A_{\mathcal{H}}$ représente la totalité de ses actionneurs. Ses capteurs et actionneurs sont fournis à la population d'agents politiques. Les actionneurs du gestionnaire de politiques ne peuvent modifier que la partie commune entre l'environnement $\mathcal{E}$ et le gestionnaire de politiques $\mathcal{H}$, soit l'ensemble $\mathcal{S}_{\mathcal{H}} \subseteq \mathcal{S}$. Par conséquent, la couche du gestionnaire de politiques ne peut pas directement changer les paramètres des agents humains mais peut, à travers différentes actions politiques, encourager des changements de comportement. Une politique qui va dans ce sens est appelée *politique incitative*. Une *politique incitative* doit être liée à un objectif, comme la diminution de la pollution de l'air. Cependant, les changements de comportements doivent avoir été envisagés au préalable dans l'automate à états finis, pour pouvoir se manifester dans la simulation. Par exemple, un agent qui ne souhaite plus utiliser la voiture doit disposer d'autres moyens de transports plus propres pour satisfaire sa personnalité.

Parmi les structures actionnables par les agents politiques, nous trouvons une catégorie particulière de structure pour laquelle certaines perceptions sont à la fois perçus par l'agent humain et modifiable par l'agent politique. L'action politique $\alpha : \Lambda \to \Lambda$ avec $\alpha(s_{\lambda_i}) = s_{\lambda'_i}$ représente la modification d'une valeur spécifique d'une perception i par l'action α sur la structure s.

4.5.4 Conclusion

La couche du gestionnaire de politiques présente un formalisme pour créer une instance de gestionnaire de politiques dont la tâche est de distribuer une population d'agents politiques pour éprouver des actions politiques et construire des politiques urbaines pertinentes. La construction d'une instance de gestionnaire de politiques repose sur le choix d'une fonction de décision pour une bonne application des actions politiques. La description des deux couches étant générique, il est nécessaire de disposer de méthodes auto-adaptatives pour construire les politiques urbaines. Le fonctionnement de cette couche est créée automatique à partir de la description, par le décideur, de l'environnement microscopique. Ainsi, le processus itératif d'essai de politiques est réalisé par les agents politiques et permet au décideur d'intervenir sur la co-construction à plusieurs niveaux que nous allons aborder dans la section suivante. La fonction de décision ρ_p^m de l'agent microscopique n'est pas décrit dans ce chapitre mais sera abordé dans la section 6.3 et un exemple concret est disponible avec le chapitre Expérimentations (chapitre 7).

4.6 Modèle de politique

Le principal atout de notre système est de permettre aux décideurs de tester une politique urbaine sans avoir à la mettre en œuvre dans la réalité, avec les moyens coûteux que cela peut engendrer. Notre objectif est ainsi de visualiser directement les conséquences d'une politique et d'en étudier la validité sur la modélisation réaliste de l'environnement qui a été faite.

Dans cette section, nous proposons de décrire les outils mis à disposition du décideur pour interagir avec l'outil de simulation SmartGov. Le décideur définit d'abord les paramètres de son environnement (section 4.6.1), il a ensuite le choix entre différents types de simulation, selon l'objectif recherché (section 4.6.2). Nous illustrons enfin le modèle proposé sur l'exemple de motivation de tarification des places de stationnement.

4.6.1 Paramètres pour la modification de politiques

Pour permettre au décideur politique d'interagir avec SmartGov, nous proposons de décrire quatre paramètres :

1. Le *périmètre environnemental* ψ ;
2. Le *domaine d'action* φ ;
3. L'*indicateur de performance* ϕ ;
4. L'*horizon de temps* H.

Le périmètre environnemental Il est défini par le tuple :

$$\psi = \langle G, I, \mathcal{S}, N \rangle \tag{4.11}$$

où :

— G est un graphe ;
— I les perceptions disponibles ;
— $\mathcal{S}$ un ensemble de structures ;
— N est un ensemble d'agents microscopiques concernés par les actions politiques.

et décrit le périmètre sur lequel la politique va être appliquée. Dans un souci de réalisme, ou si on a le besoin d'intégrer une zone urbaine plus grande, le décideur politique a la possibilité de décrire une plus grande couche microscopique que celle utilisée pour construire une politique. Il utilise dans ce cas ψ pour décrire uniquement l'espace concerné par des actions politiques. Cette zone peut être continue dans l'espace ou fragmentée. L'ensemble d'agents microscopiques $N \subseteq N_h$ correspond à des agents sensibles aux actions politiques (par exemple des structures modifiées par des actions politiques). Le décideur peut ainsi envisager une population directement concernée par l'action politique ou plus sensible aux modifications de l'environnement.

Le domaine d'action Le décideur définit un ensemble de *domaines d'action* où un domaine d'action φ, définit pour chaque action α de chaque structure la probabilité qu'elle soit utilisée dans l'environnement. Le

domaine d'action s'exprime avec $\varphi : A \times S \rightarrow [0; 1]$, où le domaine d'action appliqué à la structure $s \in S$ est

$$\varphi(s) = \{(\alpha_i, p(\alpha_i))|\alpha_i \in A, p(\alpha_i) \in [0; 1]\}$$

où α_i décrit une action et $p(\alpha_i)$ la probabilité de faire cette action. En utilisant ces probabilités, le décideur peut choisir d'autoriser ou d'interdire certaines actions sur l'environnement ou d'en limiter le nombre d'utilisations pour favoriser d'autres actions politiques. Ces probabilités ont un impact sur les actions des agents locaux et sur leur automate à états finis. En effet, lorsqu'un agent politique perçoit une structure, il peut par défaut effectuer la totalité des actions A^p disponibles pour une structure.

L'indicateur de performance L'*indicateur de performance* ϕ est une fonction $\phi : R_{\mathcal{H}} \rightarrow \mathbb{R}$ pour évaluer la pertinence d'une représentation $r_{\mathcal{H}} \in R_{\mathcal{H}}$ du gestionnaire de politiques. Le décideur politique mentionne plusieurs objectifs et leurs poids associés pour décrire une fonction mono-objectif. La modification des poids à un impact sur la solution proposée par SmartGov. Par exemple, le décideur peut considérer comme pour améliorer la circulation en centre-ville, son indicateur de performance est le débit de voitures par tronçon. Il peut également considérer à la fois le débit de voitures et la vitesse moyenne en spécifiant l'importance de chaque critère dans sa fonction objectif. Le choix d'une fonction objectif mono-critère simplifie la recherche de solutions pour la production de politiques à destination du décideur.

L'horizon de temps Le décideur définit un *horizon de temps* H permettant de spécifier la période sur laquelle les politiques urbaines vont être testées et éprouvées. Plus l'horizon de temps est grand, plus la politique subira les divergences avec la réalité, à cause de la difficulté de prédire et anticiper le futur. Un horizon de temps court peut être utilisé pour prévoir une mise à jour régulière des données et ainsi produire des politiques urbaines en continu, corrigées par l'apport de données.

4.6.2 Définitions des interactions entre le décideur et SmartGov

Pour faciliter la compréhension du processus d'interactions, nous proposons les définitions suivantes relatives aux politiques urbaines liées à SmartGov :

Définition 1 *Une* politique urbaine *représente l'application d'un ensemble ordonné d'actions sur l'environnement pendant un certain temps afin de satisfaire les objectifs visés.*

Comme mentionné précédemment, les décideurs peuvent co-construire des politiques urbaines avec l'outil de simulation, qu'ils utilisent avec des attentes différentes, en fournissant une entrée E pouvant prendre l'une des formes suivantes :

1. une simple analyse de l'environnement ζ ;
2. des objectifs spécifiques $\mathcal{O}$;
3. une politique urbaine $\mathcal{P}$.

Nous définissons maintenant chaque type d'entrée que le décideur peut utiliser dans SmartGov.

Définition 2 *Une* analyse d'environnement $\zeta = \langle \psi, \phi, H \rangle$ *est définie par un périmètre environnemental* ψ*, des indicateurs de performances* ϕ *à évaluer et un horizon de temps* H*.*

Une analyse d'environnement ζ crée un état initial à partir des structures spécifiées. Cependant, ζ ne permet pas de tester des politiques mais uniquement d'observer que le comportement des agents et l'environnement soient cohérents. L'objectif d'une analyse d'environnement est de permettre la visualisation et l'évaluation de l'environnement et des agents humains à partir des données fournies, par exemple dans un but de validation du réalisme de la simulation. Il est également possible d'évaluer une politique en spécifiant l'environnement résultant d'une application politique.

Définition 3 *Un* objectif politique $\mathcal{O} = \langle \phi, \varphi, \psi, H \rangle$*, fourni par le décideur, est défini par des indicateurs de performances* ϕ *à optimiser ;* φ *le domaine d'action choisi par le décideur ;* ψ *le périmètre environnemental sur lequel la politique va être appliquée et* H *représente la date au plus tard à laquelle* $\mathcal{O}$ *devrait être satisfait.*

Fournir au simulateur un objectif politique $\mathcal{O}$ crée un état initial à partir des actions, des structures et des perceptions disponibles, avec des performances que les agents politiques doivent satisfaire. Cette fois, la simulation applique des actions politiques parmi celles possibles, et évalue la performance de ces actions.

Définition 4 *Une* politique urbaine $\mathcal{P} = \langle \mathcal{F}, \mathcal{O} \rangle$ *est définie par* $\mathcal{F}$ *un ensemble fini de fonctions d'actions politiques, tel que :*

$$f^m : R^m \rightarrow A^m | m \in N_p, f^m \in \mathcal{F}, R^m \subseteq R, A^m \subseteq A$$

et $f_i(r_i) = \alpha_i$ *représente l'action politique courante à appliquer quand la représentation locale de l'agent politique* a_p^m *est* r^m *;* $\mathcal{O}$ *correspond aux objectifs politiques spécifiés par le décideur.*

Enfin le décideur peut proposer une politique urbaine $\mathcal{P}$ existante avec l'objectif d'en évaluer les performances en utilisant SmartGov.

4.6.3 Utilisation de l'exemple de motivation pour construire la couche macroscopique

Dans la section sur la mise en place d'une instance de la couche microscopique (section 4.4.4), nous avons déroulé la démarche d'un décideur politique pour construire un environnement qui sera simulé dans SmartGov. Nous nous intéressons maintenant à la construction d'une politique urbaine pour satisfaire les objectifs du décideur.

Le décideur exprime une *fonction objectif* à l'aide de ses objectifs politiques $\mathcal{O}$ correspondant ici à la maximisation du gain dans la ville, où le gain correspondant aux sommes perçues au titre du stationnement. Pour construire sa politique urbaine, le décideur spécifie dans un premier temps le *périmètre environnemental* ψ en décrivant un ensemble de quartiers sur lesquels il souhaite appliquer sa politique. L'*indicateur de performance*, correspondant à l'objectif $\mathcal{O}$, est donc le produit du prix d'un emplacement par son occupation (0 si libre et 1 si occupé) pour tous les emplacements de stationnement du périmètre considéré. L'ensemble $R_{\mathcal{H}}$ des représentations de l'environnement correspond donc à la combinaison des emplacements de stationnement avec leur statut et le prix de cet emplacement.

Notons p un emplacement de stationnement, o son occupation (o pour libre et $\bar{o}$ pour occupé) et t son tarif. Alors $p(\bar{o}, 2)$ est un emplacement occupé dont le tarif est de 2€. Nous adoptons ici un modèle où les usagers paient un forfait lorsqu'ils stationnent dans un emplacement. Ainsi, si l'on considère que chaque emplacement peut avoir un prix différent, alors pour N emplacements différents et M prix différents, l'ensemble des représentations possible est $|R_{\mathcal{H}}| = (2 * M)^N$. L'ensemble des actions politiques va être décrit par l'augmentation des prix ($\nearrow$) ou la diminution des prix ($\searrow$). Nous introduisons également une action permettant au gestionnaire de politique de ne pas agir ($=$). Nous avons donc l'ensemble des actions disponibles $A_{\mathcal{H}} = \{\nearrow, \searrow, =\}$. L'ensemble des perceptions disponibles à l'agent politique va être lié à l'indicateur de performance, par conséquent $I_{\mathcal{H}} = \{o, t\}$.

L'agent politique va être modélisé à partir des structures qui lui seront assignées. L'agent politique pourra percevoir l'ensemble de ses emplacements de stationnement soit $\mathcal{S}_m = \{emp\}$. Les perceptions de l'agent ainsi que ses actions sont égales aux perceptions disponibles pour le gestionnaire de politiques dans notre cas. Soit $I_m = I_{\mathcal{H}}$ et $A_m = A_{\mathcal{H}}$ les perceptions et actions politiques que possèdent l'agent politique. La fonction de décision de l'agent politique permet à l'agent d'améliorer l'état dans lequel il se trouve actuellement pour améliorer le gain.

À partir de l'exemple, nous avons décrit la création d'une instance de gestionnaire de politiques et sa méthode de résolution d'une politique urbaine. Cette description est plus simple puisqu'elle s'adapte rapidement à n'importe quelle représentation de la couche microscopique.

La section sur la couche microscopique, celle sur la couche macroscopique ainsi que la description du modèle de politique et des entrées du décideur politique forment la description de l'architecture générique de SmartGov. À partir de l'ensemble de ces formalismes, il est possible de créer une instance de simulation SmartGov afin de résoudre un problème de politique urbaine. La description des deux couches forme donc deux simulateurs multi-agents au fonctionnement général indépendant. La section suivante introduit la combinaison entre ces deux simulateurs et comment leur couplage, et les interactions qui en résultent, permettent de produire des politiques pertinentes.

4.7 Couplage dynamique entre les deux couches

La première étape consiste à construire les deux simulateurs à partir de l'entrée E du décideur. L'analyse d'environnement ζ, les objectifs $\mathcal{O}$ et la politique urbaine $\mathcal{P}$ permettent de créer une instance d'environnement $\mathcal{E}$ alors que $\mathcal{O}$ et $\mathcal{P}$ servent à créer une instance du gestionnaire de politiques $\mathcal{H}$. L'environnement de la couche microscopique $\mathcal{E}$ définit les structures et construit des agents microscopiques réalistes. Le gestionnaire de politiques $\mathcal{H}$ construit des agents politiques qu'il distribue sur les structures de l'environnement.

La seconde étape est le fonctionnement du couplage entre les deux simulateurs (figure 4.13) : la population d'agents microscopiques N_m interagit avec les structures $\mathcal{S}$ en suivant leur automate et provoque des changements microscopiques. La population d'agents politiques N_p observe une représentation de l'environnement $r_t \in R_{\mathcal{H}}$ à un instant t correspondant à une agrégation de perceptions $I_{\mathcal{H}}$ des structures modifiées par les agents microscopiques (par l'exemple l'ensemble des emplacements occupés). Un agent politique a_p^m effectue

une action α, modifiant les structures $\mathcal{S}^m$ de l'environnement (augmentation du tarif des emplacements par exemple). En retour, la population N_m observe les différences et modifie son comportement vis-à-vis de cette modification. Par la suite, des modifications du niveau microscopique seront observées (occupation plus faible des emplacements en réponse à l'augmentation des tarifs).

La troisième étape est l'évaluation des actions effectuées par les agents politiques. À n'importe quel moment, $\mathcal{H}$ peut évaluer les performances des agents politiques. Ce calcul des performances se fait à l'aide de l'indicateur de performance ϕ défini par le décideur lorsqu'il fournit $\mathcal{O}$ ou $\mathcal{P}$. En fonction de l'évolution des performances, $\mathcal{H}$ applique des actions pertinentes afin d'améliorer les résultats. La comparaison entre les performances et les objectifs permet de déterminer la pertinence de la politique actuelle.

Quand les agents politiques appliquent une action, une nouvelle simulation démarre avec les valeurs des structures mises à jour. Deux autres approches peuvent être identifiées pour représenter la réaction des agents microscopiques : (i) une réaction sans prendre en compte le temps, c'est-à-dire sans avoir à modéliser la phase d'adaptation. Dans ce cas, quand une nouvelle politique est implémentée, nous supposons que les agents microscopiques n'ont aucune mémoire des actions précédentes et qu'ils agissent comme si la situation était celle à laquelle ils sont habitués. (ii) une réaction en prenant en compte le temps d'adaptation. Dans ce cas, quand l'action est appliquée, les agents mettent du temps pour s'adapter et changer leur comportement. Dans notre contexte, pour évaluer l'efficacité de la politique, nous sommes intéressés par l'état stable obtenu après la période d'adaptation. Nous avons donc choisi d'utiliser l'approche (i).

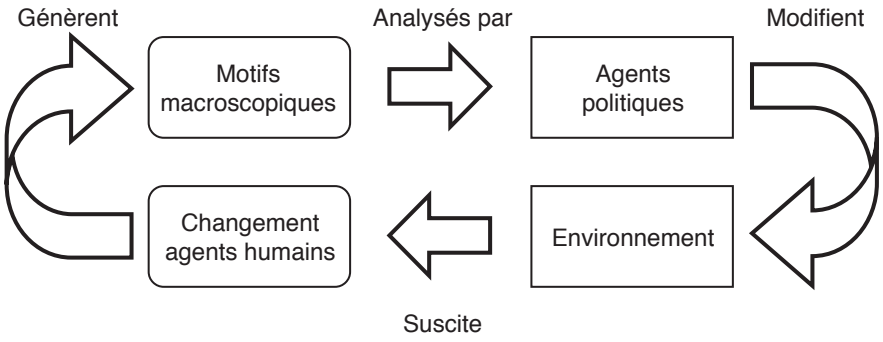


FIGURE 4.13 – Fonctionnement du couplage entre les deux simulateurs multi-agents. Les agents microscopiques agissent sur l'environnement modélisé et produisent des résultats macroscopiques qui sont analysés par le GDP à δt. Cette dernière décide ensuite d'une ou plusieurs actions à effectuer sur l'environnement de la Couche Usagers.

4.7.1 Gestionnaire de simulation et fonctionnement du simulateur

Le couplage dans SmartGov repose sur un gestionnaire de simulations pour effectuer la synchronisation entre les deux couches du simulateur. Le rôle du gestionnaire est d'une part de réinitialiser les simulations de l'environnement aux moments pertinents et d'autre part d'appliquer les actions politiques. Il a également

pour rôle la sauvegarde des états de l'environnement pour afficher l'évolution des simulations. Si le décideur souhaite par exemple observer les conséquences d'une action politique sur une période spécifique (une journée ou une heure donnée), il peut spécifier au gestionnaire de simulation de simuler cette période uniquement sur plusieurs simulations.

Gestion des itérations de la simulation

Les actions politiques effectuées par la couche du gestionnaire de politiques ne vont pas se faire aussi fréquemment que les agents microscopiques vont interagir avec leur environnement.

L'instance de l'environnement de la couche microscopique détermine l'intervalle de temps entre deux itérations du simulateur. En fonction de la granularité attendue, l'intervalle de temps peut être d'une seconde, une minute, etc. Dans le cas d'une simulation où l'intervalle est fixé à une seconde, les agents interagissent régulièrement avec leur environnement. Cependant, les actions politiques ne vont pas s'appliquer toutes les secondes mais à un intervalle de temps plus élevé. Ainsi, chaque couche gère de manière autonome sa temporalité. Le rôle du gestionnaire de simulation est par conséquent de faire la passerelle entre les deux couches et d'appliquer de manière pertinente les actions politiques.

Il est donc possible de spécifier à quel intervalle de temps les actions politiques vont être effectuées sur l'environnement. Il est également possible d'utiliser SmartGov pour étudier l'impact de politiques urbaines à certains moments spécifiques de la journée et de définir les plages horaires concernées.

Gestion de la ré-initialisation de la simulation

La présentation des résultats repose sur une analyse statistique des observations faites lors de la simulation. Ainsi, pour éviter la ré-initialisation manuelle du simulateur, le gestionnaire a pour rôle de re-créer des instances d'environnement de la couche microscopique en modifiant ses informations internes. La position et les automates à états finis des agents microscopiques sont réinitialisées. Chaque structure modifiée par une action politique voit ses valeurs modifiées pour que les agents microscopiques perçoivent des informations mises à jour. L'horloge interne de la simulation est initialisée à une date spécifique. Elle peut être exactement la même entre deux simulations, nous considérons dans ce cas que les agents microscopiques évoluent dans une même simulation avec des structures modifiées. Elle peut également être à une date ultérieure Δt correspondant à un laps de temps avant d'atteindre un nouvel état stable chez les agents microscopiques.

4.8 Co-construction avec SmartGov

La co-construction signifie que deux entités interagissent pour trouver une solution satisfaisante ensemble. Dans notre cas, la notion de co-construction fait référence à l'interaction entre le décideur et l'outil d'aide à la décision SmartGov aux différentes étapes de la conception de la politique urbaine.

Nous envisageons plusieurs niveaux d'interactions lors d'une co-construction entre un utilisateur et SmartGov :

1. l'utilisateur fournit des données pour créer une instance de l'outil ;
2. l'utilisateur modifie des éléments au cours de la simulation ;
3. l'utilisateur propose des retours lorsque l'outil a fini la simulation et fournit une politique urbaine à l'utilisateur.

De plus, la co-construction est également possible entre le décideur et l'usager en utilisant SmartGov comme support pour décrire les actions politiques et leurs conséquences.

Avec les entrées E introduites précédemment, le décideur dispose des briques nécessaires pour créer une instance de l'outil (couche microscopique, gestionnaire de politiques et gestionnaire de simulations).

Une première co-construction de politiques urbaine est réalisée par SmartGov en couplant la Couche Usagers, élaboré par le décideur, avec la couche du gestionnaire de politiques à partir de ses objectifs $\mathcal{O}$. La politique ainsi produite peut ensuite être manipulée par le décideur, ce qui donne lieu à une seconde co-construction où le décideur propose des retours en modifiant certaines valeurs de la politique, en changeant le périmètre considéré. La politique résultante sert de support de communication pour présenter les résultats obtenus aux usagers ou de conseils pour une application réelle.

Sans la couche du gestionnaire de politiques, le décideur devrait spécifier les actions politiques à effectuer sur l'environnement et la co-construction se résumerait à proposer un outil de simulation d'environnement et de l'utiliser tel quel. L'avantage du couplage dynamique est d'avoir une couche supplémentaire pour représenter cette phase d'essai des actions politiques. Ainsi, le décideur interagit avec l'outil à un niveau supérieur où il est en mesure de faire ses retours sur la pertinence des actions proposées et de la solution apportée.

Le fonctionnement de la co-construction peut être simplifiée comme dans la figure 4.14 où le décideur construit une instance de l'outil et fournit son entrée. SmartGov simule et produit un ensemble de politiques pour le décideur. Celui en sélectionne une ou plusieurs et modifie ses paramètres avant de les refournir à SmartGov. Cette étape d'interaction correspond à la troisième version envisagée de la co-construction de politiques urbaines avec SmartGov.

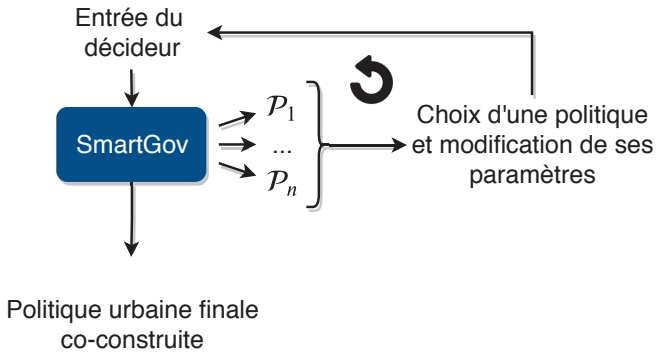


FIGURE 4.14 – Représentation de la co-construction d'une politique urbaine en produisant des politiques qui sont affinées par le décideur politique.

4.8.1 Proposition de politiques urbaines

SmartGov produit une politique urbaine à partir d'une instance complète de l'outil. En fonction de la construction de la couche du gestionnaire de politiques, la production de politiques varie. Par exemple, il est possible d'explorer l'ensemble des représentations du système si cela est possible et de proposer la politique urbaine optimale par rapport aux objectifs du décideur. Dans une perspective multicritère, le décideur politique souhaite en général plusieurs solutions différentes représentant un compromis entre ses critères. Pour simplifier le traitement, nous considérons une fonction mono-objectif agrégée où plusieurs solutions peuvent être proposées en fonction des poids associés aux critères.

4.9 Alimentation du modèle avec des données

L'objectif de SmartGov étant de produire un outil d'aide à la décision de conception politiques, il est nécessaire que l'instance de l'environnement de la couche microscopique soit réaliste pour que les politiques proposées soient pertinentes et correctement évaluées. Cet aspect réaliste repose sur :

— L'utilisation de données réelles pour instancier l'environnement et ses structures ;
— La bonne modélisation et calibration de la population d'agents microscopiques.

Cette section présente les démarches à destination du décideur politique pour réaliser la tâche d'alimentation des données pour SmartGov. Avec l'émergence des *Smart Cities* et des objets connectés, un nombre croissant de sources de données peuvent être envisagées par le décideur politique. Il peut par exemple disposer des données ouvertes rendues accessibles par les villes et les zones urbaines connectées.

4.9.1 Extracteur de données Open Street Map

Un réseau routier est nécessaire lorsque la politique urbaine s'intéresse à des problématiques de mobilité des agents microscopiques. Ainsi, lorsque le graphe G de l'environnement décrit un réseau routier, le décideur a la possibilité d'utiliser un système d'information géographique (*Geographic Information System* (GIS) en anglais) pour décrire la zone urbaine concernée.

Dans le cadre de cette thèse, nous nous sommes intéressé à l'utilisation des données du site Open Street Map [3] afin d'avoir des informations sur le réseau routier et les structures présentes pour une zone donnée.

Open Street Map est tenu à jour par ses utilisateurs et, par conséquent, les grandes villes disposent en général d'informations fournies et de qualité. Une des manières de construire le réseau routier est d'extraire d'Open Street Map n'importe quelle région et de la transformer en un réseau routier exploitable dans la simulation (figure 4.15).

Les différentes informations présentes dans OSM ont des balises renseignant les particularités des bâtiments ou de la voirie. Un outil a été conçu pour permettre au décideur de filtrer les informations qu'il juge non pertinentes lorsqu'il extrait des données de Open Street Map, c'est-à-dire filtrer les balises qui ne sont pas

3. Open Street Map est un projet collaboratif de créations de fonds de cartes. Ce projet est ouvert à toutes les contributions pour renseigner les fonds de cartes.

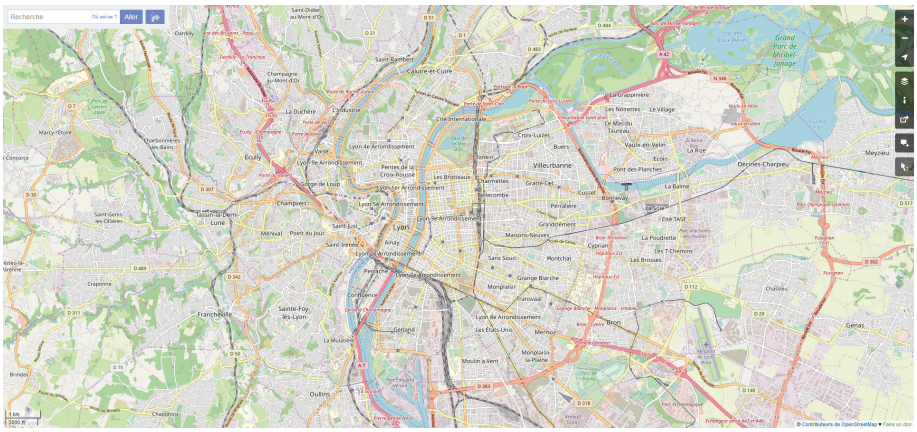


FIGURE 4.15 – Exemple de visualisation disponible de la ville de Lyon avec Open Street Map.

utiles dans la description de son réseau. Quelques exemples de balises qui ne sont pas toujours pertinentes : les pistes de courses, les chemins agricoles, les sentiers de balades hippiques [4].

Ainsi, le décideur extrait une zone particulière correspondant au futur périmètre. Il est possible d'extraire directement les données du site en décrivant la zone concernée ou alors de trouver des bases de données proposant des découpages géographiques plus larges comme un département ou un pays. Il est souvent nécessaire de découper plus finement les zones intéressantes avec un outil adapté. Il filtre ensuite les informations pertinentes pour son scénario. L'outil exploite ensuite le fichier OSM extrait et modifié par le décideur et construit un ensemble de fichiers décrivant les informations nécessaires pour la représentation graphique de l'environnement : les nœuds, les arcs ainsi que le contour des bâtiments.

Enfin, le décideur observe le rendu avec SmartGov (figure 4.16) et l'ajuste si besoin.

Cependant, la structuration de l'environnement avec OSM a des limites car OSM ne dispose pas toujours d'informations exhaustives ou correctes sur la zone géographique concernée. La qualité des informations proposées dépend de l'investissement des utilisateurs et varie beaucoup en fonction du pays. Par exemple la présence des feux ou des emplacements de stationnement sur la voirie sont renseignés de manière plus ou moins exhaustive selon les villes. Le décideur a la possibilité de compléter Open Street Map (à condition de rendre les données utilisées publiques) ou d'utiliser une autre source si celle-ci est plus exhaustive.

4.9.2 Modélisation des agents microscopiques

L'obtention des données sur les usagers et leurs comportements requièrent un travail plus important que l'acquisition de données pour établir un graphe pour l'environnement. Ces données d'usages sont propres à une zone urbaine particulière, à l'évolution sociale et culturelle de la population. Le décideur, lorsqu'il crée la population d'agents microscopiques, met en place une population synthétique décrite par un ensemble d'automates à états finis et de personnalités différentes. Il a la possibilité de collecter ces données avec

4. (Liste des balises existantes pour les routes (en anglais))

FIGURE 4.16 – Exemple de visualisation possible de la zone exportée d'Open Street Map dans SmartGov. Ici, seul le réseau routier avec les principales routes est affiché.

l'utilisation de sondages, d'échanges avec les populations, des enquêtes de ménages. Le rôle de SmartGov n'est donc pas de remplacer le décideur dans la construction ou la collecte de données pour la population synthétique mais d'offrir un support pour faciliter la construction de ces populations. L'usager peut également être directement impliqué en proposant des représentations des agents avec SmartGov.

4.9.3 Perspectives

Il existe une différence fondamentale entre l'architecture générique SmartGov, présenté au long de ce chapitre, et l'implémentation de cette architecture. L'implémentation doit permettre la construction d'une instance de simulation SmartGov avec le couplage entre les deux couches pour la conception de politiques urbaines. Dans le cadre de l'implémentation, les différentes problématiques introduites dans cette section, c'est-à-dire le besoin de disposer d'outils facilitant l'intégration de données est nécessaire. En effet, le décideur doit pouvoir intégrer des données facilement dans SmartGov en limitant le développement nécessaire pour construire une instance de SmartGov incluant ces données. Une certaine partie des données peut être traitée de manière automatique (exemple avec les données de Open Street Map de la section précédente) et d'autres requiert un développement spécifique. Ainsi, les outils pour extraire des données dont le format est connu sont disponibles, notamment l'extracteur de données Open Street Map, cependant, les données avec un format spécifique, pouvant varier d'une instance à une autre nécessite encore un développement spécifique. C'est le cas lorsqu'il s'agit d'utiliser des données pour construire la population synthétique où le décideur doit faire

le lien entre des données d'usages et la structure des agents microscopiques. Une perspective de SmartGov porte sur l'élaboration d'un outil graphique pour simplifier la construction des automates à états finis pour le décideur.

4.10 Conclusion et perspectives pour SmartGov

Ce chapitre a introduit notre première contribution dans cette thèse en proposant une architecture formelle permettant de concevoir un outil d'aide à la décision pour la co-construction de politiques urbaines. Le modèle SmartGov repose sur le couplage de deux simulateurs multi-agents et décrit l'ensemble des interactions entre notre outil et le décideur politique. L'intérêt de notre travail repose sur la description générique des différentes composantes d'une simulation multi-agents pour permettre à un non-expert de s'approprier les concepts clés pour construire sa propre instance de population d'agents capables d'interagir dans un environnement pertinent. Le second point est qu'avec la description de la couche microscopique, la couche macroscopique est en mesure de s'adapter et de proposer des politiques urbaines pertinentes quelle que soit la représentation de la couche microscopique. Le décideur peut ainsi plus facilement mettre en place un outil capable d'automatique rechercher des solutions valides pour proposer des politiques urbaines. De plus, à partir de ces propositions, il peut modifier certains paramètres comme les actions politiques possibles ou le périmètre envisagé afin d'ajuster au mieux la politique. Le couplage entre les deux simulateurs est la clé de l'interaction entre les deux couches et permet à deux simulations multi-agents aux temporalités et portées différentes d'agir ensemble pour simuler l'application d'actions politiques et d'évaluer leurs impacts sur les parties prenantes concernés par ces actions.

Nous avons décrit dans ce chapitre l'architecture générique SmartGov, qui est utilisée pour établir une instance de simulation comprenant les deux couches pour construire une politique urbaine. Il est ainsi possible d'implémenter le modèle SmartGov dans n'importe quel langage de programmation. L'implémentation requiert de disposer d'outils spécifiques permettant de faciliter la tâche du décideur politique dans la construction de l'environnement. Ainsi, un outil comme un constructeur d'automates avec des suggestions en fonction des données disponibles est envisageable. Ce chapitre décrit donc de manière théorique l'ensemble des interactions possible pour permettre au décideur de co-construire une politique afin de fournir une preuve de concept de ces méthodes, cependant elles ne sont pas toutes disponibles dans l'implémentation actuelle de SmartGov et sont des perspectives de développement futures :

— Possibilité de réutiliser une politique urbaine construite avec SmartGov pour procéder à la seconde phase de co-construction en modifiant les paramètres de la politique ;
— Proposition de plusieurs politiques différentes à partir d'une même fonction objectif.

Les perspectives envisagées pour SmartGov reposent sur l'évaluation avec le décideur des moyens d'interactions afin de valider la co-construction des politiques urbaines. De plus, le couplage entre les deux simulateurs repose actuellement sur une structure statique où la manipulation des simulations est déterminée au préalable par le décideur. Nous envisageons de disposer dans le futur d'un outil plus souple, capable d'apprendre et d'améliorer l'interaction entre les deux simulateurs pendant l'étape de simulation. Nous aimerions également

observer comment SmartGov pourrait être utilisé comme intermédiaire pour la co-construction de politiques urbaines entre le décideur et l'utilisateur.

Enfin, la description formelle de la couche macroscopique s'adapte automatique aux différentes représentations de la couche microscopique. Nous proposons dans la suite de cette thèse de s'intéresser à des problématiques d'apprentissage par renforcement multi-agents pour apprendre automatique les spécificités locales de la couche microscopique par les agents politiques et proposer des politiques urbaines pertinentes de manière entièrement autonome dans l'instance de SmartGov.

Un dernier point porte sur l'usage fait par les utilisateurs de SmartGov. En effet, même si un outil d'aide à la décision permet d'être plus transparent dans la co-construction d'une politique urbaine, il revient actuellement au décideur de choisir et d'appliquer la politique urbaine finale. Il peut ainsi imposer une politique urbaine qui pourrait être mal reçue par les usagers sans que cela n'implique que la politique urbaine proposée par SmartGov ne soit pas satisfaisante au regard des objectifs du décideur. En reprenant notre exemple sur la construction d'une politique tarifaire d'emplacements de stationnement, le décideur peut spécifier la fonction objectif suivante : diminuer l'occupation des emplacements aux heures d'affluences. Une instance de SmartGov est créée et la politique proposée suggère d'augmenter les tarifs de manière à ce qu'aucun utilisateur ne stationne par exemple, ou encore instaurer un système de pénalisation (amende, déplacement de la voiture en fourrière) trop autoritaire. D'après la fonction objectif du décideur, la politique est pertinente, cependant, il y a de fortes chances que celle-ci ne soit pas accepter par les usagers.

Chapitre 5

Apprentissage par renforcement individualisé et décentralisé dans un environnement multi-agent non stationnaire

Sommaire

5.1 Introduction

L'idée de l'apprentissage chez la machine (avant d'être envisagée chez l'agent) remonte aux années 1950 où Samuel introduit le terme *Machine Learning* pour son programme de jeu de dames [210]. Mitchell propose la définition suivante, régulièrement utilisée pour décrire le processus d'apprentissage dans le domaine du Machine Learning [167] :

> On dit d'un programme informatique qu'il **apprend** de l'expérience E par rapport à une classe de tâches T et d'une mesure de performance P si ses performances à la classe de tâches T, mesurées par P, augmentent avec l'expérience E.

Pour mettre en place l'apprentissage, il est donc nécessaire de correctement identifier la tâche (jouer au bowling, piloter un drone pour des livraisons, identifier des personnes dans une image), la mesure de performance à améliorer (score à la fin de la partie, pourcentage de livraisons effectuées, nombre de personnes correctement identifiées) ainsi que la source d'expérience (nombre de quilles tombées en fonction de la manière de lancer et du nombre de quilles présentes, séquences d'actions pour déplacer le drone, base de données avec des personnes déjà identifiées). Un des challenges présents dans le domaine du *Machine Learning* est de créer des méthodes pertinentes capables d'apprendre à partir d'entrées spécifiées par la source d'expérience. Ainsi, la mesure de performance est associée à la notion de *feedback* (retour d'information) et dépend du type d'apprentissage envisagé. Russel et Norvig [208] classent en trois grands paradigmes les types d'apprentissage :

- **l'apprentissage supervisé** : l'apprenant dispose d'un ensemble d'informations préalablement labellisées et clairement définies par un superviseur externe où la relation entre les entrées et les sorties associées est explicite. Cet ensemble est également appelé jeu d'entrainement *training data*. L'apprentissage supervisé à de nombreuses applications comme l'identification de courrier indésirable avec la classification naïve bayésienne [209] ou la reconnaissance de motifs [26].
- **l'apprentissage non-supervisé** : l'apprenant identifie et apprend des clusters récurrents progressivement avec l'algorithme de *k-moyennes* [154], sans qu'un retour explicite ne soit fourni, en se basant sur une mesure de qualité des motifs constitués. Ce type d'apprentissage est régulièrement utilisé dans le cadre de la classification pour identifier des ensembles de données similaires comme des extraits audios [144].
- **l'apprentissage par renforcement** : l'apprenant interagit dans un environnement inconnu avec pour objectif de construire un modèle de cet environnement afin d'adopter une stratégie lui permettant de maximiser son profit à long terme. L'apprenant, à l'aide sa boucle sensorimotrice, identifie des signaux positifs ou négatifs, appelé renforcements, correspondant au bénéfice ou au déficit d'avoir effectué une action dans un état donné par rapport à un objectif déterminé.

Dans le cadre de cette thèse, nous nous intéressons à l'apprentissage par renforcement afin de permettre à des agents d'apprendre uniquement à partir de leur boucle sensorimotrice et d'une fonction objectif. Celui-ci trouve de nombreuses applications, nous pouvons citer entre autres le déplacement de robots [135], la maitrise

de jeux vidéo [168] ou de celle du jeu de Go [221]. Nous souhaitons maintenant appliquer ces méthodes à la conception de politiques urbaines pour permettre à des agents d'utiliser la fonction objectif du décideur comme mesure de performance et l'environnement de la couche microscopique comme support de la boucle sensorimotrice.

Dans le précédent chapitre, le modèle SmartGov a été introduit en présentant son architecture générique et le fonctionnement du couplage des dynamiques du modèle. L'objectif étant de pouvoir produire des politiques urbaines de manière générique, la couche haute du modèle SmartGov doit être en mesure d'appliquer des actions politiques et d'identifier une politique ou un ensemble de politiques pertinentes sans connaissances a priori de l'environnement. Afin de comprendre les challenges rencontrés par la communauté et dans cette thèse ainsi que la motivation pour notre seconde contribution, ce chapitre propose un état de l'art sur l'apprentissage par renforcement multi-agents et se découpe de la manière suivante :

— **section 5.2** : présente les prérequis fondamentaux de l'**apprentissage par renforcement** dans un cadre mono-agent, avec les modèles formels des processus de décision markoviens entièrement et partiellement observables ainsi que les notations usuelles utilisées par la communauté. Les contributions récentes l'apprentissage par renforcement profond ainsi que les avantages et limites de ces approches seront également abordés ;
— **section 5.3** : introduit l'**apprentissage par renforcement profond**, comme la combinaison entre l'apprentissage par renforcement et l'apprentissage profond, en décrivant brièvement l'apprentissage profond et son apport dans l'apprentissage par renforcement ;
— **section 5.4** : propose de s'intéresser aux modèles formels associés au contexte multi-agents avec les jeux stochastiques et les Dec-POMPDs dans le cadre de l'**apprentissage par renforcement multi-agent** puis décrit les travaux récents sur l'**apprentissage par renforcement profond multi-agent** combinant les notions mentionnées dans les précédentes sections avec l'apprentissage par renforcement profond dans un contexte multi-agent.

5.2 Cadre formel pour l'apprentissage par renforcement

Avant d'envisager un contexte multi-agents, nous définissons les concepts fondamentaux de l'apprentissage par renforcement pour un unique agent. Les problèmes d'apprentissage par renforcement étant généralement décrits par des processus de décisions markoviens [230, 240], nous proposons dans un premier temps de définir le cadre formel des algorithmes d'apprentissage par renforcement classiques avec la description des processus de décisions markoviens dans le cas général puis dans le cas partiellement observable. Par la suite, nous présentons quelques algorithmes classiques d'apprentissage par renforcement avant d'intégrer les travaux récents sur l'apprentissage par renforcement profond. Afin d'approfondir les notions abordées dans ce chapitre, nous invitons le lecteur à se référer aux ouvrages de référence de Sutton et Barto [230], Russel et Norvig [208] et Kaelbling [122] pour avoir un aperçu plus exhaustif de ces notions.

5.2.1 Processus de décision markovien

Nous adopterons les abréviations anglaises pour la suite des mentions des processus de décision markoviens.

Un *processus de décision markovien* MDP (*Markov Decision Process*) est décrit formellement par le tuple suivant [20] :

$$\langle S, A, T, R \rangle$$

où :

- S représente l'ensemble fini des états ;
- A décrit l'ensemble fini des actions ;
- la fonction de transition $T : S \times A \times S \to [0; 1]$ suit la propriété markovienne et définit l'évolution du système. Ainsi $T(s, a, s')$ est la probabilité d'être dans l'état s' après avoir appliqué l'action a dans l'état s ;
- la fonction de récompense $R : S \times A \times S \to \mathbb{R}$ retourne la récompense $R(s, a, s')$ reçue après avoir appliqué l'action a dans l'état s et avoir atteint l'état s'.

La description d'un MDP considère que l'environnement est entièrement observable. La fonction de transition T d'un MDP possède la propriété fondamentale de Markov, spécifiant que seule la connaissance de l'état actuel est nécessaire pour déterminer l'état futur, la connaissance du passé n'apportant pas plus d'information (équation (5.1)). Par exemple, un joueur d'échec déterminera le coup suivant uniquement à partir de l'état actuel du plateau, les coups joués précédemment n'ayant aucun impact sur sa décision. Cette propriété peut être exprimée de la manière suivante, pour tout $t \geq 0$ et pour toute suite d'états $(s_0, \ldots, s_{t-1}, s_t, s_{t+1}) \in S^{t+2}$:

$$\mathbb{P}(s_{t+1}|s_0, \ldots, s_{t-1}, s_t) = \mathbb{P}(s_{t+1}|s_t) \tag{5.1}$$

Dans une grande partie des problèmes d'apprentissage par renforcement, l'agent doit prendre en compte la récompense immédiate obtenue après avoir agi. Afin de choisir quelle action effectuée, l'agent doit également considérer le prochain état de l'environnement, puisque les récompenses futures qu'il peut obtenir impactera sa prise de décision. Ainsi, la fonction de récompense permet de déterminer la séquence pertinente d'actions pour atteindre un objectif donné à partir de n'importe quelle situation initiale (figure 5.1). Le calibrage de la fonction de récompense a une importance cruciale puisqu'elle détermine le succès ou l'échec de l'apprentissage. Un bon calibrage vise également à limiter le phénomène de sur-apprentissage (*overfitting*) où l'agent obtient des résultats élevés dans une tâche en se spécialisant uniquement sur cette tâche au détriment de ce qu'il a appris précédemment [256].

Résolution d'un MDP

L'objectif de la résolution d'un MDP est de trouver une politique π optimale, notée π^*, respectant l'objectif de l'agent. Cet objectif peut être décrit de différentes manières :

- atteindre un état objectif spécifique ;
- maximiser la récompense totale obtenue quand l'agent effectue certaines actions dans des états

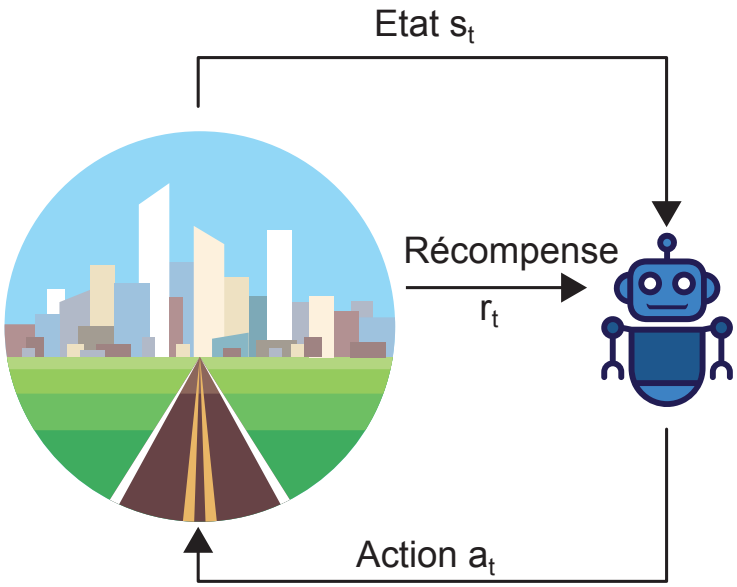


FIGURE 5.1 – L'agent agit sur l'environnement en sélectionnant une action $a \in A$. L'environnement transite de l'état s à s' avec la probabilité $T(s, a, s')$ et à l'instant d'après l'agent perçoit son environnement et reçoit l'état courant p ainsi qu'une récompense $r \in \mathbb{R}$ donnant à l'agent des indications sur la pertinence du choix de a dans s. Il répète ce processus jusqu'à atteindre l'état terminal de la simulation s'il existe.

spécifiques ;
— minimiser le coût des actions en fonction des états.

La politique définit la manière dont l'agent évolue dans l'environnement et correspond à une distribution de probabilités qui associe à chaque état s et à chaque action a la probabilité $\pi(s, a) = \mathbb{P}(a \mid s)$ de choisir l'action a dans l'état s. La politique $\pi : S \times A \to [0; 1]$ détermine l'ensemble des couples états-actions du système. Elle est dite *stochastique*. Une politique est *déterministe* lorsque pour un état s, l'action a est toujours effectuée. Nous notons ainsi $\pi(s)$ l'action que l'agent effectue dans l'état s lorsqu'il suit une politique *déterministe* et $\pi(s, a)$ la probabilité d'effectuer l'action a dans l'état s en suivant une politique *stochastique*. Une politique déterministe est un cas particulier de politique stochastique où la probabilité d'effectuer une action est de 1 et le reste des probabilités est de 0. Nous considérons dans la suite uniquement des politiques stochastiques, notées $\pi : S \to A$ par soucis de simplicité.

Deux grandes catégories sont considérées pour que l'agent puisse maximiser sa récompense et trouver la politique optimale : la recherche itérative de la politique (*policy search*), et l'apprentissage de la valeur de chaque état (*value learning*). La recherche d'une politique évalue la politique initiale puis améliore itérativement la valeur d'une politique donnée. Il faut ensuite améliorer la politique jusqu'à déterminer une politique optimale, maximisant la récompense. Au contraire, un algorithme d'apprentissage des fonctions de valeurs évalue la pertinence des couples état-action pour converger vers la fonction de valeur de la politique optimale. Certaines méthodes de programmation dynamique [19] requièrent la connaissance de la fonction de

récompense et la fonction de transition, c'est le cas par exemple avec l'algorithme d'itération de la politique d'Howard [109] (*policy iteration*).

Il existe deux fonctions de valeurs V et Q régulièrement rencontrées dans le domaine de l'apprentissage par renforcement, et qui sont notamment utilisées pour les algorithmes abordés ultérieurement :

— $V : S \rightarrow \mathbb{R}$ est la fonction de valeur décrivant la pertinence pour le système de se trouver dans l'état $s \in S$;

— $Q : S \times A \rightarrow \mathbb{R}$ est la fonction de valeur décrivant la pertinence d'effectuer l'action $a \in A$ dans l'état $s \in S$.

À partir de ces informations, l'algorithme de recherche de la meilleure politique calcule une fonction de valeur V et affine la politique π où l'on cherche la succession d'action à effectuer. À la fin de l'algorithme, π contient la solution et $V(s)$ contient l'espérance de la somme atténuée des récompenses à obtenir en suivant la politique π à partir de l'état s. Le facteur d'atténuation (*discount factor*) $\gamma \in [0, 1)$ indique l'importance des futures récompenses pour l'agent. Si γ est égal à 1, les récompenses sont dites additives et ont toutes la même importance ; sinon les récompenses sont dites réduites. D'autre part, plus γ tend vers 0, moins les récompenses futures ont d'importance. L'algorithme, pour obtenir la solution optimale, répète une mise à jour de la valeur et de la politique jusqu'à qu'il n'y ait plus d'amélioration de la valeur de la politique.

La valeur optimale d'un état est définie par :

$$V^*(s) = \max_{\pi} \mathbb{E}\left(\sum_{t=0}^{\infty} \gamma^t r_t\right) \tag{5.2}$$

où $\mathbb{E}$ est l'espérance et r_t est la récompense obtenue à l'instant t après avoir effectuée une action $\pi^*(s)$.

Ainsi, pour un MDP fini avec $\gamma \in [0 : 1)$ la fonction de valeur optimale est l'ensemble des solutions de l'équation de Bellman [20], définie par :

$$V^*(s) = \max_{a} \left(R(s, a) + \gamma \sum_{s' \in S} T(s, a, s')V^*(s')\right), \forall s \in S \tag{5.3}$$

qui signifie que la valeur de l'état s est la récompense instantanée attendue plus la récompense atténuée du prochain état en utilisant la meilleure action. À partir de cette fonction de valeur, la politique optimale est donc pour un état s :

$$\pi^*(s) = arg\max_{a} \left(R(s, a) + \gamma \sum_{s' \in S} T(s, a, s')V^*(s')\right), \forall s \in S \tag{5.4}$$

L'algorithme d'itération des valeurs [230] permet d'obtenir la valeur de fonction optimale V^*. Cet algorithme est optimal sur un horizon H et il converge quand le nombre d'itérations tend vers l'infini. Cela permet ensuite de calculer π^*.

La fonction de transition T et de la fonction de récompense R doivent d'être connues pour que l'algorithme

Algorithme 1 : Itération des valeurs

```
1 Initialiser V(s) aléatoirement (ex : V(s) = 0, ∀s ∈ S)
2 while Δ < θ do
3 |  Δ ← 0 for s ∈ S do
4 |  |  temp ← V(S)
5 |  |  V(s) ← max_a Σ_{s'} T(s'|s,a)[R(s,a,s') + γV(s')]
6 |  |  Δ ← max(Δ, |temp − V(s)|)
7 |  end
8 end
```

fonctionne, ce qui représente une hypothèse forte sur l'environnement puisqu'il n'est pas toujours possible de connaitre a priori le modèle de l'environnement (opposition entre les approches où le modèle est connu (*model-based*) et où le modèle est inconnu (*model-free*)). L'algorithme d'itération des valeurs permet de comprendre les fondements de l'algorithme de Q-learning que nous verrons dans la ligne 8.

Algorithme de Q-Learning

L'algorithme de Q-learning est populaire pour sa facilité d'implémentation et d'utilisation. Introduit par Watkins [253], il utilise les différences temporelles [229] (*temporal differences*) et fonctionne en mettant à jour la fonction de valeur $Q(s,a)$ en utilisant la récompense obtenue pour avoir effectué l'action a dans l'état s et en agissant de manière optimale dans l'état obtenu s'. On suppose que l'agent agit toujours de manière optimale dans chaque état, c'est-à-dire que l'on choisit l'action qui maximise la fonction de valeur Q. Par conséquent, la fonction de mise à jour de la fonction de valeur Q dans l'algorithme de Q-learning est décrite par :

$$Q(s,a)^{new} \leftarrow (1-\alpha)\cdot Q(s,a) + \alpha \cdot \overbrace{\left(r + \gamma \cdot \max_a Q(s',a)\right)}^{\text{valeur apprise}} \tag{5.5}$$

où :

- $Q(s,a)$ est l'ancienne valeur associée à un état s et à une action a ;
- $\alpha \in]0;1]$ est un coefficient d'apprentissage ;
- r est la récompense obtenue après avoir effectué l'action a dans l'état s ;
- $\max_a Q(s',a)$ est l'estimation de la valeur de l'état s' obtenu en agissant de manière optimale dans l'état s.

Le Q-learning converge [236, 253] à condition que tous les couples état-actions soient visités un nombre infini de fois et que les ensembles S et A soient finis. Il est évidemment irréalisable de parcourir une infinité de fois chaque couple état-action. Dans la pratique, le parcours est réalisé un grand nombre de fois, une exploration partielle étant généralement suffisante [159]. Il devient alors nécessaire de trouver un compromis pertinent entre l'exploration et l'exploitation que nous aborderons dans la section suivante.

L'avantage du Q-learning est qu'il n'est pas lié au modèle du MDP, il est dit *model-free* (modèle libre) par

Algorithme 2 : Q-learning

Initialiser $Q(s, a)$ aléatoirement $\forall s \in S, a \in A$
Initialiser l'état initial s
while *S n'est pas terminal* **do**
    Choix de l'action a à partir de Q et s
    Application de l'action a et passage à s' avec la récompense r
    $Q(s, a) \leftarrow (1 - \alpha) \cdot Q(s, a) + \alpha \cdot \left(r + \gamma \max_{a \in A} Q(s', a)\right)$
    $s \leftarrow s'$
end

opposition aux approches où le modèle est connu à l'avance ou *model-based*. Bien qu'il existe des travaux sur ces deux types d'approche, les approches *model-free* seront au cœur de cette thèse, notamment l'algorithme de Q-learning, afin de pouvoir doter nos agents d'un algorithme d'apprentissage valide notamment lorsque l'environnement où évoluent les agents n'est pas connu à l'avance.

Processus de décision markovien partiellement observable

Lorsque l'agent n'a pas accès à la totalité des informations de son environnement, on dit que l'environnement est *partiellement observable* [226]. L'agent reçoit alors une observation incomplète de l'état et doit conserver un historique des observations pour mettre à jour un niveau de croyance sur les états (figure 5.2).

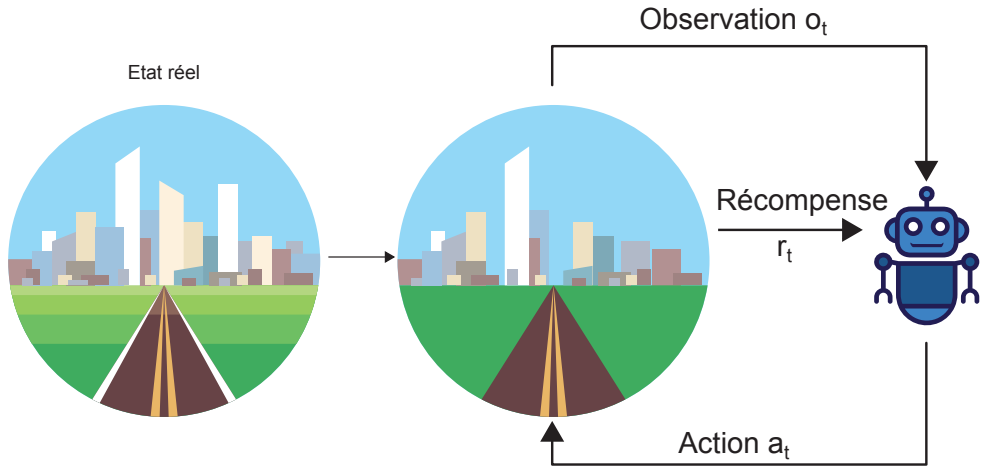


FIGURE 5.2 – L'agent reçoit une observation o_t correspondant à une description partielle de l'environnement. Il utilise un estimateur d'états pour obtenir une hypothèse de l'état réel à partir de son observation et propose une action a_t pertinente en fonction de cette estimation.

Une extension du MDP aux environnements partiellement observables est le POMDP (*Partially Observable*

Markov Decision Process). Il est formellement décrit par le tuple suivant [122] :

$$\langle S, A, T, R, \Omega, O \rangle$$

où :

— S, A, T, R sont identiques à la description du MDP (section 5.2.1) ;
— Ω est l'ensemble fini des observations que l'agent peut obtenir ;
— $O : S \times A \to [0, 1]$ est la fonction d'observation renvoyant, pour chaque action et chaque état, une distribution de probabilités sur les observations possibles.

Le POMDP décrit l'environnement partiellement observable dans le cadre mono-agent et sa résolution sort du cadre de cette thèse mais est disponible dans différentes revues [43, 174]. Cependant, le POMDP est un concept intéressant dont la version décentralisée sera approfondie lorsque nous aborderons l'apprentissage par renforcement multi-agents.

5.2.2 Vitesse d'apprentissage : compromis entre exploration et exploitation

Une problématique existante dans la cadre de l'apprentissage par renforcement est le besoin d'équilibre entre l'exploration des états et l'exploitation des solutions. En général, l'exploration sert à essayer les actions peu essayées jusqu'à présent ou pour lesquelles l'apprenant ne possède pas beaucoup d'informations. L'exploration permet donc d'approfondir la connaissance du modèle pour améliorer la récompense cumulée, au risque de perdre du temps si on est déjà à l'optimal. L'exploitation consiste à utiliser l'apprentissage pour maximiser le gain en choisissant les actions gloutonnes ($arg\max_a Q(s, a)$). Une exploitation trop poussée, sans avoir suffisamment exploré l'espace des solutions, peut par exemple accroitre le risque de régulièrement choisir une action produisant une politique sous-optimale.

Il existe de multiples façon d'explorer plus ou moins intelligemment l'espace des actions et d'obtenir un meilleur compromis entre exploration et exploitation. En effet, si on effectue une action aléatoire à chaque fois que l'on souhaite explorer, il est tout à fait possible d'effectuer une séquence d'actions optimales, ce qui apporterait peu d'informations à l'apprenant sur des états jamais explorés. La méthode la plus simple, appelée ϵ-greedy, repose sur un paramètre ϵ utilisé pour l'exploration avec une probabilité ϵ et l'exploitation avec une probabilité $1 - \epsilon$ [230]. Pour forcer l'exploration dans un premier temps, la valeur d'ϵ sera proche de 1 et diminuera au cours de l'apprentissage pour que l'agent exploite plus souvent sa politique. Ce compromis entre exploration et exploitation repose aussi sur la fonction de décision qui conduit au choix de l'action à effectuer dans chaque état. La méthode du bandit à n-bras permet également de trouver un compromis entre exploration et exploitation [24] et s'inspire du principe des machines à sous. Ce nom fait référence à la situation d'un joueur faisant face à une lignée de machines à sous et cherchant sur laquelle tenter sa chance afin de maximiser ses gains. La méthode du bandit à n-bras consiste à explorer chaque bras au moins une fois et à ensuite exploiter le bras le plus intéressant. L'algorithme UCB (*Upper Confidence Bound*, limites supérieures de confiance) [11] est une approche dite "optimiste" de résolution du compromis entre exploration et exploitation : l'agent détermine le bras qui lui rapporte le plus haut gain espéré. Cependant, pour se faire

une idée des bras intéressant, il est nécessaire d'explorer au moins une fois chaque bras, ce qui n'est pas toujours envisageable lorsque le nombre d'états est vaste. De plus, l'exploration peut être couteuse, d'autant plus que certains états n'ont pas à être explorés pour avoir l'intuition qu'ils ne sont pas pertinents pour la politique. La méthode calcule ainsi la moyenne des récompenses pour chacune des machines afin de déterminer quelle machine exploiter en se fiant non pas à la performance estimée d'un bras, mais à une borne supérieure de confiance.

Il existe d'autres approches pour résoudre le problème d'exploration et d'exploitation en fonction du problème abordé [230]. Dans le cadre de cette thèse, nous utiliserons l'approche ϵ-greedy avec atténuation de l'ϵ pour sa facilité de mise en place.

5.2.3 Conclusion et limites de ces approches

Cette section a introduit les fondements de l'apprentissage par renforcement avec le formalisme du MDP, qu'une majorité des problèmes d'apprentissage utilise, ainsi que les algorithmes fréquemment utilisés comme le Q-learning, pour la classe d'itération des valeurs. Le compromis entre exploration et exploitation, nécessaire pour produire une meilleure politique, est également abordé. Le problème avec ces approches classiques est la nécessité de la représentation tabulaire des couples états-actions du système. Dans les années 2000, des approches ont été utilisées pour envisager d'avoir une approximation de la représentation de l'état avec des règles floues [87] ou des cartes auto-organisatrices [234] par exemple. Une autre approche est l'acteur-critique (*actor-critique*) utilisant deux mémoires séparées pour un agent : l'*acteur* et la *critique*. L'acteur sélectionne une action à partir de l'observation de son état et la soumet à la critique pour évaluation. Ce verrou, présent dans l'apprentissage par renforcement classique, est lié au passage à l'échelle limité dû au fléau de la dimension (*curse of dimensionality*) et la prise en compte d'un nombre croissant d'états. Récemment, la communauté de l'apprentissage par renforcement a obtenu une résolution de ces verrous avec l'introduction de l'apprentissage par renforcement profond.

5.3 Récents succès de l'apprentissage par renforcement profond

L'apprentissage par renforcement profond, abrégé par DRL pour *Deep Reinforcement Learning*, combine l'apprentissage par renforcement et les avancées en apprentissage profond (*deep learning* (DL)) [143]. Par conséquent, l'apprentissage est dit profond puisqu'il combine les réseaux de neurones multicouches avec des algorithmes d'apprentissage par renforcement comme le Q-Learning. Une importante propriété du DL est la possibilité d'extraire des représentations de bas niveau à partir de vecteurs de grandes dimensions. Le réseau de neurones multicouches agit comme une fonction d'approximation universelle et permet donc de manipuler des espaces d'états et d'actions bien plus vastes et ainsi de mieux passer à l'échelle car il est moins impacté par le fléau de la dimension [22, 10].

L'apprentissage par renforcement profond a récemment gagné en popularité dans la communauté de l'apprentissage en proposant des solutions aux verrous identifiés dans les approches classiques de l'apprentissage par renforcement. Les deux points forts du DRL sont la fonction d'approximation (*function approximation*)

et l'apprentissage de représentations (*representation learning*) [10] qui reposent sur l'utilisation de réseaux de neurones. L'apprentissage de représentations facilite l'identification d'informations pertinentes [22]. Plusieurs travaux sont à l'origine de l'intérêt récent pour le DRL, notamment les travaux de Mnih *et al.* sur le développement d'un algorithme capable d'atteindre un niveau de maitrise supérieur à l'humain sur des jeux Atari en utilisant directement comme entrée les pixels de l'image [168, 169]. L'algorithme qu'ils proposent, le Deep Q-Network (DQN), combine un réseau de neurones recevant des images brutes comme perception de l'environnement.

Même si la fonction d'approximation est déjà utilisée dans l'apprentissage par renforcement classique, l'apport des travaux de Mnih *et al.* montrent comment le recours aux réseaux de neurones permet de stabiliser la fonction d'approximation pour apprendre dans des environnements avec un nombre d'états important. Plus récemment, en 2017, AlphaGo a vaincu un champion du monde de Go en utilisant des réseaux de neurones couplé à de l'apprentissage supervisé et de l'apprentissage par renforcement et en apprenant en jouant contre lui-même (*self-play*) [221].

5.3.1 Algorithme de *Deep Q-Network*

L'approche de Mnih *et al.* [168] utilise des images comme perception des agents, par conséquent, le DQN est couplé à un Réseau Neuronal Récurrent (*Recurrent Neural Network* (RNN)) [207]. Les jeux Atari explorés dans leurs travaux ont un état terminal lorsque l'agent meurt ou perd la partie. Il reçoit ainsi une observation de l'environnement auquel il associe un état s.

Algorithme 3 : Deep Q-Network with Experience Replay [168]

Initialiser la mémoire $\mathcal{M}$ de taille N
Initialiser $Q(s,a)$ aléatoirement $\forall s \in S, a \in A$
`/* J est le nombre maximal d'époques */`
for *époque = 1, 2, ..., J* **do**
  `/* T est le nombre maximal d'itérations pour une époque */`
  for $t = 1, 2, \ldots, T$ **do**
    Avec une probabilité ϵ, choisir une action aléatoire a_t
    sinon choisir l'action $a_t = \max_a Q^*(s_t, a; \theta)$
    Exécuter l'action a_t et recevoir la récompense r_t et l'état s_{t+1}
    Stocker l'expérience (s_t, a_t, r_t, s_{t+1}) dans $\mathcal{M}$
    Prendre un échantillon des expériences (s_j, a_j, r_j, s_{j+1}) dans $\mathcal{M}$
    Poser $y_i = \begin{cases} r_j & \text{si état terminal} \\ r_j + \gamma \max_{a'} Q(s_{j+1}, a'; \theta) & \text{si état non terminal} \end{cases}$
    Effectuer une descente de gradient sur $(y_i - Q(s_j, a_j; \theta))^2$
  end
end

Le réseau de neurones sert de fonction d'approximation transformant les images brutes de l'environnement en états exploitables par l'agent pour sélectionner une action. Le réseau de neurones est paramétré par θ qui

est continuellement entrainé pour atteindre une politique optimale. Le DQN repose sur l'équation de Bellman pour minimiser la fonction de perte $\mathcal{L}(\theta)$ telle que :

$$\mathcal{L}(\theta) = \mathbb{E}\Big[\big(r + \lambda \max_{a'} Q(s', a'|\theta) - Q(s, a|\theta)\big)\Big] \tag{5.6}$$

où $\mathbb{E}$ représente l'espérance. Avec cette fonction, les représentations de l'environnement sont apprises et affinées.

Pour comprendre les verrous auxquels la communauté de l'apprentissage profond est confrontée, il est intéressant d'introduire les différentes composantes de l'algorithme du DQN.

Réutilisation de l'expérience acquise

Une faiblesse des réseaux de neurones est liée au phénomène de sur-apprentissage des paramètres (*overfitting*). Au fur et à mesure que l'agent reçoit des observations de l'environnement, la fonction de perte est mise à jour avec l'équation (5.6). Le réseau de neurones modifie à chaque fois ses attributs pour affiner sa fonction d'approximation. Nous avions introduit précédemment l'équilibre à atteindre entre exploration et exploitation. Une fois qu'un agent a suffisamment exploré son environnement, il commence à exploiter sa politique pour maximiser les récompenses obtenues. La conséquence de cette exploitation est que le réseau de neurones oublie la politique π qu'il a apprise pour se concentrer sur une approximation des états exploités. Un autre problème est que les réseaux de neurones ont des difficultés pour apprendre différentes tâches successivement, connu sous le nom de *catastrophic forgetting* [131] (oubli catastrophique) car l'agent oublie la tâche précédemment apprise.

Dans son travail de thèse, Lin utilise le terme d'*expérience* pour décrire la transition opérée par l'agent à chaque pas de temps $e_t = (s_t, a_t, r_t, s_{t+1})$ [148] correspondant à l'action a_t effectuée dans l'état s_t, l'état résultant s_{t+1} et la récompense associée r_t à l'application de a_t dans s_t. Lin propose de réutiliser l'expérience (*experience replay*) acquise et de la stocker afin de réutiliser les connaissances acquises de manière off-line. La réutilisation de l'expérience présente plusieurs intérêts : d'une part l'accélération de la propagation de la récompense obtenue, particulièrement intéressante dans les environnements où l'acquisition d'informations est coûteuse, et d'autre part la diminution du sur-apprentissage en réutilisant l'expérience acquise précédemment [148]. Ainsi, en stockant les expériences, il est possible de les réutiliser pour réapprendre ce qui a été exploré précédemment.

L'une des nouveautés proposée par Mnih *et al.* dans leurs travaux est d'exploiter la réutilisation d'expérience pour mettre à jour l'algorithme du DQN en rejouant une partie des expériences tirée de manière aléatoire dans l'ensemble de la mémoire d'expériences. Le premier avantage est la mise à jour des poids du réseau de neurones, permettant d'accroître l'efficacité de l'acquisition de données (phase d'exploration moins longue avec la conservation de l'expérience). Le second avantage est l'utilisation d'expériences aléatoirement tirées plutôt que de rejouer les dernières expériences, afin d'éviter la corrélation entre les échantillons et d'améliorer l'apprentissage. Enfin, le dernier intérêt est qu'en rejouant sur des échantillons aléatoires, le système évite de n'explorer qu'une seule partie de l'environnement et d'accumuler un ensemble d'expériences moins pertinentes

qui, sur le long terme, peuvent réduire la qualité de la politique.

Récompenses tronquées

La seconde contribution de leurs travaux justifie la robustesse de leur approche où, avec une architecture unique, le même algorithme d'apprentissage et des hyper-paramètres identiques, il est possible d'entrainer des réseaux pour différents jeux et d'obtenir des performances supérieures à celle d'un humain. L'utilisation des récompenses tronquées (*clipped rewards*), correspondant à la réduction de toutes les récompenses dans l'intervalle $[-1, 1]$, permet de limiter le besoin de connaissances expertes définies a priori pour le système et permet de conserver les mêmes paramètres pour l'apprentissage entre différents jeux. En effet, certains jeux comme Pong utilisent naturellement une récompense de 1 ou -1. Alors que détruire les vaisseaux extra-terrestres dans Space Invaders incrémente le score de 5, 10 ou 15 points par exemple en fonction du vaisseau et 200 points pour le vaisseau amiral. Ou encore, dans le jeu Ms. Pac-Man, manger un fantôme donne une récompense de 100 alors que manger la nourriture au sol donne une récompense de 25. Par conséquent, quelle que soit la récompense (5,10,15 ou 200) pour Space Invaders, l'apprenant reçoit une récompense tronquée à 1 et une récompense de -1 lorsqu'il perd le jeu. Les récompenses tronquées ne prennent pas en compte la différence entre les points et perd la notion que le vaisseau amiral est plus *important* que les autres vaisseaux.

Néanmoins, l'apprentissage est plus autonome face à des contextes différents et obtient de meilleures performances qu'avec les récompenses classiques mais il perd une partie de l'information sur les différences entre les conséquences de ses actions [169, 243]. Les travaux de van Hasselt *et al.* proposent de normaliser les récompenses plutôt que de les réduire pour que l'agent identifie une différence dans l'importance de ses récompenses [243]. Cependant, normaliser les récompenses peut poser problème lors de l'exploration car l'état optimal et donc le meilleur gain atteignable est a priori inconnu.

5.3.2 Perspectives de l'ajout des réseaux de neurones dans l'apprentissage par renforcement

L'apprentissage profond apporte l'usage d'une fonction d'approximation sous la forme de réseau de neurones pour gérer des espaces d'états et d'actions plus grands que les versions tabulaires de l'apprentissage par renforcement classique. Là où le Q-Learning doit explorer chaque couple état-action pour déterminer une politique, le DQN extrait des motifs à partir de données d'entrées dépendantes du problème (par exemple des images brutes d'un jeu vidéo [168, 138]) pour construire un état de l'environnement. Les travaux sur la combinaison de l'apprentissage profond avec l'apprentissage par renforcement [168, 169, 221] ont changé le visage de l'apprentissage par renforcement en permettant d'envisager des espaces d'états et/ou d'actions continus. L'autonomie de ces approches est également accrue dans la mesure où, avec la fonction d'approximation, l'algorithme peut proposer une action pertinente dans des états qu'il n'a encore jamais explorés. Ces travaux sont appliqués à un agent unique dans l'environnement. Le contexte de notre travail gagne à disposer d'une approche décentralisée où plusieurs agents vont apprendre en même temps. Cependant, dans un cadre multi-agents, de nouveaux problèmes apparaissent, notamment la difficulté de trouver un bon

objectif d'apprentissage pour les agents. Quel sera l'impact de l'apprentissage concurrent dans ce cas ? La prochaine section s'intéresse donc à l'apport de l'apprentissage par renforcement profond dans un contexte multi-agents avec les avantages et les limites de ces approches.

5.4 Application au contexte multi-agents

L'apprentissage par renforcement multi-agent, appelé *Multi-Agent Reinforcement Learning* (*MARL*) par la communauté anglophone, a pour objectif de proposer des algorithmes permettant à un ensemble d'agents autonomes d'apprendre par exemple des tâches complexes, de coopérer en disposant de communications limitées ou encore de maximiser un gain individuel au détriment des autres agents. Il existe plusieurs types de tâches dans le MARL en fonction de l'objectif des agents. Dans un cadre coopératif, les agents maximisent une récompense jointe. Dans un cadre compétitif, les agents maximisent leurs bénéfices individuels au détriment des autres agents. On trouve aussi des approches mixte (ni entièrement compétitive, ni entièrement coopératives) [38]. De multiples agents interagissant dans un environnement dynamique et non déterministe ne permettent pas d'élaborer un comportement pour chaque agent à l'avance, l'apprentissage est donc crucial pour le système que nous voulons mettre en place. Dans le cadre de cette thèse, l'apprentissage a pour objectif de construire des politiques urbaines, nous considérons donc les tâches a priori coopératives où les agents coopèrent pour satisfaire la fonction objectif du décideur. Cependant, nous verrons que dans un contexte d'apprentissage individualisé et décentralisé, l'agent cherche plutôt à satisfaire ses propres intérêts, sans pour autant être en compétition avec d'autres agents. Par conséquent, les tâches que nous considérons relèvent davantage des approches mixtes.

L'apprentissage par renforcement profond multi-agent ou *Multi-Agent Deep Reinforcement Learning (MADRL)* rapproche deux domaines en utilisant les progrès récents dans le domaine du DRL dans un contexte multi-agent [36]. Le premier travail combinant ces deux approches peut être celui de Tampuu *et al.* proposant à deux joueurs de s'affronter au jeu de Pong [231] s'inspirant des travaux effectués en DRL sur les jeux Atari [168]. Ils explorent un contexte mixte dans lequel la récompense détermine le type de joueur (compétitif ou collaboratif) présent dans le jeu et proposent un équivalent d'*Independent Deep Q-Learner* basé sur un Deep Q-Network [168] par agent.

Nous consacrons cette section au MARL et au MADRL en commençant par les challenges que posent cette discipline puis par une présentation de son contexte formel.

5.4.1 Motivation

Le besoin de décentralisation dans l'apprentissage multi-agents permet une représentation plus réaliste des systèmes complexes où ses composantes sont régulièrement des entités indépendantes qui partagent leurs connaissances, par exemple en communiquant. Le système multi-agents est robuste par nature en éliminant le point individuel de défaillance des systèmes centralisés. Ainsi, si un ou plusieurs agents apprennent une mauvaise politique par exemple, le système est toujours en mesure de fonctionner avec les autres agents. En

distribuant l'apprentissage, les agents peuvent également apprendre en parallèle, que ce soit sur la même tâche ou sur une tâche différente.

Le système multi-agents apparait comme une réponse aux contraintes de passage à l'échelle rencontré dans l'apprentissage par renforcement mono-agent. Ains, en fragmentant l'environnement, il est possible de disposer d'agents dont l'ensemble d'états est moins conséquent. L'agent ne doit plus nécessairement avoir une vision globale avec l'intégration de tous les capteurs et actionneurs à chaque itération. Le passage à l'échelle en utilisant un apprentissage multi-agents est plus facile à réaliser que dans un cadre centralisé.

5.4.2 Challenges liés au passage multi-agents

Cependant, le passage de l'apprentissage par renforcement mono-agent à l'apprentissage par renforcement multi-agents fait émerger un ensemble de nouveaux verrous. En effet, chaque agent présent dans l'environnement va agir en même temps que les autres. L'ensemble de ces actions, également appelée *action jointe*, est appliquée pour modifier l'environnement.

Tuyls et Tumer posent les deux questions suivantes dans le cadre du MARL [239] :

— Comment les agents prennent-ils en compte l'action jointe effectuée par les autres agents du système ?
— Comment les agents choisissent-ils des actions permettant d'apporter un bénéfice immédiat mais également de structurer les actions des autres agents dans le futur ?

Ces notions se réfèrent à la complexité de l'apprentissage concurrent que l'on rencontre dans le cadre multi-agents et plus particulièrement à la notion de non-stationnarité de l'environnement. Ces deux questions sont abordées dans cette section et notre contribution cherche également à y répondre.

Environnements non-stationnaires et impact dans le MADRL

Un environnement est dit *stationnaire* lorsque les probabilités des transitions et des récompenses sont indépendantes du temps. Ainsi, quel que soit l'itération où l'état est rencontré, la meilleure action à appliquer dans cet état est toujours la même. Un agent unique dans un système n'est concerné que par les conséquences des actions qu'il effectue. Cependant, dans un contexte multi-agents, l'agent n'observe plus seulement les conséquences de ses actions mais également les conséquences des actions des autres agents. Lorsque des agents apprennent de manière concurrente, l'environnement devient non-stationnaire. Ainsi, en même temps qu'ils apprennent de leur environnement, ils modifient leur politique, ce qui impacte la fonction de transition. Le résultat de l'action a dans l'état s valable au temps t ne le sera pas nécessaire au temps $t+1$. L'interaction entre les différents agents modifie régulièrement la politique des autres agents et affectent également leur politique optimale [177].

La non-stationnarité a un impact sur la preuve de convergence du Q-learning puisque la propriété de Markov n'est plus respectée [240]. Hernandez-Leal *et al.* proposent de formaliser la non-stationnarité de la manière suivante [98]. Soit $\mathcal{N}$ l'ensemble des agents apprenants. Considérons un agent apprenant $i \in \mathcal{N}$ et notons $-\boldsymbol{i} = \mathcal{N} \setminus \{i\}$ l'ensemble des autres agents apprenants. La fonction de valeur dépend de l'action jointe

$\boldsymbol{a} = (a_i, \boldsymbol{a_{-i}})$ et la politique jointe est égale à $\boldsymbol{\pi}(s, \boldsymbol{a}) = \Pi_j \pi_j(s, a_j), j \in \mathcal{N}$:

$$V_i^{\boldsymbol{\pi}}(s) = \sum_{\boldsymbol{a} \in A} \boldsymbol{\pi}(s, \boldsymbol{a}) \sum_{s' \in S} T(s, a_i, \boldsymbol{a_{-i}}, s')[R(s, a_i, \boldsymbol{a_{-i}}, s') + \gamma V_i(s')]$$

Par conséquent, la politique optimale de l'agent i est la meilleure réponse (MR) en fonction de la politique des autres agents :

$$\pi_i^*(s, a_i, \boldsymbol{\pi_{-i}}) = MR_i(\boldsymbol{\pi_{-i}}) = arg\max_{\pi_i} V_i^{(\pi_i, \boldsymbol{\pi_{-i}})}(s)$$

$$= arg\max_{\pi_i} \sum_{\boldsymbol{a} \in A} \pi_i(s, a_i) \boldsymbol{\pi}(s, \boldsymbol{a_{-i}}) \sum_{s' \in S} T(s, a_i, \boldsymbol{a_{-i}}, s')[R(s, a_i, \boldsymbol{a_{-i}}, s') + \gamma V_i^{(\pi_i, \boldsymbol{\pi_{-i}})}(s')]$$

Comme nous l'avons abordé dans les succès récents sur l'apprentissage par renforcement profond, l'une des contributions de Mnih *et al.* est la réutilisation d'expériences acquises pour alimenter le réseau de neurones [168]. Cependant, dans un environnement non-stationnaire, l'expérience qui est enregistrée peut rapidement être non-pertinente pour l'apprentissage. L'usage des réseaux de neurones permet d'utiliser des espaces d'états et d'actions plus grands [203, 231] cependant la non-stationnarité de l'environnement multi-agent impacte négativement l'*experience replay* [77, 186] et plusieurs travaux cherchent à améliorer sa stabilité comme les travaux de Foerster *et al.* [77] dans le cadre d'un apprentissage centralisé. Leibo *et al.* [145] proposent de réduire la taille de la mémoire pour l'*experience replay* et Foerster *et al.* [78] de le supprimer entièrement. Cependant, ces approches sont plus impactées par le fléau de la dimension [203].

Nous abordons ce verrou dans notre seconde contribution de thèse afin de permettre l'usage de la réutilisation d'expérience dans un cadre non-stationnaire.

Identification de la contribution de chaque agent

Dans un contexte coopératif, la transition de l'environnement $T(s, \boldsymbol{a}, s')$ repose sur l'action jointe $\boldsymbol{a}$ effectuée par l'ensemble des agents. Cependant, comment l'agent identifie-t-il s'il a contribué à l'évolution des performances du système ou si ces modifications sont liées à un autre élément de l'environnement ? Afin de pouvoir mettre à jour sa politique à partir de l'action qu'il a effectuée et de son état, l'agent a besoin d'une récompense représentant sa contribution au système. Ce verrou, appelé *multiagent credit assignment problem* en anglais, est le problème lié à l'attribution d'une récompense pour un agent donné basée sur la performance d'un ensemble d'agents [244].

L'attribution de la récompense dépend du problème considéré [239] :

1. les agents reçoivent tous la récompense globale du système. Bien que cette approche fonctionne avec un nombre restreint d'agents, elle est moins efficace quand le nombre d'agents augmente et réduit la qualité de l'apprentissage puisque les agents évaluent plus difficilement leurs contributions ;
2. les agents reçoivent une partie de la récompense globale du système. L'agent n'a plus besoin d'avoir l'information sur la totalité de l'état mais uniquement les composantes locales de son état. Cette récompense ne dépend normalement pas des autres agents. Cette approche peut être impactée par la factorisation du problème où il est possible qu'une bonne récompense pour l'agent ne signifie pas une

amélioration de performances au niveau global ;

3. les agents calculent une récompense égale à la différence entre la récompense du système avec l'action effectuée et la récompense du système si l'agent n'avait pas effectué d'action. L'avantage de cette méthode est qu'il est plus facile d'identifier les contributions de chaque agent, le désavantage est qu'elle requiert une connaissance de l'état global du système.

Dans le cadre de notre étude, nous considérons le second cas avec des récompenses locales pour les agents.

Problème de la cible mouvante

Un agent apprend à atteindre une cible au tir à l'arc en calibrant son tir et en apprenant au fur et à mesure à faire les bons gestes pour atteindre sa cible régulièrement. Si pour une raison quelconque, un second agent déplace la cible de deux mètres en arrière, l'agent devra à nouveau apprendre à calibrer son tir pour atteindre la cible. Ce problème du *moving target* est lié au fait d'essayer d'atteindre un objectif que l'on pense être optimal mais qui peut changer pendant l'apprentissage [246, 36]. Le terme vient initialement de travaux portant sur des prédateurs poursuivant des proies pour les capturer. En effet, à cause de la non-stationnarité de l'environnement, une politique optimale à un instant t ne l'est plus forcément à l'instant $t + \Delta t$.

Il s'agit d'un verrou de l'apprentissage multi-agents puisqu'un agent peut produire une politique optimale π^* par rapport à son environnement à un instant t et exploiter cette solution et avoir des difficultés à s'adapter aux changements de l'environnement quand les autres apprenants modifient leurs propres politiques locales. Dans le cadre de notre thèse, les approches que nous envisageons sont concernées par le problème de la cible mouvante.

Difficultés liées à l'augmentation du nombre d'agents

L'ajout d'agents fait croître de manière exponentielle la taille de l'espace d'actions jointes rendant complexe le calcul de toutes les solutions, et augmentant le coût de l'exploration et de l'exploitation des différentes solutions. Le problème du fléau de la dimension, déjà présent dans le cadre de l'apprentissage par renforcement mono-agent, est accentué ici avec l'augmentation du nombre d'agents.

Les verrous énoncés précédemment ont un impact sur le passage à l'échelle des approches multi-agents pour la gestion d'un espace plus complexe. Le type d'apprenant peut donc jouer sur le passage à l'échelle en réduisant l'espace d'actions à prendre en compte dans le cadre d'apprenants indépendants par exemple. Ces approches ont leurs avantages et leurs inconvénients que nous verrons dans une prochaine partie.

Conclusion

Les verrous que nous venons d'aborder dans cette section sont ceux rencontrés dans le cadre de cette thèse. Il existe d'autres verrous comme la gestion des espaces continus ou le transfert d'apprentissage qui ne seront pas abordés dans cette thèse mais le lecteur est invité à consulter les revues de Nguyen *et al.* [177], Arulkumaran *et al.* [10] pour plus de détails. L'approche de ces verrous dépend du cadre formel des problèmes considérés. Nous aborderons ensuite la différence entre les apprenants indépendants et les apprenants d'actions

jointes. Enfin, nous présenterons des techniques existantes attaquant certains de nos verrous dans un contexte différent.

5.4.3 Cadre formel du MARL

Le MARL trouve beaucoup d'applications, notamment dans la représentation de systèmes complexes et d'outils d'aide à la décision dans un cadre réel [181]. Deux catégories sont identifiées pour permettre l'apprentissage multi-agent : d'une part les jeux stochastiques, qui serons brièvement présentés, et d'autre part les applications du monde réel utilisant les Dec-POMDP, abordés dans la section 5.4.3 [183, 53].

Jeux stochastiques

Un jeu stochastique (*Stochastic Game*) [187] ou jeu de Markov (*Markov Games*) [150] s'intéresse à la conception de stratégies permettant à des agents de vaincre leurs adversaires [54]. Il est décrit par le tuple :

$$G = \langle n, S, A, q, \tau, \pi^1, \ldots, \pi^n \rangle$$

où :

- n est le nombre de joueurs (d'agents) ;
- S est l'ensemble des états du jeu ;
- A est l'ensemble fini des actions ;
- π^i est la stratégie de l'agent i ;
- $\tau : \Pi_{i=1}^{n} A^i \to \mathbb{R}^n$ est la fonction de récompense et fait correspondre l'action jointe $\boldsymbol{a} = (a^1, \ldots, a^n)$ à une récompense immédiate pour chaque joueur.

Dans chaque état $s \in S$, chaque joueur i choisi une action a^i à partir de l'ensemble d'actions $A^i(s)$ disponibles dans cet état suivant sa stratégie $\pi^i(s)$.

Un ensemble de travaux porte sur la résolution de jeux coopératifs itérés comme l'algorithme d'apprentissage des actions jointes (*Joint Action Learners* JAL) [50, 12] où l'agent apprend les Q-valeurs des actions jointes plutôt que des actions individuelles. Les algorithmes de Minimax Q-learning [150], Nash Q-learning [110] et Correlated Q-learning [89] apprennent une action jointe pour chaque agent $i \in n$. Ces algorithmes nécessitent de connaitre les actions effectuées par les autres agents et ont donc besoin de modéliser leurs adversaires (*opponent modeling*). Par opposition, les agents peuvent également apprendre sans avoir à modéliser leurs adversaires avec des algorithmes comme *Win or Learn Fast (WoLF)* [33, 31] consistant à s'adapter rapidement quand l'agent perd et lentement quand il gagne. Le regret est utilisé pour modifier sa politique en fonction du choix de l'action [94, 95]. L'algorithme AWESOME (*Adapt When Everybody Is Stationary, Otherwise Move to Equilibrium*) [54] converge contre des adversaires stationnaires. Hansen *et al.* [91] permettent la gestion d'observations locales et individuelles dans les jeux stochastiques partiellement observables *Partially observable stochastic games (POSGs)*. Des états de l'art plus complets sur les jeux stochastiques sont disponibles [193, 36, 5, 219].

Processus de décision markovien partiellement observable et décentralisé

Lorsque le contexte devient décentralisé, un modèle couramment utilisé est le processus de décision markovien décentralisé dans les environnements partiellement observables ou *Decentralized Partially Observable Markov Decision Process* (Dec-POMDP) (figure 5.3). Le Dec-POMDP est plus régulièrement utilisé dans la communauté de la planification où le modèle est connu au début de la tâche de planification. Le Dec-POMDP est un sous cas des POSG que nous choisissons pour la structure factorisée de l'environnement que nous introduisons dans la partie suivante.

Le Dec-POMDP est défini par [23] :

$$\langle \mathcal{D}, \mathcal{S}, \mathcal{A}, T, \mathcal{R}, \Omega, O \rangle$$

où :

- $\mathcal{D} = \{1, \ldots, n\}$ est l'ensemble fini des agents ;
- $\mathcal{A} = \times_{i \in D} A_i$ est l'ensemble des actions jointes où A_i est l'ensemble des actions disponibles pour l'agent i ;
- $\mathcal{S}, T, \mathcal{R}$ sont identiques à la description du MDP (section 5.2.1) ;
- $\Omega = \times_{i \in D} \Omega_i$ est l'ensemble des observations jointes où Ω_i est l'ensemble des observations disponibles pour l'agent i ;
- O est identique à la description du POMDP (ligne 8).

À chaque pas de temps t, chaque agent i applique une action $a_{i,t}$. On obtient donc l'action jointe $\boldsymbol{a}_t = \langle a_{1,t}, \ldots, a_{n,t} \rangle$. Dans un Dec-POMDP, les agents ne disposent pas d'informations concernant les actions effectuées par les autres agents. Par défaut, les agents agissent uniquement à partir de leurs observations courantes (ou de l'historique de leurs observations) et aucune communication n'est envisagée. La définition du Dec-POMDP nous permet uniquement de poser les bases formelles permettant d'introduire le Factored Dec-POMDP [184]. Le lecteur intéressé par ce modèle spécifique peut consulter des ouvrages approfondissant le sujet des Dec-POMDP [70, 181].

Dec-POMDP factorisés

Dans un Dec-POMDP, la fonction de récompense R spécifie uniquement la récompense immédiate pour l'action jointe. Un agent ne sait pas dans quelle mesure sa contribution impacte la récompense obtenue.

Le Factored Dec-POMDP [6, 182] est une extension du Dec-POMDP où les états sont découpés en sous-états (figure 5.4). La particularité de cette approche est de considérer l'état comme un état factorisé, composé de l'ensemble des sous-états de l'environnement. Le factored Dec-POMDP est décrit par le tuple suivant [184] :

$$G = \langle \mathcal{D}, \mathcal{S}, \mathcal{A}, T, \mathcal{O}, O, \mathcal{R}, \gamma \rangle$$

où

- $\mathcal{D} = \{1, \ldots, n\}$ est l'ensemble fini des agents ;

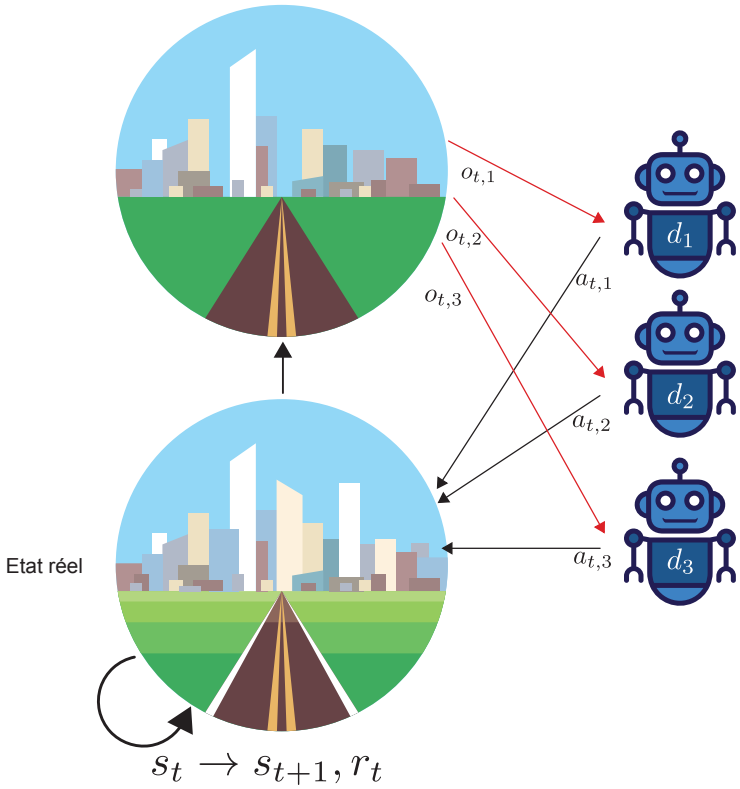


FIGURE 5.3 – Dans le Dec-POMDP, plusieurs agents interagissent en même temps sur l'environnement et reçoivent l'observation jointe, la récompense jointe et appliquent une action jointe.

— $\mathcal{S} = \chi_1 \times \cdots \times \chi_{|\chi|}$ est l'espace d'états factorisé. $\mathcal{S}$ est défini par un ensemble $\chi = \{\chi_1, \ldots, \chi_{|\chi|}\}$ de *variables d'états* ou *facteurs* et un état est une allocation de chaque facteur $s = \langle x_1, \ldots, x_{|\chi|} \rangle$ pour chaque $x_i \in \chi_i$. On note s_0 l'état décrivant les valeurs initiales des facteurs d'états pour tous les états au temps $t = 0$;
— $\mathcal{A} = \times_i \mathcal{A}_i$ est l'ensemble des actions jointes, où $\mathcal{A}_i$ est l'ensemble des actions disponibles pour l'agent i ;
— $T = \mathbb{P}(s'|s, a$ est la fonction de transition spécifiant la probabilité de transition entre les états ;
— $\mathcal{O} = \mathcal{O}_1 \times \ldots \mathcal{O}_n$ est l'ensemble des observations jointes $\boldsymbol{o} = \langle o_1, \ldots, o_n \rangle$ et où $\mathcal{O}_i$ est l'ensemble des observations pour l'agent i ;
— O est la fonction d'observation, spécifiant les probabilités d'avoir une observation jointe $\boldsymbol{o}$ à l'état s' : $\mathbb{P}(\boldsymbol{o}|\boldsymbol{a}, s')$;
— $\mathcal{R} = \{R_1, \ldots, R_n\}$ est l'ensemble des fonctions de récompenses factorisées ;
— $\gamma \in [0, 1)$ est un facteur de réduction.

Si on considère qu'un état est égal à un facteur d'état alors chaque agent a un état *local* avec ses observations et sa récompense locale. La fonction de récompense est unique à chaque agent local.

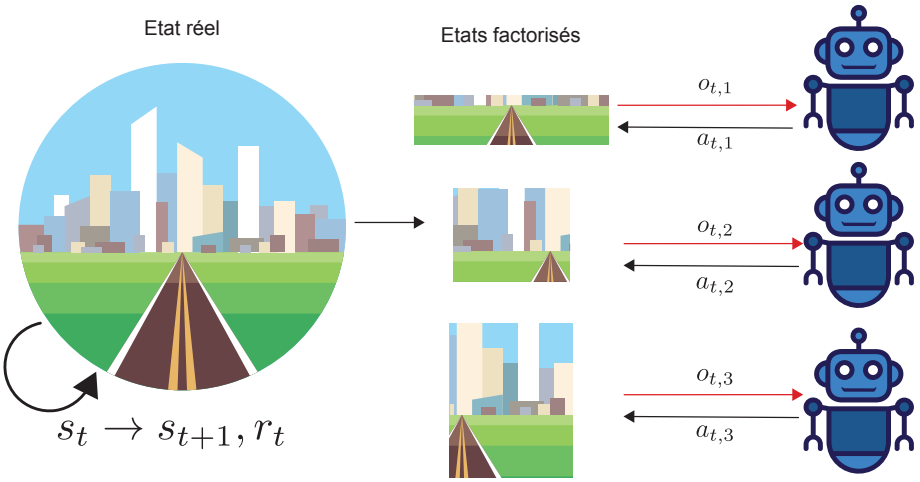


FIGURE 5.4 – Les agents observent un espace factorisé de l'environnement total et les actions qu'ils proposent modifient cet environnement factorisé.

Ce modèle propose une factorisation forte de l'environnement qui convient à notre contexte d'étude.

5.4.4 Différents types d'approches

Dans un cadre d'apprentissage multi-agents, la littérature identifie deux grandes catégories d'agents : les apprenants individuels, abrégé IL pour *Independent Learners* et les apprenants d'actions jointes, JAL les *Joint-Action Learners* [36, 27].

Apprenants indépendants

Les agents indépendants ignorent les autres agents parce qu'ils n'ont pas, par exemple, de moyen de communication adéquat. L'apprentissage multi-agent se réduit donc à un apprentissage mono-agent où l'agent observe un environnement stochastique. Cette approche facilite le passage à l'échelle avec le nombre d'agents, cependant, comme le note Bloembergen *et al.*, les performances vont dépendre du domaine d'application et du choix des algorithmes [27]. Cependant, pour un agent, l'environnement apparait non-stationnaire et cela doit être pris en compte dans l'apprentissage. Des algorithmes comme le Q-learning peuvent être applicables dans des cas simples [36]. Un algorithme populaire dans le cadre du MARL est l'*Independant Q-Learning* de Tan [232] qui compare les résultats d'agents indépendants avec des agents partageant des connaissances dans un problème de capture de proies. Le Distributed Q-learning [140] ainsi que le FMQ (*Frequency Maximum Q Value*) [125] ne requièrent pas de communication entre agents pour converger. Cependant, un problème de ces approches concerne les restrictions fortes sur le fonctionnement de leur environnement, puisqu'ils nécessitent des environnements déterministes. Même impactés par les problèmes de non-stationnarité, les apprenants indépendants obtiennent de bons résultats [160].

Apprenants d'actions jointes et politiques centralisées

Les JAL sont l'opposé des IL dans la mesure où ils considèrent les autres agents lors de l'apprentissage. Introduit par Claus et Boutilier, les JAL apprennent dans l'espace des actions jointes plutôt que dans l'espace des actions individuelles [30]. En observant les actions des autres apprenants, la coordination est améliorée car l'agent agit de manière optimale en fonction des autres apprenants. Les apprenants ont cependant besoin de pouvoir observer les actions des autres agents, ce qui implique de disposer d'un moyen de communication ou de perception des autres agents. Ils doivent également être capables de s'adapter et de prédire les actions futures des autres apprenants. L'apprentissage d'action jointes gère plus facilement les problèmes de coordination et limite la non-stationnarité mais le nombre d'agents dans la simulation impacte davantage ce type d'approche [203]. Par conséquent, cette approche entièrement centralisée passe mal à l'échelle avec le nombre d'agents.

Les approches entièrement collaboratives sont regroupées en fonction des interactions possibles entre les agents. La coordination entre les agents permet d'accéder plus rapidement à l'objectif en agissant ensemble. Certaines approches n'utilisent pas de coordination en émettant des hypothèses sur la manière dont les agents agissent, notamment le *Team Q-learning* [151] qui considère que les actions jointes optimales sont uniques. Dans le cadre de la *coordination indirecte*, les agents définissent un modèle des autres agents [50]. Dans le cadre de la coordination explicite, les agents peuvent négocier les actions à effectuer [247].

Dans les approches hybrides entre coopération et compétition, les agents peuvent coopérer en conservant des objectifs individuels.

5.5 Conclusion

L'apprentissage par renforcement multi-agents apparait comme une solution évidente pour modéliser des systèmes complexes tels que les zones urbaines. L'avantage de l'utilisation d'un apprentissage multi-agents est la possibilité de décentraliser l'apprentissage de politiques et de disposer d'un découpage local des agents plutôt que de devoir prendre en compte l'espace global. Cependant, le recours à l'apprentissage par renforcement multi-agents amène de nouveaux verrous par rapport à l'apprentissage par renforcement mono-agent avec l'introduction d'environnements non-stationnaires et la recherche d'une politique optimale dans un apprentissage concurrent. Nous avons abordé les différentes tâches d'apprentissage existantes ainsi que les types d'agents apprenants existants pour nous inscrire dans un contexte où nos agents apprenants sont indépendants, avec des espaces d'états locaux et des récompenses locales.

L'apprentissage par renforcement profond multi-agents apporte déjà des éléments de réponses à certains verrous de la communauté, notamment avec la fonction d'approximation permettant de considérer des espaces d'états ou d'actions beaucoup plus vastes. Cependant, la réutilisation d'expériences, l'une des clés du succès des approches fondatrices de l'apprentissage par renforcement profond, est impactée par la non-stationnarité induite par l'apprentissage concurrent dans un contexte multi-agents. Bien qu'il existe des méthodes pour palier à la réutilisation d'expériences dans un cadre non-stationnaire, aucun travail à notre connaissance n'est fait sur la réutilisation d'expériences avec des algorithmes de fonction de valeurs sans connaissance du modèle.

Une remarque sur le développement des techniques dans la communauté de l'apprentissage par renforcement multi-agents reproche aux approches de s'intéresser à des scénarios-jouets où le nombre d'agents est faible (généralement 2) [240]. L'objectif de l'utilisation de l'apprentissage multi-agents dans le cadre de cette thèse est de construire un ensemble d'agents apprenants à l'échelle d'une zone urbaine. Il faudrait faire croitre la taille de la zone urbaine sans que les performances au niveau qualité des solutions ne soient impactées.

Afin de permettre un passage à l'échelle simplifié dans un cadre multi-agents, nous envisageons un apprentissage décentralisé avec des agents apprenants indépendants. Dans un cadre de construction de politiques urbaines, il n'est pas toujours possible de connaitre a priori le modèle de l'environnement. Nous nous intéressons donc à la construction d'un modèle permettant d'apprendre des spécificités locales sans connaissance a priori de l'environnement et de son modèle. Les approches décentralisées semblent être des hypothèses valables pour représenter des systèmes complexes avec des capteurs dispersés dans une zone urbaine par exemple. L'apprentissage multi-agents est donc envisagé comme une solution pour doter notre outil d'aide à la décision SmartGov d'une couche auto-adaptative pour assister le décideur dans la construction de politiques urbaines. Le prochain chapitre présente un algorithme avec plusieurs objectifs : d'une part proposer de répondre aux verrous de la réutilisation d'expériences dans un cadre non-stationnaire et également de pouvoir identifier la contribution individuelle de chaque agent dans un contexte sans communications et d'autre part de doter SmartGov d'un module d'apprentissage pour la couche macroscopique.

Chapitre 6

Une nouvelle méthode d'apprentissage multi-agents pour la coordination d'agents sans communication

Sommaire

6.1 Introduction

L'étude d'exemple réels pour la simulation nécessite de prendre en compte des espaces d'états ou d'actions conséquents. Ainsi, un environnement réaliste aura un nombre d'états croissant en fonction de la description plus ou moins complexe de celui-ci. Les précédentes approches d'apprentissage par renforcement, utilisant des représentations tabulaires des couples états-actions, sont donc limitées. Dans le cadre de la conception d'une politique urbaine, nous souhaitons simplifier la tâche du décideur pour l'exploration de politiques pertinentes parmi l'ensemble des politiques possibles. L'utilisation d'apprentissage par renforcement permet de traduire ses objectifs en une fonction de récompense exploitable par les agents pour produire une politique urbaine adaptée. Ce chapitre introduit la seconde contribution de cette thèse où nous proposons une architecture multi-agents, multi-niveaux, appelée CDQN (*Clustered Deep Q-Network*) permettant un contrôle autonome et adaptatif d'un ensemble d'agents à l'aide d'apprentissage par renforcement multi-agents profond.

Avant l'introduction de l'apprentissage par renforcement profond, les approches d'apprentissage par renforcement multi-agents reposaient en partie sur une représentation tabulaire des couples états-actions $Q(s, a)$. L'une des limites des méthodes tabulaires dans le cadre de l'apprentissage est la taille de l'espace d'états ou d'actions qu'il est possible de considérer. L'apport de l'apprentissage par renforcement profond, avec l'utilisation d'une fonction d'approximation, permet désormais de considérer des espaces d'états et d'actions bien plus conséquents. Dans le cadre de ce chapitre, nous nous intéressons à l'utilisation de l'apprentissage par renforcement profond dans un cadre multi-agents et aux problématiques identifiées par la communauté. En effet, une des difficultés liée à l'apprentissage par renforcement multi-agents est celui de l'apprentissage dans un cadre non-stationnaire, considéré comme un problème clé par la communauté [38, 141]. La non-stationnarité est due à la perte de l'environnement de sa propriété markovienne aux yeux d'un agent. La fonction de transition et de récompenses évoluent avec le temps en même temps que les agents apprennent et leur choix d'actions évoluent. Le chapitre précédent introduit les verrous auxquels la communauté de l'apprentissage par renforcement multi-agents est confronté et plus particulièrement les verrous suivants :

— **Difficulté d'apprendre sans communication :** dans la mesure où les autres agents évoluent également, apprendre en présence d'autres agents apprenants complexifie l'identification de la contribution de l'agent sur les performances du système et la récompense qu'il doit prendre en compte pour l'apprentissage [46] ;

— **Réutilisation d'expériences passées obsolètes :** dans un environnement non-stationnaire, les transitions $\langle s, a, r, s' \rangle$ stockées sont faussées car la transition qui était valable au temps t ne sera plus valable au temps $t + \Delta t$. Par conséquent, rejouer une partie de la mémoire de travail pour alimenter le réseau de neurones pose problème.

— **Problème de la cible mouvante :** Les techniques d'apprentissage de politiques décentralisées doivent prendre en compte la possibilité d'avoir la politique optimale de chaque agent changée pendant l'apprentissage [32].

Notre objectif avec cette seconde contribution est de proposer des méthodes de résolutions à ces problèmes clés identifiés par la communauté notamment sur la réutilisation d'expériences dans un environnement non-stationnaire pour l'apprentissage par renforcement profond, ainsi que la possibilité de passer à l'échelle

avec des agents indépendants capable de se coordonner sans communications.

Dans une première partie, ce chapitre présentera les problématiques et motivations amenant à l'architecture CDQN. Puis, nous aborderons le formalisme associé à notre approche. Ensuite, nous nous intéresserons en détail au fonctionnement de chaque niveau de notre approche et de leur interaction. Enfin, le couplage entre CDQN et SmartGov sera introduit.

6.2 Clustered Deep Q-Network (CDQN)

Nous proposons le modèle Clustered Deep Q-Network (CDQN) comme une contribution à l'apprentissage par renforcement multi-agents profond. Le CDQN, composé d'une population multi-agents et multi-niveaux, propose de s'intéresser aux problématiques mentionnées dans la section précédente. D'une part en disposant d'agents apprenants appliquant une action jointe contrainte pour limiter l'impact de la non-stationnarité pendant l'apprentissage. D'autre part avec l'utilisation des scores de confiance et des récompenses locales pour coordonner des actions dans un environnement partiellement observable sans aucune communication (section 6.2). Le CDQN utilise une hiérarchie d'agents tel que :

— le premier niveau possède des agents apprenants ayant une perception locale de leur environnement ;
— le niveau supérieur dispose d'agents de contrôle manipulant des clusters d'agents apprenants.

Chaque agent apprenant possède son propre Deep Q-Network pour avoir un apprentissage individuel.

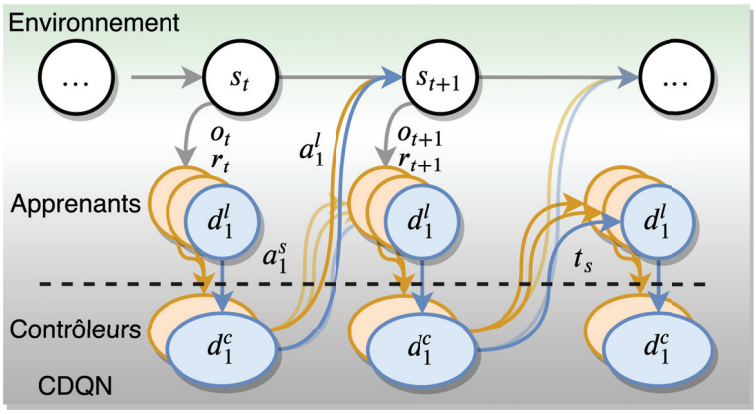


FIGURE 6.1 – Les trois agents apprenants locaux (deux apprenants orange et un apprenant bleu) sont séparés en deux clusters (orange et bleu). Un agent apprenant local d_1^l reçoit une observation o_t de l'environnement et suggère une action a_1^s à son agent de contrôle d_1^c. À l'aide d'un système de vote, un agent de contrôle d_1^c choisit l'action a_1^l que l'apprenant d_1^l applique sur l'environnement. À l'itération suivante, d_1^l reçoit une nouvelle observation o_{t+1} ainsi qu'une récompense r_{t+1}.

Cette section introduit dans un premier temps le modèle formel du CDQN avant de s'intéresser au détail de chaque composante de l'architecture.

Soumissions L'architecture formelle Clustered Deep Q-Network, présentée dans ce chapitre, a donné lieu aux publications :

— d'un résumé étendu (*Extended Abstract* à l'*Autonomous Agents and Multi-Agent Systems* (AAMAS), présenté avec un poster en 2019 à Montréal au Canada [192] ;

— d'un article à la Conférence Nationale sur les *Applications Pratiques de l'Intelligence Artificielle* (APIA), présenté en 2019 à Toulouse en France [191].

6.2.1 Modèle formel du CDQN

Le modèle formel du CDQN est une extension du *factored Dec-POMDP* [6, 182]. Le CDQN est décrit par le tuple suivant :

$$\langle \hat{\mathcal{D}}, \hat{\mathcal{S}}, \hat{\mathcal{A}}, T_t, \hat{\mathcal{O}}, O, \mathcal{R}, \tau, \gamma \rangle$$

où :

— $\hat{\mathcal{D}} = \mathcal{D}^l \cup \mathcal{D}^c$ est l'ensemble fini des agents avec $\mathcal{D}^l$ l'ensemble fini des agents apprenants et $\mathcal{D}^c$ l'ensemble fini des agents de contrôle. $|\mathcal{D}^l| = M \in \mathbb{N}^{+*}$ est le nombre d'agents locaux. $|\hat{\mathcal{D}}|$ est borné par $|\mathcal{D}^l| + 1$ dû au nombre minimal de cluster de 1 et par $|\mathcal{D}^l| \times 2$ dans le cas où il existe un cluster par agent apprenant ;

— $\hat{\mathcal{S}} = \mathcal{S}^l \cup \mathcal{S}^c$ est l'ensemble des états de l'environnement où $\mathcal{S}^l$ est l'ensemble fini des espaces factorisés des apprenants et $\mathcal{S}^c$ est l'ensemble fini des états des agents de contrôle ;

— $\hat{\mathcal{A}} = \mathcal{A}^l \cup \mathcal{A}^c$ est l'ensemble fini des actions jointes avec $\mathcal{A}^l$ l'ensemble fini des actions disponibles pour les agents apprenants et $\mathcal{A}^c$ l'ensemble fini des actions de contrôle ;

— $T_t(s, a, s') = \mathbb{P}(s'|s, a)$ est la fonction de transition non-stationnaire spécifiant la probabilité de transition entre états s et s' à l'instant t ;

— $\hat{\mathcal{O}} = \mathcal{O}^l \cup \mathcal{O}^c$ est l'ensemble fini des observations jointes où $\mathcal{O}^l$ est l'ensemble fini des observations jointes des agents apprenants et $\mathcal{O}^c$ est l'ensemble fini des observations jointes des agents de contrôle ;

— $O : \mathcal{O}^l \times \mathcal{A}^l \times \mathcal{S}^l \to [0, 1]$ est la fonction d'observation, spécifiant la probabilité $\mathbb{P}(o|\boldsymbol{a}, s)$ de recevoir l'observation o dans l'état s avec l'action jointe $\boldsymbol{a}$;

— $\mathcal{R} = \{R_1, \ldots, R_N\}$ est l'ensemble des fonctions de récompenses et R_i est la fonction de récompense de l'agent apprenant d_i^l ;

— $\tau : \mathcal{A}^l \times \mathcal{A}^l \times \mathcal{F}(\mathcal{S}^l) \times \mathcal{F}(\mathcal{S}^l)^n \to \{-\epsilon, 0, \epsilon\}$ est la fonction de score de confiance, où $\mathcal{F}$ est la fonction objectif utilisée dans le CDQN ;

— $\gamma \in [0, 1)$ est un facteur d'atténuation.

Pour expliquer en détail les éléments du modèle formel, chaque couche sera d'abord explicitée avant de voir la jointure de ces deux couches.

6.2.2 Objectif du CDQN

L'objectif du CDQN est de trouver la politique jointe $\boldsymbol{\pi} = \langle \pi_1, \ldots, \pi_n \rangle$ maximisant la récompense cumulée espérée. Chaque population d'agents dispose de ses propres objectifs. L'objectif d'un agent apprenant i est

d'avoir une politique π_i^l spécifiant l'action à effectuer pour chaque couple action/observation $\theta_i = (a_i, o_i)$ de l'environnement tel que $\pi_i^l(\theta_i) = a_i$. L'objectif d'un agent de contrôle j est de maximiser d'une part son gain cumulé et d'autre part de maximiser le gain global cumulé. Nous verrons dans la section 6.2.7 que l'agent de contrôle peut choisir une configuration baissant son gain agrégé si cela augmente les performances globales. Il y a donc une différence entre l'agent apprenant qui suit une politique individuelle et l'agent de contrôle qui suit une politique collective. Cela se traduit par l'obtention de la meilleure répartition d'apprenants dans son cluster. Cette répartition se fait en fusionnant ou en séparant une partie ou la totalité de leurs apprenants. Par conséquent, l'un des objectifs de la simulation est de proposer la meilleure répartition en clusters des agents locaux. Nous considérons donc une architecture où le nombre d'agents de contrôle et les agents locaux qui leur sont affectés peuvent évoluer pendant la simulation.

6.2.3 Approche hiérarchique des agents

Notre approche repose sur deux populations d'agents décentralisés et hiérarchisés : les **agents apprenants locaux** et les **agents de contrôle**.

Agents apprenants locaux

L'ensemble fini des agents apprenants locaux est défini par $\mathcal{D}^l = (d_1^l, \dots d_M^l)$ où d_i^l est un *apprenant local* ou plus simplement *apprenant*. Les apprenants n'ont aucune connaissance sur l'existence d'autres apprenants et ne peuvent donc pas communiquer avec eux. Le nombre d'apprenants M est immuable au cours de la simulation.

Chaque agent dispose d'un espace factorisé tel que $\mathcal{S}_i^l = \chi_{1,i} \times \cdots \times \chi_{|\chi_i|,i}$ où $\mathcal{S}_i^l$ est recouvert par un ensemble $\chi_i = \{\chi_{1,i}, \dots, \chi_{|\chi_i|,i}\}$ de variables d'états ou facteurs. Un état $s_i^l \in \mathcal{S}_i^l$ pour l'apprenant d_i^l est une attribution de tous ces facteurs tel que $s_i^l = \langle x_{1,i}, \dots, x_{|\chi_i|,i}\rangle$. L'espace d'états factorisés des agents locaux est donc défini comme l'espace factorisé joint de chaque agent, tel que $\mathcal{S}^l = \mathcal{S}_1^l, \dots, \mathcal{S}_M^l$. Un état global est donc $s^l = \langle s_1^l, \dots, s_M^l\rangle$.

Considérons par exemple deux apprenants d_1^l et d_2^l. d_1^l possède un espace factorisé avec trois facteurs $\chi_1 = \{\chi_{1,1}, \chi_{1,2}, \chi_{1,3}\}$ avec :

— $\chi_{1,1} = \{10, 15, 20, 25\}$ la température de la pièce ;
— $\chi_{1,2} = \{1, 2, 3, 4\}$ la luminosité de la pièce ;
— $\chi_{1,3} = \{0, 1\}$ l'état d'un interrupteur.

Un état de l'espace factorisé de d_1^l peut être par exemple $s_1^l = \{10, 4, 1\}$. L'apprenant d_2^l possède un espace factorisé avec deux facteurs $\chi_2 = \{\chi_{2,1}, \chi_{2,2}\}$ où :

— $\chi_{2,1} = \{10, 15, 20, 25\}$ est la température ;
— $\chi_{2,2} = \{0, 1\}$ l'état d'un interrupteur.

À un instant t, un état global possible du système est $s^l(t) = \langle\{10, 4, 1\}, \{20, 1\}\rangle$.

Avec l'exemple précédent, nous observons que les deux espaces factorisés disposent de deux facteurs d'états communs : la température de la pièce ($\chi_{1,1}$ de l'espace factorisé χ_1 de l'apprenant d_1^l et $\chi_{2,1}$ de l'espace factorisé χ_2 de l'apprenant d_2^l) et l'état d'un interrupteur ($\chi_{1,3}$ et $\chi_{2,2}$). En fonction de la description du

problème, les espaces factorisés peuvent être similaires si pour deux espaces factorisés distincts, ils possèdent un facteur d'état commun où la valeur est égale. Par exemple, si l'état de l'espace factorisé de d_1^l est $s_1^l = \{10, 4, 1\}$ et l'espace factorisé de d_2^l est $s_2^l = \{10, 0\}$ alors le facteur d'état de la température est égal dans les deux espaces factorisés. Nous notons la similarité entre deux états comme $s_1^l \equiv s_2^l$ si pour deux valeurs $x, y \in \mathbb{R}$ $x \in \chi_{1,1} = y \in \chi_{2,1}$. Nous introduisons maintenant $\mathcal{S}_\alpha^l$ comme l'ensemble fini des états globaux initiaux similaires.

Choix de l'action chez l'apprenant L'ensemble fini des actions jointes des apprenants est défini par $\mathcal{A}^l = \mathcal{A}_i^l \times \cdots \times \mathcal{A}_M^l$ où $\mathcal{A}_i^l$ est l'ensemble des actions disponibles pour l'apprenant d_i^l. Ces actions sont dépendantes du scénario et sont appelées *actions environnementales*. Pendant l'apprentissage, l'apprenant d_i^l explore avec un algorithme ϵ-*greedy* en suivant sa politique telle que :

$$\pi_i(s) = \begin{cases} \text{argmax}_{a_i^l \in \mathcal{A}_i^l} Q_i(s, a^l) \text{ avec une probabilité de } 1 - \epsilon & \text{(6.1a)} \\ \mathcal{U}(\mathcal{A}_i^l) \text{ avec une probabilité de } \epsilon & \text{(6.1b)} \end{cases}$$

où $\mathcal{U}(\mathcal{A}_i^l)$ est une sélection sur une distribution uniforme sur $\mathcal{A}_i^l$. Par la suite, chaque apprenant met à jour sa politique à partir des transitions de sa mémoire de travail tel que :

$$Q_i(s, a_i^l) \leftarrow Q_i(s, a_i^l) + \alpha\big[r_i + \gamma \max_{a_i'^l \in \mathcal{A}_i^l} Q_i(s', a_i'^l) - Q_i(s, a_i^l)\big]$$

Agents de contrôle

L'ensemble des *agents de contrôle* ou *clusters* est défini par $\mathcal{D}^c = (d_1^c, \ldots, d_N^c)$ où l'agent de contrôle $d_j^c, j \in [1, N]$, $N \leq M$ gère un ensemble d'apprenants. Nous notons η le nombre d'apprenants dans un cluster.

À chaque itération, un apprenant appartient toujours à un agent de contrôle. Par conséquent, le nombre possible de configurations pour les agents de contrôle est le nombre de manières dont l'ensemble des apprenants peut être partitionné en sous-ensembles non-vides et disjoints (figure 6.2). Le nombre de configurations est décrit par les nombres de Bell [1] et l'ensemble des configurations possibles des clusters est notée $\mathcal{C}$.

L'ensemble fini des actions jointes des agents de contrôle est défini par $\mathcal{C} = \times_j \mathcal{A}_j^c$ où $\mathcal{A}_j^c$ est l'ensemble des actions disponibles pour l'agent de contrôle d_j^c. Ces actions, appelées *actions de contrôle*, permettent de modifier la distribution des apprenants dans les clusters et sont indépendantes du scénario.

Les agents de contrôle ont la possibilité de communiquer entre eux lors de la simulation pour permettre d'évaluer leurs actions de contrôle que nous verrons dans la section 6.2.7.

Observations de l'environnement

Un apprenant d_i^l observe son état factorisé o_i^l avec une probabilité $O(o_i^l, s_i^l, i)$ à chaque itération. On note $\mathcal{O}^l = \mathcal{O}_1^l \times \cdots \times \mathcal{O}_M^l$ l'ensemble fini des observations jointes des apprenants où $\mathcal{O}_i^l$ est l'ensemble fini des

1. Le nombre de Bell est le nombre de partitions d'un ensemble à n éléments distincts.

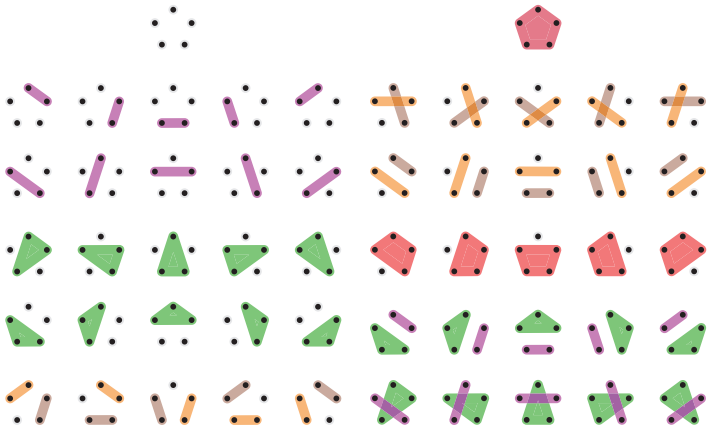


FIGURE 6.2 – Représentation du nombre de partitions pour 5 apprenants locaux. Pour 5 apprenants locaux, le nombre de partitions est de 52. Chaque groupe de couleur représente un cluster ou agent de contrôle. Chaque point noir n'appartenant pas à un groupe de couleur représente également un cluster avec un seul individu.

observations de l'apprenant d_i^l et $\boldsymbol{o}^l = \langle o_1^l, \ldots, o_M^l \rangle$ une observation jointe.

L'ensemble fini des observations jointes des agents de contrôle, noté $\mathcal{O}^c = \mathcal{O}_1^c \times \cdots \times \mathcal{O}_m^c$, où $\mathcal{O}_j^c$ est l'ensemble des observations de l'agent de contrôle d_j^c et $\boldsymbol{o}^c = \langle o_1^c, \ldots, o_N^c \rangle$ une observation jointe. Les agents de contrôle n'observent pas directement l'environnement et n'ont aucune information sur l'état factorisé des apprenants composant les clusters. Cependant, un agent de contrôle d_j^c utilise les informations transmises par les apprenants de son cluster comme perception, noté o_j^c.

6.2.4 Application des actions environnementales

À chaque pas de temps, un apprenant d_i^l suggère une action $a_i^s \in \mathcal{A}_i^l$ à son agent de contrôle. Les actions suggérées par les agents apprenants d'un cluster j sont notées $\boldsymbol{a}_j^s = \langle a_i^s, \ldots, a_\eta^s \rangle$, l'*action jointe suggérée* des agents apprenants à leur cluster j.

L'agent de contrôle j utilise un système de vote pour sélectionner l'*action choisie* $a_j^a \in \boldsymbol{a}_j^s$ à partir de l'action jointe suggérée $\boldsymbol{a}_j^s$. Actuellement, le système de vote sélectionne l'action proposée en majorité. En cas d'égalité, elle est arbitrairement sélectionnée parmi celles-ci. L'action ainsi choisie a_j^a est appliquée par chaque apprenant du cluster sur leur état factorisé formant ainsi l'*action jointe appliquée* $\boldsymbol{a}_j^a = \langle a_1^a, \ldots, a_\eta^a \rangle$, où :

$$a_1^a = a_2^a = \ldots = a_\rho^a = \boldsymbol{a}_j^a$$

La première contribution de notre modèle CDQN est l'action unique imposée par l'agent de contrôle. À n'importe quelle itération, tous les apprenants d'un même cluster effectuent la même action et ont donc la même transition dans leur mémoire. Cela a pour effet de réduire la non-stationnarité de l'environnement et d'amoindrir son impact sur la réutilisation d'expériences pendant l'apprentissage. L'ensemble des actions, disponibles pour l'agent de contrôle, permettant de modifier la configuration des clusters, sera décrit plus

tard dans la section section 6.2.7.

6.2.5 Attribution de la récompense

Dans le CDQN, nous faisons une différence entre la fonction objectif et la fonction de récompense.

Fonction objectif La fonction objectif $\mathcal{F}: \mathcal{S}^l \to \mathbb{R}$ est propre au scénario. L'apprenant d_i^l l'utilise pour déterminer le gain g_i^l d'un état de son espace factorisé $s_i^l \in \mathcal{S}^l$. Chaque apprenant dispose de la même fonction objectif, cependant le gain obtenu est unique à l'apprenant en fonction de son état s_i^l factorisé. L'évolution du gain est utilisée pour déterminer la pertinence d'effectuer une action par rapport à une autre.

L'apprenant d_i^l conserve l'information sur les gains suivants :

— gb_i^l : la plus haute valeur de gain obtenue pendant l'exploration ;
— $g_{i,k}^l = (g_i^l(t-k), \ldots, g_i^l(t-1))$: un vecteur contenant les valeurs du gain des k précédentes itérations ;
— $g_i^l(t)$: la valeur du gain à l'itération courante.

Le gain est indirectement utilisé dans l'apprentissage et permet aux agents d'avoir une représentation pertinente de leur environnement. Il est donc nécessaire d'identifier la fonction objectif correcte pour que les agents soient en mesure de proposer une politique pertinente.

Fonction de récompense La fonction de récompense $\mathcal{R} : \mathcal{F}(\mathcal{S}^l) \times \mathcal{F}(\mathcal{S}^l) \times \mathcal{F}(\mathcal{S}^l)^k \to \{-1, 1\}$, est propre au CDQN et évalue l'évolution relative du gain entre deux états factorisés après avoir appliqué l'action a. Cette évolution peut être la conséquence d'une action particulière d'un autre apprenant ou d'un évènement extérieur. Afin d'éviter le problème de l'identification de la contribution des agents (*multi-agent credit assignment problem*), nous utilisons l'attribution d'une récompense individuelle. De plus, les agents de contrôle préviennent les comportements individuels grâce à la différence entre les actions suggérées et les actions appliquées. Ainsi, chaque apprenant dispose d'une fonction de récompense identique mais l'applique sur ses propres variables. Cette approche s'inspire des travaux sur les récompenses différentes [2] (*difference rewards*) où chaque agent apprend à partir d'une récompense individuelle plutôt que d'une récompense globale. Il est ainsi plus simple d'évaluer la contribution de chaque agent dans les performances du système. Une récompense de -1 ou 1 s'inspire des travaux sur les récompenses tronquées [243] (*clipped rewards*). L'utilisation de cette fonction de récompense présente plusieurs avantages :

1. une représentation générique de l'attribution de la récompense, quel que soit la fonction objectif ;
2. une meilleure évaluation de la contribution de chaque agent.

Pour un apprenant d_i^l, la fonction de récompense utilise son gain courant $g_i^l(t) \in \mathbb{R}$, son meilleur gain obtenu $gb_i^l \in \mathbb{R}$ et les k précédents gains $g_{i,k}^l \in \mathbb{R}^k$. La récompense est attribuée de la manière suivante :

$$r(g_i^l(t), gb_i^l, g_{i,k}^l) = \begin{cases} 1 \text{ si } g_i^l(t) > g_{i,k}^l & \text{(6.2a)} \\ 1 \text{ si } g_i^l(t) = g_{i,k}^l \text{ et } g_i^l(t) = gb_i^l & \text{(6.2b)} \\ -1 \text{ si } g_i^l(t) \leq g_{i,k}^l \text{ et } g_i^l(t) < gb_i^l & \text{(6.2c)} \end{cases}$$

Les équations ci-dessus comparent une valeur de gain $g_i^l(t)$ avec un vecteur de valeurs $g_{i,k}^l$. Dans le cas où $k = 1$, il s'agit d'observer l'évolution du gain courant $g_i^l(t)$ par rapport au gain précédent $g_i^l(t-1)$. Si k est supérieur à 1, il existe différentes méthodes pour effectuer la comparaison entre la valeur à l'état courant et le vecteur de valeurs : prendre en compte la valeur maximale du vecteur sur une fenêtre glissante, la valeur moyenne, la valeur avec le plus grand nombre d'apparitions.

6.2.6 Score de confiance

Nous introduisons un score de confiance $\tau \in [-1, 1]$, utilisé par les agents de contrôle, pour évaluer la contribution de chaque apprenant au sein du cluster suite à l'*action jointe suggérée* $\boldsymbol{a}_j^s$. L'agent de contrôle j reçoit de son cluster le gain agrégé $g_j^c(t) \in \mathbb{R}$ correspondant aux gains obtenus par l'ensemble des apprenants du cluster, agrégés avec une moyenne ou une somme par exemple. Il utilise la fonction de confiance τ, définie par $\tau : \mathcal{A}^l \times \mathcal{A}^l \times \mathcal{F}(\mathcal{S}^l) \times \mathcal{F}(\mathcal{S}^l)^n \to \{-\mu, 0, \mu\}$, pour mettre à jour le score de confiance de l'apprenant i à partir de l'action suggérée $a_i^s \in \mathcal{A}^l$, de l'action appliquée $a_j^a \in \mathcal{A}^l$, du gain agrégé aux k précédentes itérations $g_{j,k}^c \in \mathcal{F}(\mathcal{S}^l)^n$ et du gain agrégé à l'itération actuelle $g_j^c(t) \in \mathcal{F}(\mathcal{S}^l)$. Le gain agrégé d'un agent de contrôle correspond à l'agrégation du gain de chaque apprenant de son cluster. Il peut donc être égal à la somme des gains de chaque apprenants par exemple.

L'attribution du score de confiance aux apprenants suit les règles suivantes :

$$\tau(a_i^s, a_j^a, g_j^c(t), g_{j,k}^c) = \begin{cases} 0 \text{ si l'action proposée est choisie aléatoirement} & \text{(6.3a)} \\ 0 \text{ si } a_i^s \neq a_j^a \text{ et } g_j^c(t) < g_{j,k}^c & \text{(6.3b)} \\ \mu \text{ si } a_i^s = a_j^a \text{ et } g_j^c(t) \geq g_{j,k}^c & \text{(6.3c)} \\ -\mu \text{ si } a_i^s = a_j^a \text{ et } g_j^c(t) < g_{j,k}^c & \text{(6.3d)} \\ -\mu \text{ si } a_i^s \neq a_j^a \text{ et } g_j^c(t) \geq g_{j,k}^c & \text{(6.3e)} \end{cases}$$

L'apprenant voit donc son score de confiance augmenté si l'action qu'il propose est sélectionnée par l'agent de contrôle et que les performances évoluent. Au contraire, son score de confiance décroît si l'action qu'il propose est sélectionnée et que les performances diminuent.

Lors de la mise-à-jour du score de confiance, si l'incrément devrait augmenter le score au-dessus de 1 (respectivement en dessous de -1), le score de confiance sera fixé à 1 (respectivement à -1).

Identification des actions similaires L'incrément 6.3e conduit au résultat souhaité (pénalisation des agents n'ayant pas proposé la meilleure action) s'il existe une unique action permettant d'avoir un gain agrégé supérieur ou égal à partir de l'état précédent. Cela signifie que si le gain agrégé s'améliore mais qu'un apprenant suggère une action a_i^s différente de l'action appliquée a_j^a alors son score de confiance sera pénalisé. Cependant, un agent peut être pénalisé à tort dans ces deux cas :

1. deux actions pourraient augmenter le gain de façons différentes, et l'action a_i^s pourrait amener à un meilleur gain agrégé mais ne pas avoir été choisie. Il s'agit dans ce cas d'une exploration incomplète et l'agent apprenant d_i^l est pénalisé à tort ;

2. deux actions différentes ont le même effet sur les performances.

Afin de gérer le problème du second cas, nous introduisons une manière d'identifier les *actions similaires*. Nous considérons comme similaires deux actions différentes si, dans un même état $s \in \mathcal{S}^{\updownarrow}$, elles ont exactement le même impact sur le gain agrégé d'un agent de contrôle. L'agent de contrôle utilise la transition entre deux états qui produisent le meilleur gain agrégé comme information. Ces transitions fournissent des informations sur le gain agrégé à l'état précédent $g_j^c(t-1)$ et à l'état courant $g_j^c(t)$ à partir des observations de l'agent de contrôle. Une transition est décrite par :

$$tr = (g_j^c(t-1), a_j^a(t-1), g_j^c(t))$$

Deux actions sont considérées *similaires* si pour deux transitions $tr_1 = (g_{j,1}^c(t-1), a_{j,1}^a(t-1), g_{j,1}^c(t)$ et $tr_2 = (g_{j,2}^c(t-1), a_{j,2}^a(t-1), g_{j,2}^c(t)$:

$$a_{j,1}^a(t-1) \neq a_{j,2}^a(t-1) \text{ et } g_{j,1}^c(t) = g_{j,2}^c(t)$$

et que :

$$g_{j,1}^c(t) = gb_j^c$$

où gb_j^c est le meilleur gain agrégé atteint par l'agent de contrôle d_j^c. Si deux transitions sont *similaires*, alors nous notons la relation entre les deux actions par $a_{j,1}^a(t-1) \equiv a_{j,2}^a(t-1)$. Avec l'identification des actions similaires, nous mettons à jour la règle d'incrément du score de confiance de la façon suivante :

$$\tau(a_i^s, a_j^a, g_j^c(t), g_{j,k}^c) = \mu \text{ si } a_i^s \equiv a_i^a \text{ et } g_j^c(t) = gb_j^c \ -\mu \text{ sinon} \tag{6.4}$$

qui signifie que le score de confiance augmente si l'apprenant d_i^l suggère une action a_i^s différente de celle sélectionnée a_i^a mais que les deux sont considérées équivalents et que le gain agrégé obtenu est égal au meilleur gain agrégé obtenu par l'agent de contrôle. Son score de confiance est diminué dans le cas où l'action suggérée n'est pas similaire à l'action appliquée.

Convergence du score de confiance Seuls les incréments 6.3a et 6.3b ne modifient pas le score de confiance. Par conséquent, au cours de la phase d'apprentissage, le score de confiance des apprenants va tendre vers 1 ou -1 pour un cluster donné. Suggérer une action (non aléatoire) pour le cluster signifie généralement que l'apprenant, au niveau individuel, obtient une récompense positive si cette action est appliquée car l'apprenant obtient pendant l'exploration une récompense positive. Pour que le score de confiance augmente, le gain agrégé de l'agent de contrôle doit s'améliorer, ou atteindre le meilleur gain agrégé précédemment obtenu. Cependant, un apprenant peut avoir une récompense positive alors que l'agent de contrôle voit son gain agrégé diminué. C'est pour cette raison qu'il existe des apprenant suggérant des action non pertinente pour le cluster. Les apprenants continueront de proposer des actions qui vont baisser leur score de confiance car eux souhaitent avant tout maximiser leurs récompenses locales. Le but de l'agent de contrôle est désormais de prendre en compte les disparités entre les propositions d'actions pour séparer les apprenants en clusters

pertinents.

6.2.7 Actions de l'agent de contrôle

Afin de satisfaire son objectif, l'agent de contrôle doit sélectionner les apprenants ayant le meilleur impact sur son gain agrégé. Deux actions sont disponibles pour lui permettre d'ajuster les apprenants de son cluster : la séparation et la fusion.

La séparation a pour but de permettre à des agents ayant des scores de confiance négatifs d'augmenter leur gain local en appliquant des actions plus appropriées (donc augmentation du gain global). Ils seront regroupés dans un nouveau cluster et pourront donc y choisir des actions différentes de celles majoritaires dans leur cluster initial.

La fusion diminue le nombre de clusters, mais seulement si le gain agrégé est au moins aussi bon. La minimisation du nombre de clusters est un objectif secondaire du CDQN. Tout en maintenant ou améliorant la cohérence des clusters, on cherche à minimiser leur nombre de manière à réduire l'impact de la non-stationnarité dans une configuration d'*Independent Q-Learning*. Un plus faible nombre de clusters réduit l'impact d'un environnement non stationnaire à l'aide de l'action jointe simplifiée $\boldsymbol{a}_j^a$ car chaque apprenant d'un même cluster effectue la même action environnementale.

Plus la répartition des apprenants dans les clusters est pertinente, plus le gain agrégé global est important.

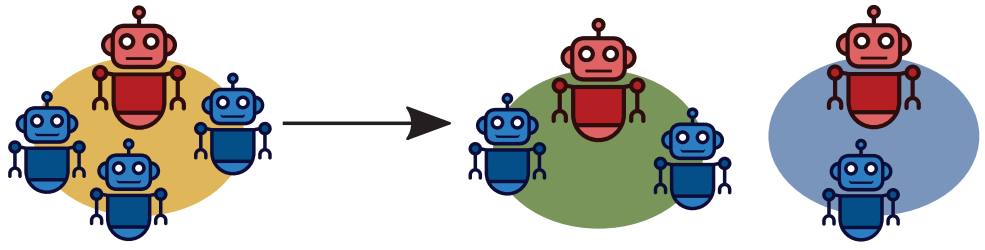
Les actions disponibles pour l'agent de contrôle d_j^c sont définies par :

$$\mathcal{A}_j^c = \{\texttt{Fusion},\ \texttt{Separation},\ \texttt{Pas\_de\_Modification}\}$$

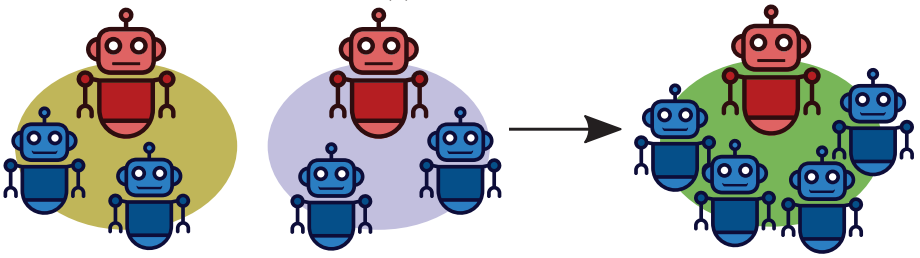
Chaque action de contrôle $a^c \in \mathcal{A}^c$ est une action ayant des conséquences sur l'organisation des clusters, à l'opposé des actions environnementales $a^a \in \mathcal{A}^l$ qui modifient l'environnement (figure 6.3).

Quand une action `Fusion` ou une action `Separation` est appliquée, le système enregistre la configuration actuelle avant d'appliquer l'action de contrôle. Le score de confiance de chaque apprenant des clusters concernés par l'action est remis à 0 pour évaluer la contribution des apprenants dans les clusters modifiés ou créés. Cela permet de considérer de manière égale les nouveaux apprenants et les anciens dans le choix de l'action (prise en compte du score de confiance par exemple).

Les agents de contrôle disposent de règles pour déterminer à quel moment appliquer ces actions ainsi que des critères pour évaluer la pertinence de cette configuration de clusters sur le gain global agrégé. Quand une action `Fusion` ou une action `Separation` est appliquée, les agents de contrôle ont plusieurs itérations pour évaluer l'efficacité de leurs décisions. Pendant la période d'évaluation, les agents de contrôle qui évaluent leurs clusters ne peuvent effectuer que l'action `Pas_de_Modification`. Les agents de contrôle qui ne réalisent pas d'évaluation à ce moment applique également l'action `Pas_de_Modification` afin de simplifier la gestion des clusters pendant l'évaluation. Par exemple, si un agent de contrôle applique une action de contrôle alors qu'un autre cluster est évalué, l'évaluation sera faussée car ses performances peuvent varier de manière importante. La raison de ces variations est qu'une action de contrôle modifie la stationnarité de l'environnement. L'évaluation étant effectuée en considérant un environnement suffisamment stationnaire pour évaluer la pertinence de



(a) `Separation`



(b) `Fusion`

FIGURE 6.3 – La séparation d'un agent de contrôle d_1^c représente la scission de son cluster d'agents locaux en deux agents de contrôle : il conserve une partie de son cluster et un nouvel agent de contrôle d_2^c récupère le reste des agents locaux (6.3a). La fusion représente la combinaison de deux agents de contrôle en un seul qui prend le contrôle des agents locaux de chaque cluster (6.3b).

l'action politique, il est donc préférable d'interdire d'autres actions de contrôle en même temps.

Une fois l'évaluation effectuée, les agents de contrôle concernés choisissent de conserver cette configuration ou de revenir à la configuration précédente. Dans tous les cas, les actions de contrôle `Fusion` et `Separation` sont de nouveau disponibles pour tous les agents de contrôle.

La configuration des clusters borne le gain global agrégé maximal qu'il est possible d'obtenir car certains apprenants ne peuvent pas nécessairement effectuer les actions nécessaires pour augmenter leur gain local. L'exploration de nouvelles configurations permet de modifier cette borne et une modification positive indique que cette configuration permet un nouveau gain global agrégé maximal.

Action de contrôle `Separation` L'action de contrôle `Separation` permet la création d'un nouvel agent de contrôle à partir d'un agent de contrôle existant. Elle a un impact direct sur l'évolution du gain agrégé global. Celui-ci partage alors ses apprenants avec le nouveau cluster (figure 6.3a). Cette action a donc pour conséquence d'augmenter le nombre d'agents de contrôle de 1, alors que le nombre d'apprenants reste le même. La décision de séparer en deux les agents locaux du cluster et la façon dont s'effectue cette séparation se basent sur l'évolution du score de confiance des apprenants dans un cluster donné.

En effet, l'agent de contrôle identifie les apprenants proposant des actions impactant positivement son gain agrégé et modifie leur score de confiance en conséquence. À l'inverse, la présence d'apprenants avec un score de confiance faible implique que les actions suggérées jointe n'étaient pas proposées par ces apprenants,

et ne leur permet pas nécessairement d'améliorer leur gain local. Le gain global sera ainsi potentiellement plus faible que dans une configuration où ces agents peuvent effectuer d'autres actions pour augmenter leur gain local. La séparation s'effectue donc en identifiant la contribution de chaque agent et en séparant les agents ayant un score de confiance élevé des agents ayant un score de confiance faible. Cependant, comment s'assurer que la séparation fait bien la différence entre ces deux groupes et que les apprenants se retrouvent dans les clusters appropriés ?

L'action de contrôle `Separation` pourra être effectué si les agents apprenants vérifient deux contraintes. La première est une **zone interdite**, un intervalle de valeurs autour de 0 dans lequel aucun score de confiance ne doit être présent pendant un certain **nombre d'itérations** pour permettre une séparation. En effet, l'absence d'agent dans la zone interdite pendant un certain nombre d'itérations est un indicateur de la stabilité de la configuration actuelle. La **zone interdite** est paramétrable avec les conditions suivantes :

$$0 \geq \text{ borne haute } \geq 1$$

$$-1 \geq \text{ borne basse } \geq 0$$

La taille de la zone interdite doit être suffisamment grande pour que les agents soient bien séparés lorsque l'action de contrôle `Separation` est effectuée (figure 6.4)

La seconde contrainte est que les agents apprenants puissent être séparés en deux groupes en fonction de leur score de confiance, un des groupes étant composé d'apprenants avec un score de confiance négatif et un autre groupe d'apprenant avec un score de confiance positif.

Considérons l'exemple suivant avec un agent de contrôle disposant de 9 apprenants. Cinq d'entre eux ont un score de confiance de -1 et l'autre moitié à un score de confiance de 1. L'agent de contrôle applique une action de séparation pour séparer son cluster en deux nouveaux agents de contrôle ayant respectivement 5 et 4 apprenants. La conséquence de cette séparation est que chaque agent de contrôle peut désormais appliquer les actions environnementales pertinentes basées sur les suggestions de ses apprenants et ainsi en avoir la majorité avec un score de confiance de 1. Il est également possible qu'une partie des apprenants obtiennent à nouveau un score de confiance de -1, si parmi les 4 apprenants du nouveau cluster les actions souhaitées n'étaient pas homogènes.

Lorsqu'un agent de contrôle 1 se sépare en deux agents de contrôle 2 et 3, la séparation est pertinente si :

$$gb_1^c(bs) < g_2^c(as) + g_3^c(as)$$

avec :

— $gb_j^c(bs)$: le gain agrégé de l'agent de contrôle d_j^c avant la séparation ;
— $g_j^c(as)$: le gain agrégé de l'agent de contrôle d_j^c après la séparation.

La réduction du nombre d'agents de contrôle étant un objectif secondaire, la séparation n'est pas considérée comme pertinente en cas d'égalité, si :

$$g_1^c(bs) = g_2^c(as) + g_3^c(as)$$

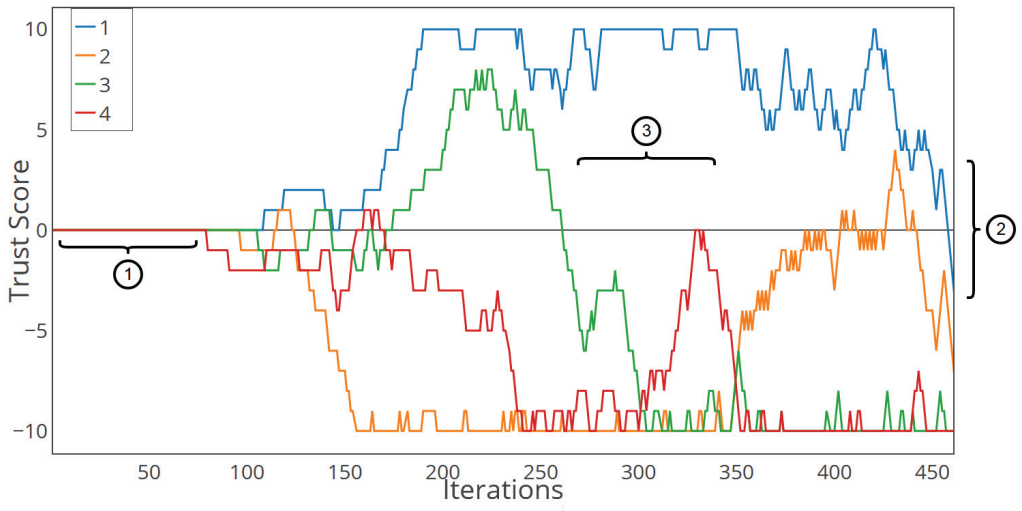


FIGURE 6.4 – Représentation du score de confiance pour 4 apprenants dans un même cluster. (1) correspond à la phase d'exploration où les apprenants proposent une action sélectionnée aléatoirement parmi leur ensemble d'actions environnementales à l'agent de contrôle. (2) est la zone interdite permettant d'observer une disparité marquée entre les apprenants. (3) est une période d'un certain nombre d'itérations, pendant laquelle la consistance des actions suggérées par les apprenants est évaluée. Lorsque l'agent de contrôle identifie une séparation nette entre deux groupes d'apprenants (un avec un score de confiance supérieur ou égal à la borne haut de la zone interdite et l'autre groupe avec un score de confiance inférieur ou égal à la borne basse de la zone interdite) stable pendant un nombre d'itérations égal à la période mentionnée en (3), il applique alors une action de `Separation`.

Action de contrôle `Fusion` L'action de contrôle `Fusion` rassemble deux agents de contrôle ensemble (figure 6.3b). Par conséquent, le cluster résultant contient la totalité des apprenants des deux agents de contrôle initiaux. Cette action réduit le nombre d'agents de contrôle par 1 mais le nombre d'apprenants reste le même. L'objectif de l'action de `Fusion` est de satisfaire l'objectif secondaire de l'algorithme CDQN en minimisant le nombre d'agents de contrôle quand ceux-ci agissent de manière similaire.

L'action de contrôle `Fusion` porte sur l'identification pertinente de *séquences équivalentes d'actions environnementales*. À chaque pas de temps, les agents de contrôle vérifient si les autres agents de contrôle ont la même séquence d'actions environnementales et appliquent une action de `Fusion`.

Quand un agent de contrôle d_j^c reçoit son plus haut gain $gb_j^c \in \mathbb{R}$, il cherche des *séquences équivalentes* chez d'autres agents de contrôle. Après une exploration suffisante, l'agent de contrôle est en mesure d'estimer le meilleur gain agrégé pour son ensemble courant d'apprenants. Lors de la re-génération d'un état aléatoire de l'environnement, l'agent de contrôle sauvegarde la séquence d'actions qui le mène du nouvel état initial à l'état produisant le plus haut gain. La séquence d'actions résultante est ensuite comparée avec les séquences des autres agents de contrôle du système. Pour être considérées comme équivalentes, les séquences d'actions doivent avoir le même nombre de chaque type d'action, quel que soit l'ordre des actions dans la séquence. Considérons trois actions environnementales a, b et c et deux agents de contrôle d_1^c et d_2^c. La séquence $abca$ de d_1^c et la séquence $aacb$ de d_2^c sont considérées équivalentes, par conséquent d_1^c et d_2^c peuvent se fusionner.

Les agents de contrôle peuvent donc uniquement fusionner si l'état des apprenants des agents de contrôle sont *similaires*, c'est-à-dire que les états factorisés ont les même variables d'états. Quand une simulation réinitialise les états de l'environnement, elle utilise une description de l'environnement similaire, spécifiée dans $\mathcal{S}_\alpha^l$. Ainsi, nous considérons les séquences d'actions débutant à partir de n'importe quel nouvel état global initial $s^l \in \mathcal{S}_0^l$ de l'environnement et s'arrêtant lorsque l'agent de contrôle pense avoir atteint son état optimal. Dans notre contexte où les états initiaux des agents sont similaires et les conséquences des actions indépendantes de leur ordre d'exécution, nous considérons que les séquences d'actions équivalentes mènent aux mêmes états pour les apprenants locaux des clusters les ayant effectuées.

En fonction du contexte, il est possible que certaines actions environnementales aient leur action *opposée*, c'est-à-dire une action demandant un changement inverse à celui de l'action à laquelle elle est opposée. Formellement, cela revient à dire que l'action $a_1^l \in \mathcal{A}^l$ effectue une modification ϵ sur un facteur d'état χ_1 et que l'action opposée $a_2^l \in \mathcal{A}^l$ effectue une modification $-\epsilon$ sur le même facteur d'état. Considérons une action environnementale a et son opposée b. Effectuer a dans l'état s_1 mène à l'état s_2, cependant, effectuer l'action b dans l'état s_2 ne mène pas forcément à l'état s_1. Par exemple, considérons les trois précédentes actions environnementales et disons que a est l'opposé de b. Alors la séquence abc et c effectuées à partir du même état initial vont permettre d'atteindre le même état local mais vont être considérées différentes. En effet, l'agent de contrôle avec la séquence c possède un cluster avec de meilleurs actions suggérées que l'agent de contrôle avec la séquence abc. Les scores de confiance des agents des deux clusters seront différents. Par conséquent, nous considérons que les deux clusters ne devraient pas fusionner, même s'ils atteignent un même état avec des séquences qui pourraient être considérées équivalentes.

Pour deux agents de contrôle 1 et 2 qui fusionnent en un agent de contrôle 3, la fusion est pertinente si :

$$gb_1^c(bf) + gb_2^c(bf) \leq g_3^c(af)$$

est vraie après un certain nombre d'itérations, avec :

— $gb_j^c(bf)$: le gain agrégé de l'agent de contrôle d_j^c avant la fusion ;

— $g_j^c(af)$: le gain agrégé de l'agent de contrôle d_j^c après la fusion.

Cette règle a également pour conséquence de minimiser le nombre d'agents de contrôle car la fusion est considérée pertinente même si les gains agrégés avant et après la fusion sont égaux.

À la fin de cette période de temps, les agents de contrôle restent dans la configuration actuelle si l'action est considérée pertinente, sinon l'action de contrôle est annulée et les agents de contrôle reviennent à la configuration avant l'application de leurs actions de contrôle.

Action de contrôle `Pas_de_Modification` L'action `Pas_de_Modification` est particulière dans le cas des agents de contrôle. Cette action a été mise en place pour simplifier le fonctionnement du CDQN. Effectivement, les agents de contrôle n'effectuent pas l'action `Fusion` ou `Separation` à chaque itération mais ont besoin d'explorer un certain temps avant de proposer une des deux actions. Le choix a ainsi été fait d'ajouter la possibilité de ne rien faire aux agents de contrôle et de leur laisser la possibilité d'explorer avant d'exploiter. De plus, l'action `Pas_de_Modification` est imposée aux agents de contrôle lorsqu'un autre agent de contrôle est en train de vérifier la validité d'une action de fusion ou de séparation, permettant ainsi de réduire les cas complexes à gérer.

Stabilité et convergence des clusters La configuration des clusters est stable lorsqu'elle atteint un équilibre de Nash. Notons $\mathcal{C}_j$ l'ensemble des d'apprenants que l'agent de contrôle d_j^c peut posséder, $\mathcal{C}$ l'ensemble des configurations possibles telle que $\rfloor = \rfloor_1 \times \cdots \times \rfloor_\eta$ est une configuration spécifique et $\mathcal{F}$ la fonction d'agrégation pour évaluer le cluster $c \in \mathcal{C}$. Posons c_j est une configuration pour l'agent de contrôle d_j^c et c_{-j} la configuration de tous les autres agents de contrôle. Une configuration $c^* \in \mathcal{C}$ est dans un équilibre de Nash s'il n'existe aucune autre configuration c_j qui serait plus profitable à un cluster tel que :

$$\forall j, c_j \in \mathcal{C} : \mathcal{F}(c_j^*, c_{-j}^*) \geq \mathcal{F}(c_j, c_{-j})$$

Le CDQN atteint donc une configuration stable, potentiellement mauvaise, en atteignant un équilibre de Nash.

Nous allons maintenant voir les deux cas où nous considérons que l'équilibre de Nash peut être atteint :

1. Tout d'abord, la configuration est dans un équilibre de Nash si chaque apprenant présent dans un cluster a un score de confiance de 1 et que le nombre de clusters est minimal (ils ne peuvent plus fusionner car leurs séquences d'actions ne sont pas similaires).
2. D'autre part, la configuration est dans un équilibre de Nash si certains apprenants ont un score de confiance négatif mais que la séparation de ce cluster en deux nouveaux n'améliore pas le gain global.

Voyons maintenant comment le CDQN peut converger en théorie vers une solution optimale et posons :

— $\mathcal{C}_{\text{Nash}} \subseteq \mathcal{C}$ l'ensemble des configurations qui sont dans un équilibre de Nash ;

— $\mathcal{C}^0 \subseteq \mathcal{C}$ l'ensemble des configurations initiales possibles des agents de contrôle ;

— $\mathcal{C}_{\text{Opti}} \subseteq \mathcal{C}_{\text{Nash}}$ l'ensemble des configurations optimales tel que le gain global est supérieur ou égal à au gain global de toutes les autres configurations ;

— $H_{c^0} = (c^0, c^1, \ldots, c^n_{\text{Nash}})$ est l'historique des configurations, qui à partir de n'importe quelle configuration initiale $c^0 \in \mathcal{C}^0$, à une configuration finale se trouvant dans un équilibre de Nash $c_{\text{Nash}} \in \mathcal{C}_{\text{Nash}}$.

L'équilibre de Nash ne correspond pas forcément à une configuration optimale d'où la nécessité de poser $\mathcal{C}_{\text{Opti}}$. Nous faisons l'hypothèse que le système peut converger vers une solution optimale $c_{\text{opti}} \in \mathcal{C}_{\text{opti}}$ à partir de toutes les configurations initiales mais uniquement de certains historiques de configurations $H^*_{c^0}$.

Nous envisageons à l'avenir de faire la preuve de cette hypothèse.

6.2.8 Conclusion et perspectives

Apport des agents de contrôle L'une des contributions de ce chapitre est l'utilisation de l'approche hiérarchique avec l'ajout des agents de contrôle. Pour la replacer dans son contexte, il est important de noter les deux cas extrêmes du CDQN :

— Dans le cas où il n'existe qu'un seul apprenant, le CDQN fonctionne de manière identique à l'algorithme du DQN présenté dans le chapitre précédent ;

— Dans le cas où il y a autant d'agents de contrôle que d'apprenants, c'est-à-dire que chaque agent de contrôle possède un cluster d'un seul apprenant, et si aucun agent de contrôle n'a fusionné, le fonctionnement du CDQN est identique à celui de l'Independant Deep Q-Network.

Les clusters permettent de réduire l'impact de la non-stationnarité sur l'apprentissage, qui se retrouve dans l'IDQN, en forçant l'application d'une action unique pour tous les membres d'un même cluster. Les apprenants sont toujours en mesure d'apprendre de manière individuelle et leurs contributions sont sélectionnées à l'aide de l'attribution du score de confiance. Cependant, il est important de noter que les apprenants ont la même transition si l'application des actions est déterministe. Si l'apprenant dispose d'un effecteur défectueux, l'action appliquée sera différente des autres agents et les états locaux seront différents.

Identification de la contribution Lors de l'apprentissage, les apprenants construisent une politique où il est possible de faire une action sous-optimale pour obtenir un meilleur gain par la suite. Au niveau de l'agent de contrôle, cela se traduit par un gain agrégé également plus élevé. Cependant, l'agent de contrôle ne considère pas une application de score de confiance pour gérer une action qui apparait sous-optimale pour obtenir un meilleur gain agrégé par la suite. Considérons par exemple deux actions environnementales a et b et un état s où le gain agrégé est de $x \in \mathbb{R}$. À l'instant t appliquer l'action a dans l'état s conduit à l'obtention du gain agrégé $y \in \mathbb{R}$ dans l'état s_a alors qu'appliquer l'action b produit le gain agrégé $z \in \mathbb{R}$ dans un état s_b tel que $z > x > y$. Alors si l'action b est majoritairement proposée (et donc sélectionnée), les apprenants proposant b ont un score de confiance augmenté alors que les apprenants proposant a ont un score de confiance diminué. Si l'action a est majoritairement proposée, les apprenants proposant a verront leur score de confiance

diminué. Cependant, si nous considérons qu'à l'instant $t + \Delta t$, effectuer l'action a ou b dans l'état s_a donne un gain y' alors qu'effectuer l'action a ou b dans l'état s_b donne un gain z' tel que $y' > z > z'$ alors les apprenants proposant a ont eu leur score de confiance pénalisés alors que leur apprentissage individuel permet cette anticipation. L'agent de contrôle n'est pas en mesure de faire la différence au niveau de l'attribution du score de confiance.

Impact des actions environnementales Nous considérons ici que les actions environnementales ont un impact équivalent sur les facteurs d'états similaires. Deux états sont compatibles si une partie, ou la totalité de leurs facteurs d'états sont homogènes par rapport à une certaine distance. Une action environnementale modifie la valeur d'un ou plusieurs facteurs d'états de l'espace factorisé d'un apprenant. Pour un facteur d'état χ_1 présent dans l'espace factorisé de deux apprenants i et j, l'action a applique une modification ξ identique pour les deux $x_i \in \chi_1$. Cependant, il est possible que pour une même action environnementale le facteur d'état pour un apprenant d_i^l soit modifié par une valeur x et pour un apprenant j par une valeur y. Le problème dans ce cas est que les séquences d'actions similaires seules ne garantissent plus que les états factorisés des apprenants soient dans des états compatibles.

Choix de vote Actuellement, le vote effectué par l'agent de contrôle lorsque ses apprenants lui suggèrent des actions repose sur un choix à la majorité. Ce type de vote pose problème lorsque deux groupes d'agents de même taille suggèrent des actions différentes, auquel cas l'agent de contrôle choisit une action aléatoirement. Il est possible d'envisager d'autres choix de vote, notamment un choix basé sur la confiance moyenne des agents proposant les actions, soit pour déterminer l'action à prendre en cas d'égalité ou de manière générale. Cependant, l'utilisation d'un choix de vote par majorité permet de forcer l'exploration au lieu d'exploiter toujours l'action proposée par les agents qui ont déjà un score élevé.

6.3 Utilisation du CDQN pour la conception de politiques urbaines

La section précédente présente formellement le modèle du CDQN avec la description de ses différentes composantes de manière générique. Ce modèle peut être utilisé pour représenter l'aspect décisionnel pour la conception de politiques urbaines dans SmartGov. Nous présentons dans cette section comment le CDQN est employé dans SmartGov pour le doter d'une couche macroscopique autonome et adaptative.

6.3.1 Jonction du CDQN avec SmartGov

La simulation urbaine de SmartGov se comporte comme une boite noire aux yeux du CDQN, cependant, il est nécessaire de définir l'ensemble des espaces factorisés $\mathcal{S}^l$ a priori lors de la création de la simulation. Lors de la conception de l'ensemble des structures S, le décideur défini comment celles-ci sont perçues par les agents microscopiques et comment leurs espaces factorisés sont observés par les agents apprenants. Ainsi, la

population d'apprenants $\mathcal{D}^l$ est distribuée sur l'ensemble des structures S définies dans l'environnement de la couche microscopique. L'ensemble des actions environnementales $\mathcal{A}^l$ est donc définie par les structures actionnables.

6.3.2 Découpage de zones géographiques

La zone urbaine considérée par les décideurs lors de la conception de politiques dispose de régions avec un intérêt intrinsèque. La contrainte est alors d'identifier un découpage pour attribuer les structures de l'environnement aux agents politiques. Dans le cadre de SmartGov, il est possible de considérer un découpage expert proposé par le décideur politique à l'aide des entrées de SmartGov. Cette solution a cependant ses limites, car le découpage peut ne pas être pertinent et un des objectifs de la conception de politique sera alors d'en fournir un. Par conséquent, l'un des objectifs du CDQN, dans le cadre de SmartGov, est d'être capable de proposer ces découpages pertinents et de ne pas être restreint à l'application de découpages experts et ce quel que soit le découpage initial, géographique ou non. Les agents locaux étant localisés, nous pouvons alors considérer le cluster comme un découpage géographique de cet environnement. Ainsi, la meilleure configuration de clusters, celle maximisant le gain cumulé global, correspond également à la politique urbaine qui sera proposée au décideur politique. On peut noter toutefois que chaque cluster ne correspond pas nécessairement à une zone géographique continue. Cette contrainte pourrait être ajoutée au niveau de la séparation et de la fusion des clusters, ainsi que d'autres critères comme par exemple la forme de chaque cluster. Nous choisissons ici de ne pas prendre cet aspect de continuité géographique en compte car cela pourrait conduire à des solutions de gain plus faible.

6.3.3 Recouvrement

Les apprenants se partagent l'espace de la simulation urbaine en étant distribués sur les structures de l'environnement. Nous parlons de recouvrement lorsque les apprenants se partagent l'observation d'une ou plusieurs structures (section 6.3.3). C'est-à-dire que leur espace factorisé partage une partie des facteurs d'états. Un espace factorisé peut donc être partiellement partagé par plusieurs apprenants ou uniquement par un apprenant. Cependant, puisque les agents ne communiquent pas, ils ne savent pas qu'ils partagent l'observation des mêmes facteurs d'états. Dans un cadre de recouvrement, la considération de l'action environnementale est plus complexe. Si les apprenants se partageant les facteurs d'états sont dans le même cluster, alors l'action appliquée sur la structure sera la même, à condition que cette action ne soit bien appliquée qu'une seule fois, quel que soit le nombre d'apprenants. Cependant, si les apprenants sont dans des clusters différent, comment l'action environnementale sera-t-elle appliquée ? Est ce qu'il faut considérer la jointure de ces actions ? Une seule d'entre elles ? Ou encore considérer que les actionneurs des apprenants peuvent être faillibles ? Les réponses à ces questions dépendent de la volonté du décideur et du problème modélisé. Il est plus facile de gérer les actions environnementales dans un cadre d'environnement sans recouvrement car le choix de l'action environnementale à appliquer est plus complexe lorsqu'un espace factorisé est géré par deux apprenants présents dans deux clusters différents. Nous considérons ici un espace d'états factorisés sans recouvrement, où

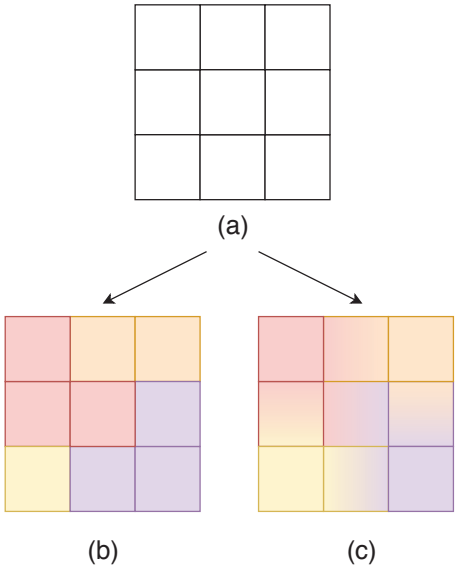


FIGURE 6.5 – Deux découpages d'un même environnement (a) : (b) représente un découpage sans recouvrement et (c) avec recouvrement.

l'apprenant est le seul à observer son état factorisé et n'a pas accès aux observations des autres apprenants.

6.3.4 Fonctionnement général dans SmartGov

La simulation urbaine dispose d'agents microscopiques évoluant dans l'environnement en agissant sur des structures. Au temps t, l'apprenant d_i^l reçoit l'observation o_i^l de son état factorisé s_i^l et suggère une action a_i^s en suivant sa politique π_i. Chaque agent de contrôle reçoit le gain agrégé ainsi que l'ensemble des actions suggérées comme perception et détermine l'action à appliquer $a_j^a, \forall j \in \mathcal{D}^c$. Entre deux itérations du CDQN, les agents microscopiques de la simulation urbaine réagissent aux modifications effectuées sur les structures, ce qui a pour conséquence de modifier leurs facteurs d'états et donc de créer une transition entre les deux états factorisés d'un apprenant. Au temps $t + \Delta t$, l'apprenant reçoit une nouvelle observation de son nouvel état factorisé ainsi qu'une récompense r correspondant à sa fonction de récompense R. Il stocke cette transition et suggère une nouvelle action avec sa politique π mise à jour après apprentissage. L'agent de contrôle attribue pour chaque apprenant dans son cluster un score de confiance en fonction de l'évolution de gain agrégé et de la suggestion de l'action de l'apprenant par rapport à l'action sélectionnée.

Le couplage de SmartGov avec le CDQN introduit une couche évoluant entre deux itérations du CDQN, par conséquent, la fonction de transition T du CDQN peut être déterminée en prenant en compte l'évolution de l'environnement sous l'action d'agents ayant des personnalités différentes pour déterminer le passage d'un état factorisé à un autre. Le chapitre suivant détaille un exemple de couplage entre le CDQN et SmartGov.

6.4 Conclusion et perspectives

Ce chapitre introduit la seconde contribution de la thèse avec le Clustered Deep Q-Network (CDQN). Cette méthode d'apprentissage par renforcement profond multi-agents adresse les problèmes clés identifiés par la communauté, notamment l'apprentissage d'agents indépendants dans des environnements non-stationnaires, et l'identification de la contribution individuelle des agents (*multi-agent credit assigment problem*). Nous avons utilisé l'algorithme d'apprentissage DQN pour calculer la politique individuelle de chaque apprenant en le couplant à un niveau hiérarchique supérieur capable de coordonner l'effort des agents apprenants.

Des améliorations peuvent être apportées pour garantir une meilleure convergence des solutions proposées par le CDQN. Une possibilité est de changer le choix de l'action suggérée par exemple, qui repose actuellement sur une sélection à la majorité. Cette approche est intéressante pour explorer au début de l'apprentissage mais il est possible d'avoir un choix d'action différent au cours de l'apprentissage par l'exploitation des suggestions avec le meilleur score de confiance par exemple. Une autre possibilité est de considérer une forme de méta-action. Dans ce cas, l'action sélectionnée par l'agent de contrôle pourrait être interprétée de différentes manières par les agents apprenants. Les apprenants disposent ainsi d'une plus grande liberté. Cependant, il faut garantir que ces actions impactent dans une moindre mesure les autres apprenants, et que les actions de fusion sont toujours pertinentes. Enfin, une amélioration du CDQN serait de disposer d'agents de contrôle apprenant les configurations pour accélérer la convergence de la configuration des clusters.

Notre seconde contribution, le CDQN, est intégrée à la couche macroscopique de SmartGov en distribuant des agents locaux sur des structures de l'environnement. La fonction de décision des agents politiques décrit dans SmartGov correspond à la politique du DQN de chaque agent apprenant après avoir observé son environnement. La politique urbaine correspond donc à un découpage en zones géographiques obtenue à partir de la configuration des clusters de l'environnement. Les capteurs de la Smart City alimentent en données l'état initial de l'environnement. Les agents locaux apprennent ensuite en complète autonomie les spécificités de leur environnement factorisé. L'instance du CDQN dans le cadre de SmartGov repose sur le choix pertinent d'un ensemble de paramètres que nous avons abordé au long de ce chapitre : le recouvrement des espaces factorisés pour les apprenants, la valeur de l'incrément pour le score de confiance. Le chapitre suivant propose une expérimentation avec le couplage de nos deux contributions pour détailler le fonctionnement du CDQN et l'intérêt des actions de fusions et de séparations pour la création d'une politique urbaine.

Chapitre 7

Co-construction d'une politique urbaine avec SmartGov

Sommaire

7.1 Introduction

L'objectif de SmartGov est de proposer un outil d'aide à la décision pour la co-construction de politiques, c'est-à-dire de permettre aux décideurs politiques d'utiliser SmartGov pour façonner une instance du contexte politique (environnement et agents), d'évaluer des politiques, d'obtenir des suggestions de nouvelles politiques et au final de pouvoir ajuster ces propositions. Le modèle générique SmartGov, introduit dans le chapitre 4, pose les fondations nécessaires (définition d'un environnement, des agents de contrôle, des actions possibles, de la fonction objectif), pour la co-construction de politiques urbaines.

À travers ce chapitre, nous proposons un scénario concret expliquant en détail le fonctionnement de la boucle de co-construction complète à partir de SmartGov. Sur la base de ce scénario, nous proposons différentes expérimentations permettant d'observer les échanges entre la couche microscopique et la couche macroscopique du simulateur. La construction d'une instance du contexte politique est abordée avec la description des étapes nécessaires à la mise en place de l'environnement et des agents de la couche microscopique, ainsi que l'utilisation de l'algorithme CDQN pour construire le niveau macroscopique adaptatif, en cohérence avec le simulateur du niveau microscopique. Enfin, nous discutons des résultats du CDQN et de sa pertinence pour produire des politiques urbaines.

7.2 Construction initiale de l'environnement

Le scénario, précédemment introduit dans le chapitre 4, porte sur une politique de tarification des emplacements de stationnement dans une ville connectée où les usagers cherchent un emplacement pour se rendre à leur lieu de travail. Il va servir d'illustration de la construction d'une politique urbaine avec notre système. Imaginons que le décideur politique exprime le besoin suivant : compte-tenu des différences entre les quartiers dans ma ville, quelle serait la meilleure distribution des places de stationnement et des tarifs à appliquer afin de maximiser les tarifs par l'occupation ? Cependant, d'autres objectifs sont tout à fait envisageable comme la diminution de la congestion dans le centre-ville.

Le gain du décideur est exprimé comme le produit du prix de l'emplacement (p_{emp}) par son occupation (occ_{emp} : 0 si libre ou 1 si occupé). Formellement, l'objectif principal du décideur politique est exprimé par $\max G$ où $G = \sum_{i=0}^{N} p_{emp_i}\ occ_{emp_i}$ avec N le nombre d'emplacements. L'environnement étant naturellement constitué de quartiers plus ou moins attractifs, il faut prendre en compte des zones tarifaires qu'il s'agit de définir de façon pertinente. Par conséquent, l'objectif secondaire du décideur est d'obtenir un découpage pertinent des emplacements dans la ville, pour constituer des fronts de rue ou des quartiers, sur lesquels le même tarif sera appliqué.

Afin de bien définir l'instance, nous devons dans un premier temps extraire toutes les contraintes spécifiques au scénario pour en construire une instance pertinente. Plus le nombre de fronts de rue à prendre en considération est important, moins l'exploration exhaustive de l'espace de recherche est envisageable (temps d'exécution pour parcourir l'espace de recherche important). Il serait d'autre part peu efficace d'appliquer des combinaisons aléatoires de prix et d'emplacements et d'évaluer le gain pour ensuite garder le meilleur obtenu, toujours à cause de la taille de l'espace de recherche.

Les données utilisées pour ce scénario sont la position géographique de capteurs pour des emplacements dans une partie de la ville de Los Angeles [51, 259]. Le réseau routier et les bâtiments sont extraits d'Open Street Map [1]. Cette simulation est réalisée en utilisant la plateforme multi-agent Repast Simphony [179] et l'architecture SmartGov (détails disponibles en annexe).

Une fois l'instance de SmartGov complétée, le décideur politique disposera d'un simulateur représentant des usagers se déplaçant chaque jour jusqu'à leur lieu de travail en trouvant un emplacement de stationnement satisfaisant, sur lequel il pourra observer les actions politiques appliquées pour modifier les prix des emplacements des fronts de rue, et trouver ainsi la meilleure configuration possible maximisant son gain.

Nous allons maintenant présenter les instances des modèles élaborées pour les deux niveaux de SmartGov, en détaillant les choix de modélisation effectués pour ce scénario.

7.2.1 Instance de la couche microscopique

Le modèle formel de l'environnement de SmartGov (section 4.4.1) est utilisé pour décrire les éléments constituant la couche microscopique par équation (4.1). Les usagers se déplaçant en voiture dans la ville vers leur lieu de travail sont modélisés par des agents dits microscopiques. Nous avons choisi de considérer différents profils de conducteurs (personnalités), représentant les différents comportements possibles dans la recherche de places de stationnement, pour avoir une population hétérogène. Il y a ainsi les personnes pressées, les économes, les personnes qui acceptent de se garer loin et de marcher, etc.(section 7.3.1). Les origines et destination sont aussi décrits dans le modèle.

Structures présentes dans la ville et actions considérées L'ensemble des structures de la ville est défini par :

$$\mathcal{S} = \mathcal{S}_{emp} \cup \mathcal{S}_{bat}$$

où :

— $\mathcal{S}_{emp}$ est l'ensemble fini des emplacements de stationnement disponibles ;

— $\mathcal{S}_{bat}$ est l'ensemble fini des bâtiments de la ville.

Un bâtiment $s_{bat} \in \mathcal{S}_{bat}$ est un tuple tel que :

$$s_{bat} = \langle \text{lieu de travail}, \{position\}, \emptyset \rangle$$

où la *position* est perçue par les agents microscopiques. Pour notre scénario, les bâtiments sont tous du type *lieu de travail*, mais il serait tout à fait possible d'envisager d'autres types : par exemple un bâtiment de type *point d'intérêt*, représentant des lieux de restauration ou de divertissement, ou encore un bâtiment de type *résidentiel*. Ces types ont une influence sur l'intérêt que l'agent leur porte pour stationner à proximité. Nous considérons aussi que l'agent ne peut pas agir sur les bâtiments, mais seulement sur les emplacements avec l'action *Stationner*. En fonction de l'objectif de la simulation, il pourrait être pertinent de représenter

1. Les données sont disponibles sur le site d'Open Street Map où le décideur peut choisir de représenter n'importe quelle zone disponible.

des actions sur les bâtiments, comme l'agent entrant ou sortant de son lieux de travail par exemple (actions *Entrer* et *Sortir*), pour un scénario où il serait nécessaire de compter le nombre de personnes présentes dans un immeuble. Ici seule la position de l'entrée du bâtiment est perceptible et connue de l'agent à tout instant, pour évaluer la distance de marche jusqu'à la cible. L'action *Sortir* de l'emplacement n'est pas envisagée sur le temps de simulation.

Pour le scénario considéré, les emplacements sont tous de type *on-street*, ce qui signifie qu'ils sont présents dans la rue, par opposition aux emplacements *off-street* comme les parkings couverts.

Un emplacement $s_{emp} \in \mathcal{S}_{emp}$ est défini par :

$$s_{emp} = \langle \text{on-street}, \{tarif, occupation, position\}, \{Stationner\} \rangle$$

où

- $tarif$ est le prix actuel de l'emplacement ;
- *occupation* est l'état de l'emplacement (libre ou occupé) ;
- *position* correspond à la position de l'emplacement et permet de connaitre sa distance avec la destination cible de l'agent.

À partir de l'ensemble $\mathcal{S}_{emp}$ il est possible de définir l'ensemble des fronts de rue $\mathcal{S}_{fdr}$ tel que $\mathcal{S}_{fdr} \subseteq 2^{S_{emp}}$ et où un front de rue $s_{fdr} \in S_{fdr}$ correspond à une agrégation de plusieurs emplacements : $s_{fdr} = (s_{emp,1}, s_{emp,2}, \dots) \in 2^{S_{emp}}$. C'est une structure qui n'est pas perceptible par l'agent microscopique, et elle sert d'interface avec la couche macroscopique :

$$s_{fdr} = \langle \text{on-street}, \emptyset, \emptyset \rangle$$

Cette structure sera définie de façon plus approfondie au niveau de la couche macroscopique.

Graphe pour les déplacements Le choix du graphe G est un réseau routier où les nœuds correspondent aux intersections, les arcs aux tronçons routiers, et la fonction de poids ω correspond à la longueur d'un tronçon routier. Il est également possible d'envisager que ω représente d'autres métriques, comme la densité de véhicules sur le tronçon. Dans ce cas pour son déplacement, l'agent pourrait chercher à minimiser le trafic de la route alors que dans notre scénario, il va chercher à minimiser la distance à parcourir. Ainsi à partir de l'entrée de l'agent dans le périmètre étudié, jusqu'au nœud le plus proche de son lieu de travail, le plus court chemin possible sur le réseau est trouvé avec l'algorithme A* [93], extension de l'algorithme de Dijkstra [66]. Cependant, tout autre algorithme pourrait être utilisé pour modéliser le déplacement des agents et optimiser d'autres paramètres que le temps ou la distance.

Comportement de l'agent microscopique Le comportement des agents cherchant des emplacements de stationnement dans une ville est modélisé par un automate à états finis (figure 7.1). Pour cela, nous nous sommes inspirés de travaux décrivant les méthodes de recherche d'emplacements selon les profils [100, 252]. Il faut ainsi observer que si l'on veut une modélisation réaliste, un pré requis est la capacité à injecter des

connaissances métier (ici sur les usagers) liées à la politique urbaine étudiée, le plus souvent avec l'aide d'experts. Des études sociologiques peuvent être utilisées, comme nous l'avons fait pour notre scénario.

Les perceptions de l'environnement que l'agent peut prendre en compte sont les suivantes :

— $\lambda_{d_{bat}}$: la distance entre son lieu de travail et sa position courante (en mètres) ;
— $\lambda_{i_{occupation}}$: la disponibilité de l'emplacement (occupé ou libre) ;
— $\lambda_{i_{tarif}}$: le prix du stationnement de l'emplacement ;
— $\lambda_{d_{emp}}$: la distance entre son lieu de travail et l'emplacement envisagé.

La perception $\lambda_{d_{bat}}$ est utilisée pour définir une partie des transitions des états de l'automate pour décrire le comportement de recherche d'emplacements. Les autres perceptions servent à déterminer la validité d'un *emplacement*. La manière dont un agent microscopique exploite ses perceptions pour évaluer les emplacements perçus dépend de la personnalité des agents et sera décrite dans la section 7.3.1.

L'automate à états finis est donc défini par son ensemble d'états, son ensemble de perceptions pour les transitions d'états, sa fonction de transition, son état initial et l'ensemble de ses états finaux, de la manière suivante pour notre scénario :

$$\mathcal{A}_{a_h^n} = (\{avance, recherche, deambule, pause\}, \{\lambda_{d_{bat}}, chemin, emplacement\}, \delta, avance, \{pause\})$$

tel que δ est défini par :

$$\delta(\text{avance}) = \begin{cases} \text{recherche} & \text{si } \lambda_{d_{bat}} \leq 250\text{mètres est vrai,} \\ \text{avance} & \text{sinon} \end{cases}$$

$$\delta(\text{recherche}) = \begin{cases} \text{pause} & \text{si } emplacement = valide \text{ est vrai,} \\ \text{recherche} & \text{sinon si } chemin \neg complet \text{ est vrai,} \\ \text{deambule} & \text{sinon} \end{cases}$$

$$\delta(\text{deambule}) = \begin{cases} \text{pause} & \text{si } emplacement = valide \text{ est vrai,} \\ \text{deambule} & \text{sinon} \end{cases}$$

Chaque agent possède un lieu de travail $s_{bat} \in S_{bat}$ et un point de départ attribués aléatoirement. Son état initial e_0 est *avance* et il entre dans la simulation à partir de son point de départ. Il effectue l'action *déplacer* lui permettant de se rapprocher de son lieu de travail en suivant le plus court chemin entre sa source et une intersection la plus proche possible du lieu de travail. Sa vitesse de déplacement est déterminée à l'aide du modèle de Gipps, un algorithme de poursuite gérant la vitesse en fonction des obstacles [86] (ralentissement en cas de congestion, accélération quand la voie est libre). Il rentre ensuite dans l'état *recherche* lorsque la distance est inférieure à 250 mètres dans lequel il évalue les emplacements de stationnement qu'il perçoit. La transition (emplacement = valide) signifie que l'agent a trouvé un emplacement pertinent sur lequel il effectue l'action *Stationner*. Tant que l'agent ne trouve pas d'emplacement valide, il reste dans l'état *recherche* jusqu'à arriver à sa destination. À partir de ce moment et comme il n'a toujours pas trouvé d'emplacement, il

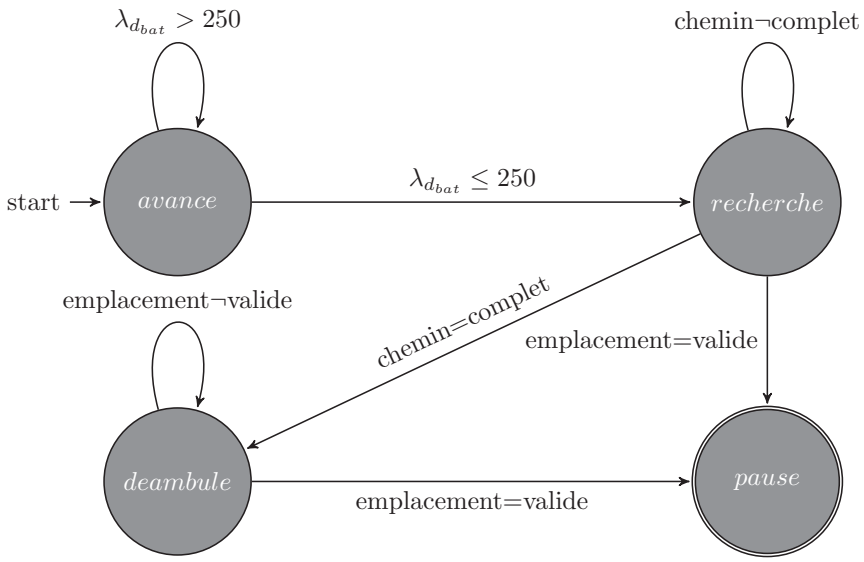


FIGURE 7.1 – Cet automate à états finis est utilisé pour la recherche d'emplacements par les éléments de N_h. L'état initial est *avance* et l'état final est *pause*, atteint lorsque l'agent stationne dans un emplacement valide.

entre dans l'état *déambule* où il en cherche un dans les rues avoisinantes en suivant un déplacement en *corolle* ou *spirale* (*circling* en anglais) [220]. Pour représenter ce type de déplacement, nous proposons l'algorithme 4 où l'agent détermine la route à suivre lorsqu'il arrive à une intersection en considérant la distance entre la fin du tronçon et son lieu de travail, son espace de recherche et le nombre de fois qu'il a déjà traversé ce tronçon dans sa recherche. La taille maximale de l'espace de recherche *edr* dépend de la personnalité de l'agent, c'est-à-dire de la distance qu'il est prêt à couvrir pour rejoindre son lieu de travail à pied. L'évolution $\lambda(t)$ de son espace de recherche dépend du temps qu'il passe à chercher et également de sa personnalité. En effet, un agent pressé par le temps sera prêt à chercher un emplacement plus éloigné. L'évolution de la taille de l'espace de recherche peut être linéaire ou exponentielle par exemple. Nous faisons ici le choix d'une évolution linéaire. L'agent possède également une information sur les tronçons déjà parcourus ainsi que le nombre $M(t)$ de fois qu'ils ont été traversés pendant la recherche d'une place.

Une fois tous ces éléments définis, l'instance de la couche microscopique est disponible et prête pour la simulation. Le décideur politique peut dès à présent observer le fonctionnement de la ville et analyser le comportement des agents. Pour qu'il puisse ensuite tester des politiques urbaines, il est nécessaire d'instancier le modèle décisionnel du niveau macroscopique, pour faire ainsi évoluer l'environnement et observer comment les agents microscopiques ajustent leur comportement en réponse à ces changements.

Algorithme 4 : Choix de tronçon lors de la recherche d'emplacement de stationnement

Input : Un ensemble de tronçons T adjacents au tronçon courant, nombre de parcours des tronçons $M(t)$, edr distance maximale, $\lambda(\tau)$ la fonction d'accroissement de l'espace de recherche, τ le temps passé à rechercher un emplacement

Result : Le tronçon choisi t, edr mis à jour, $M(t)$ mis à jour

Initialiser le score $S(t)$ de chaque tronçon t
$edr \leftarrow edr + \lambda(\tau)$
for $t \in T$ **do**
    $d \leftarrow distance(t_{end}, s_{bat})$ // t_{end} est la fin du tronçon (dans le sens de la marche) et s_{bat} la destination
    $S(t) \leftarrow d \times (1 + M(t))$
end
$t = \text{argmin}(S)$
$M(t) \leftarrow M(t) + 1$

7.2.2 Instance de la couche macroscopique

Le niveau macroscopique correspond au gestionnaire de politiques : c'est un modèle décisionnel composé d'agents politiques de contrôle et d'agents politiques locaux qui préconisent des décisions (mesures politiques) en fonction d'un objectif défini. Le modèle du gestionnaire de politiques (équation (4.8)) décrit donc l'ensemble des états possibles, les perceptions disponibles des agents présents dans cette couche ainsi que l'ensemble fini des actions qu'ils peuvent appliquer. Par rapport aux actions de la couche microscopique, il s'agit cette fois d'actions politiques en ce sens qu'elles font partie d'une séquence d'actions, traduisant une politique visant un objectif déterminé. Les entrées du modèle sont fournies par la couche microscopique ; en retour la couche macroscopique préconise des actions.

Cette section présente comment l'algorithme CDQN, notre contribution présentée au chapitre précédent, peut servir de couche macroscopique à SmartGov et comment il est possible de l'interfacer avec n'importe quelle instance microscopique de SmartGov.

Dans un premier temps, nous décrivons comment le décideur choisit la granularité souhaitée pour sa politique. Nous définissons ensuite l'instance complète du niveau décisionnel correspondant au gestionnaire de politiques de notre scénario. Enfin, nous décrirons les deux composantes de l'algorithme CDQN : les agents locaux (qui apprennent quelle meilleure action appliquer sur leur périmètre), et les agents de contrôle qui gèrent des clusters d'agents locaux.

Granularité des actions politiques Dans un premier temps, le décideur politique doit identifier la granularité nécessaire de l'environnement pour que sa politique ait un sens et que les actions politiques soient pertinentes. En effet, si le décideur choisit une granularité très fine où par exemple l'action porte directement sur *un* emplacement, alors la visibilité globale de la politique tarifaire produite sera difficile à percevoir, des tarifs différents pouvant être proposés pour des emplacements d'un même front de rue. Ce type de politique est généralement peu appréciée par les usagers, ou tout simplement complexe à mettre en place en pratique.

À l'inverse, si le décideur choisit une granularité *gros grain* où l'action politique modifie un ensemble de fronts de rue, la politique tarifaire obtenue proposera des tarifs identiques pour des fronts de rue peut-être très différents, certains plus demandés que d'autres. Cela risque d'aller à l'encontre de l'attente du décideur puisque sans tenir compte des spécificités des quartiers, son gain risque d'être plus faible.

Doté de connaissances expertes sur la ville, les quartiers existants et leurs spécificités, le décideur politique propose ainsi un découpage de la ville en zones (appelés fronts de rue dans le cadre du stationnement) avec un niveau de granularité qui a du sens, sachant que qu'une action politique va modifier de façon identique tous les emplacements d'un même front de rue. En fonction des résultats, il pourrait également modifier la politique tarifaire produite en spécifiant une granularité hétérogène : les actions politiques modifieraient tous les emplacements d'un même front de rue sauf pour un quartier composé de plusieurs fronts de rue, où l'action politique jouerait sur la totalité des emplacements.

L'objectif secondaire du décideur est d'obtenir le découpage géographique le plus intéressant afin de satisfaire la fonction objectif du décideur.

Instance du gestionnaire de politiques Le gestionnaire de politiques décrit l'interface entre la couche microscopique et la couche macroscopique en décrivant les perceptions et les actions disponibles pour les agents politiques. Le décideur politique considère donc le front de rue comme la structure modifiée par les actions politiques. Puisque le front de rue contient un ensemble d'emplacements, les données de ces emplacements sont agrégées pour être perçues par les agents politiques. Les perceptions disponibles ainsi que la méthode d'agrégation de chacune des données sont le résultat des choix du décideur. Le scénario se déroulant dans une ville connectée (ou une ville dans laquelle des statistiques sur les usagers sont disponibles via des enquêtes), nous imaginons que chaque emplacement dispose de données sur les conducteurs qui l'occupent : leur satisfaction, leur utilité, la distance entre l'emplacement et leur lieu de travail ainsi que leur temps de recherche. Dans notre cas d'étude, le décideur politique utilise la moyenne des données comme méthode d'agrégation, et l'ensemble fini

$$I_{\mathcal{H}} = \{i_{prix}, i_{empD}, i_{empT}, i_{sas}, i_{uti}, i_{dis}, i_{tps}\}$$

tel que :

- i_{prix} : le prix du front de rue (en euro, allant de 50 centimes à 4 €) ;
- i_{empD} : le nombre d'emplacements disponibles ;
- i_{empT} : le nombre total de places ;
- i_{sas} : la moyenne de la satisfaction des agents stationnés (de 0 à 1) ;
- i_{uti} : la moyenne de l'utilité des agents stationnés (de 0 à 1) ;
- i_{dis} : la moyenne de la distance entre l'emplacement et le lieu de travail des agents stationnés (exprimé en mètres. Un emplacement sans agent renvoie une valeur jugée maximale, ici fixée à 250) ;
- i_{tps} : le temps de recherche moyen (exprimé en secondes. Un emplacement sans agent à la fin de la simulation renvoie une valeur de 150. Cette valeur permet à l'agent d'apprendre à reconnaitre les cas où les agents ne stationnent pas, plutôt que d'avoir une valeur égale à 0 par exemple).

Une représentation de l'environnement $r_{\mathcal{H}} \in R_{\mathcal{H}}$ est donc définie par la combinaison de ces 7 valeurs.

Ensuite, toujours pour compléter l'interface entre la couche microscopique et la couche macroscopique, le décideur doit définir les actions environnementales dont l'agent local apprendra les spécificités. Le scénario d'exemple porte sur une politique urbaine de tarification des emplacements de stationnement. Nous imaginons que pour réaliser cela, le décideur joue sur le tarif et le nombre d'emplacements, d'un même front de rue. Le décideur ajoute donc les actions de modifications des prix dans à l'ensemble des structures emplacements $\mathcal{S}_{emp}$, et des actions d'ajouts et de suppressions d'emplacements à l'ensemble des structures fronts de rue $\mathcal{S}_{fdr}$ (dans la ville connectée, les emplacements pourraient être gérer sur voirie, par exemple avec par un marquage dynamique ou encore des bornes sortant du sol). L'ensemble des actions possibles des agents locaux du niveau macro est ainsi :

— une augmentation des prix (représentée par $\nearrow$) par incrément de 50 centimes ;
— une diminution des prix ($\searrow$) par décrement de 50 centimes ;
— une augmentation du nombre d'emplacements ($+$) par incrément d'un emplacement ;
— une réduction du nombre d'emplacements ($-$) par décrement d'un emplacement.

L'ensemble fini des actions du gestionnaire est donc

$$A = \{\nearrow, \searrow, =, +, -\}$$

où ($=$) correspond à l'action n'ayant aucun impact sur la tarification (soit l'équivalent de *ne rien faire*).

La description complète du front de rue $s_{fdr} \in \mathcal{S}_{fdr}$ est donc :

$$s_{fdr} = \langle \text{on-street}, \{\emptyset, i_{\mathcal{H}}\}, \{\emptyset, A\} \rangle$$

Le nombre d'agents politiques locaux de la couche macroscopique correspond au nombre d'agents apprenants de l'algorithme de CDQN, ces agents sont maintenant présentés en détail pour notre scénario.

Agent politique local Dans un premier temps, le décideur politique doit associer les agents politiques locaux aux structures de l'environnement (figure 7.2), donc ici aux ensembles de fronts de rue $\mathcal{S}_{fdr}$. Un agent local est associé au même front de rue pendant toute la durée de la simulation. Le nombre d'agents locaux ne varie pas pendant la simulation.

Le décideur propose ensuite une répartition initiale des agents locaux en clusters correspondant au niveau de granularité choisie par le décideur. Chaque cluster sera géré par un agent politique de contrôle, qui appliquera une unique action à chaque pas de la simulation.

Le choix de la granularité détermine la distribution initiale des agents locaux et a une influence sur la qualité de l'apprentissage. L'espace factorisé de chaque agent local va correspondre aux facteurs d'états des fronts de rue qui lui sont attribués. Un agent politique local d_m^p est donc défini par :

$$d_m^p = \langle s_{fdr}^m, I_{s_{fdr}}, \{\nearrow, \searrow, =, +, -\}, \rho \rangle$$

La fonction de décision ρ détermine l'action politique qui s'appliquera à tous les emplacements qui lui sont

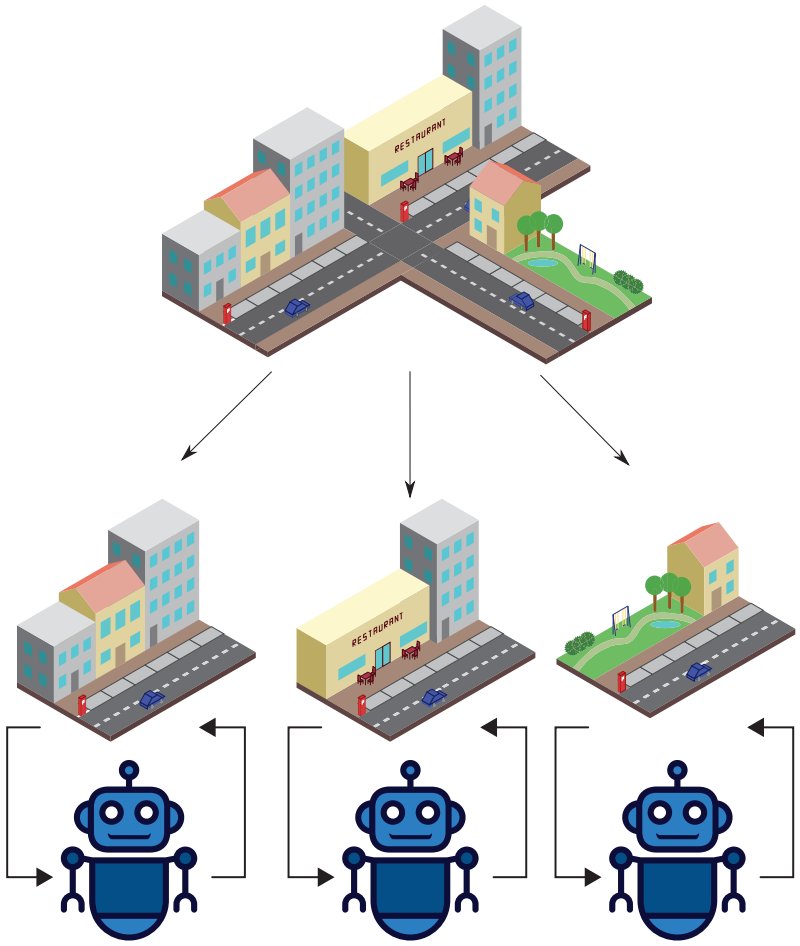


FIGURE 7.2 – Un quartier (en haut) peut être découpé en plusieurs fronts de rue contrôlés par des agents politiques locaux (en bas). Ces derniers ne perçoivent que les fronts de rue auxquels ils sont associés. Le quartier est un cluster géré par un agent politique de contrôle. Le cluster définit la même action à appliquer par tous ses agents locaux.

attribués. Afin de conserver une approche adaptative et autonome, l'agent politique local connait le nombre d'actions qu'il possède, lorsqu'il se voit attribuer une ou plusieurs structures, mais sans disposer d'information a priori sur leur impact.

La fonction objectif du décideur étant de maximiser le gain obtenu par l'occupation d'emplacement l'action $(-)$ n'est jamais pertinente. Cependant, l'ajout de cette action parmi l'ensemble des actions environnementales disponibles pour les agents permet de représenter les cas où les décideurs donnent plus d'informations que nécessaire au système, et sert donc à tester la robustesse de l'approche vis-à-vis de ce cas.

Dans le scénario de tarification, les agents politiques locaux sont homogènes dans la mesure où la description des perceptions, des actions et de la fonction de décision sont identiques pour tous les agents. Mais comme la

perception de l'agent local dépend de son front de rue, deux agents locaux ont généralement des ensembles de perceptions différents. Par exemple, un front de rue est composé de 12 emplacements et un autre en a seulement 5. Cela implique qu'il n'est pas possible d'avoir un apprentissage centralisé où un agent apprendrait les caractéristiques d'un front de rue et le distribuerait ensuite à l'ensemble des agents.

Chaque agent politique local reçoit les informations agrégées des agents microscopiques stationnés sur les emplacements du front de rue qu'il perçoit. Ces informations sont disponibles sous forme d'un vecteur d'entrées, sans autre connaissance de l'environnement microscopique (figure 7.3). Il s'agit de données d'usagers moyennées mais le vecteur de perception est juste un ensemble de valeurs numériques, sans informations sur son type (tarif, occupation des places, satisfaction moyenne, ...).

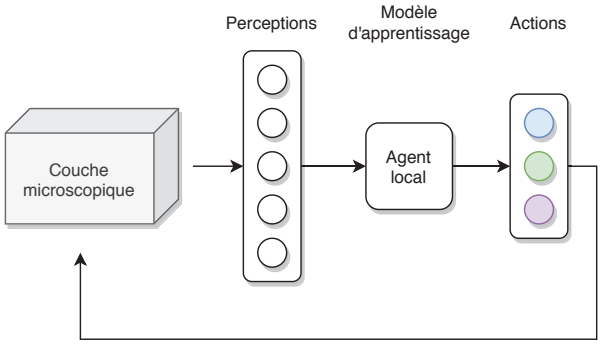


FIGURE 7.3 – L'agent reçoit en entrée un vecteur non annoté correspondant aux perceptions et propose en sortie un nombre correspondant à une action à appliquer sur l'environnement. La couche microscopique est inconnue de l'agent politique.

Agent politique de contrôle L'agent local observe son front de rue et souhaite améliorer ses performances individuelles en proposant, à son agent global, une action.

Afin de conserver une cohérence entre la description de l'agent politique de SmartGov et le fonctionnement multi-niveaux du CDQN, l'agent de contrôle est également décrit en utilisant l' équation (4.10).

Il est défini par :

$$d_j^c = \langle \mathcal{S}_{j,fdr}^c, I_j^c, \{Fusion, Separation\}, \rho^c \rangle$$

où :

— $\mathcal{S}_{fdr}^c \subseteq \mathcal{S}_{j,fdr}$ l'ensemble fini des fronts de rue sous la responsabilité de l'agent de contrôle j.

Notons d_j^c l'agent de contrôle j et d_i^l l'agent apprenant local i.

L'ensemble des structures d'un agent politique global d_j^c est l'ensemble joint des ensembles des structures des agents locaux qu'il gère, soit $\mathcal{S}_j^c = \cup_{i \in d_j^c} \mathcal{S}_i$ où N_j est le nombre d'agents locaux associés à l'agent de contrôle d_j^c. Ce nombre N_j est susceptible de changer lorsque les agents de contrôle affinent leurs clusters d'agents locaux pour améliorer leurs gains agrégés.

Il faut noter que les perceptions de l'agent de contrôle sont différentes de l'ensemble joint des perceptions

des agents locaux composant le cluster. En effet comme perception, l'agent de contrôle reçoit 2 éléments : le gain agrégé, obtenu par agrégation du gain de ses agents locaux à partir de leur fonction objectif, ainsi que l'action jointe suggérée par ses agents locaux. Ainsi, l'ensemble fini des perceptions de l'agent de contrôle d_j^c est décrit par :

$$I_j = \{\boldsymbol{g}_{d^l}, \boldsymbol{a}_j^s\} \tag{7.1}$$

avec $\boldsymbol{g}_{d^l} = (g_{d_0^l}, g_{d_1^l}, \ldots, g_{d_n^l})$ où $g_{d_i^l}$ est le gain de l'agent local i et où $\boldsymbol{a}_j^s$ est l'*action jointe suggérée.*

Nous considérons ici que les agents de contrôle disposent de perceptions exactes, les informations transmises par les agents apprenants étant toujours considérées comme parfaites.

Les actions de l'agent de contrôle ainsi que la fonction de décision sont les mêmes que celles définies dans le chapitre précédent (section 6.2.7) avec la fusion et la séparation.

L'instance de la couche macroscopique est désormais définie. Nous illustrons maintenant comment cette couche peut être couplée à l'algorithme de CDQN pour proposer des actions sur l'environnement et disposer d'un découpage géographique.

7.2.3 Mise en œuvre de CDQN et couplage des dynamiques

Nous avons mentionné que la couche macroscopique ne connait pas le fonctionnement de la couche microscopique ni ce qu'elle représente. La couche microscopique fonctionne comme une boite noire aux yeux de la couche macroscopique. La couche microscopique fournit simplement aux agents politiques locaux un vecteur de perceptions similaire à celui décrit dans la section 7.2.2. L'avantage de ce fonctionnement est de permettre à la couche macroscopique d'être entièrement indépendante en n'ayant aucune connaissance a priori de la couche microscopique dont elle modifie l'environnement.

Afin de construire un gestionnaire de simulations pertinent, nous introduisons les notations suivantes :

Un **simulateur** est une instance de SmartGov composée de deux **modules** : la couche microscopique et la couche macroscopique. Une **simulation** est ici l'évolution des agents dans la ville pendant un certain nombre d'itérations ou **ticks** donnés. Les agents microscopiques arrivent en même temps dans la simulation et cherchent un emplacement pendant une heure dans la matinée. Une **séquence de simulations** représente ici un ensemble de journées pendant lesquelles les agents cherchent un emplacement avec des prix qui n'évoluent pas. Une **série de séquences** ou **époques** correspond à la re-génération d'un état initial de la simulation pour forcer l'exploration.

La configuration du gestionnaire de simulations permet au décideur politique de gérer le couplage entre les deux modules. Elle conduit à se poser les questions suivantes :

— Pendant combien de temps la simulation va-t-elle se dérouler ?
— Comment évolue l'agent humain dans une même séquence de simulations ?
— De combien de simulations est composée une séquence de simulations ?

La première question relative au calibrage de la durée de la simulation, a un impact important sur le temps nécessaire à l'apprentissage pour trouver une solution stable. Bien que le décideur politique soit moins sensible à la problématique du temps de simulation pour l'apprentissage, il est nécessaire qu'il le calibre

correctement pour bien exploiter SmartGov. En effet, la simulation d'un système complexe est coûteuse en temps et la couche macroscopique enregistre une transition entre chaque séquence de simulations. Par conséquent, plus une simulation dure longtemps, plus l'accumulation d'expériences pour l'apprentissage sera importante. Imaginons par exemple le cas où un décideur choisit d'avoir une centaine d'agents microscopiques et qu'il simule leur trajet quotidien pour se rendre à leur lieu de travail dans la matinée. S'il fait le choix d'étaler les trajets entre 6h et 11h alors une partie du temps de simulation sera alors inutile, certains agents étant déjà arrivés à leur lieu de travail et d'autres tournant en boucle sans trouver d'emplacement. Il serait ici plus efficace de restreindre le temps de simulation entre 8h et 9h, cela diminue notamment le nombre de *ticks* nécessaire pour obtenir un résultat significatif. Nous observons avec cet l'exemple que, bien que le choix d'une plage horaire particulière soit arbitraire (ici entre 8h et 9h), il impacte le nombre d'itérations nécessaires pour la simulation et doit donc être déterminé avec soin. Pour nos expérimentations, le *tick* d'itération de la simulation de la couche microscopique a été calibré à une seconde.

La deuxième question porte sur l'évolution du comportement des agents humains simulés. Dans notre modèle, c'est la personnalité de l'agent qui définit une fonction de calcul de score de chaque emplacement en fonction de son prix, de la distance à son lieu de travail et d'autres paramètres définis par le décideur. Après un certain laps de temps paramétrable, l'agent considère qu'il ne va pas trouver d'emplacement de stationnement satisfaisant avant la fin de la simulation, c'est-à-dire ayant une utilité supérieure ou égale à une limite fixée par la personnalité de l'agent. Un conducteur humain ne trouvant pas d'emplacements satisfaisants (trop chers, ou toute autre raison) peut être amené, après quelques jours, à modifier son comportement de mobilité [218] (il va privilégier les transports en commun par exemple). Nous avons donc intégré le changement de mobilité (choix d'une alternative à la voiture) pour les agents dans cette situation. La décision dépend du niveau de satisfaction de l'usager, selon le processus suivant : dans la première simulation, comme l'agent ne connait pas les prix des emplacements, il va donc chercher pendant un certain temps une place où se garer. Il est tout à fait possible que l'agent finisse par stationner mais dans ce cas, sa satisfaction sera assez faible, et si l'agent ne trouve pas d'emplacement satisfaisant à la fin de la simulation, sa satisfaction est minimale. À la seconde simulation, l'agent tente une nouvelle fois de trouver un emplacement de stationnement, généralement sans plus de succès que lors de la précédente étape. Lors des simulations suivantes, l'agent va alors calculer la probabilité qu'il a de tenter à nouveau de trouver un emplacement, par rapport à celui du choix d'une mobilité alternative.

Cependant, pour des raisons de simplification et parce que ce changement de comportement précédent n'était pas utile pour tester l'apprentissage de politiques, notre étude ne représente pas les mobilités alternatives mais permet juste à l'agent de faire le choix entre l'utilisation de sa voiture ou une alternative quelconque. Toutefois pour traiter cette situation d'échec du stationnement, nous avons donné la possibilité au décideur politique de spécifier, via la taille d'une *fenêtre de décision*, le nombre de simulations pendant lesquelles l'agent peut tenter de trouver un emplacement. Ainsi, une fenêtre de décision de taille trois implique que l'agent va tenter, pendant encore deux nouvelles simulations, de trouver un emplacement après la première simulation, même s'il n'a trouvé aucun emplacement de stationnement satisfaisant. Après chaque simulation, l'agent conserve un historique de sa satisfaction h (comprise entre 0 et 1 et égale à 0 lorsqu'il ne trouve pas

d'emplacement) où h_i est la satisfaction à la simulation i et apporte plus d'importances aux échecs récents que passés avec un vecteur de poids λ, dont les éléments décroissent (λ est similaire au *discount factor* du Q-learning). La satisfaction est calculée à l'aide de l'équation (4.7). Le décideur politique choisi également une valeur minimale de score à partir de laquelle l'agent continue à chercher des emplacements. Dès que cette valeur est dépassée, l'agent calcule alors la probabilité d'utiliser une modalité alternative (figure 7.4). À partir du moment où l'agent décide de prendre une modalité alternative, il cesse d'être simulé jusqu'à la prochaine séquence de simulations.

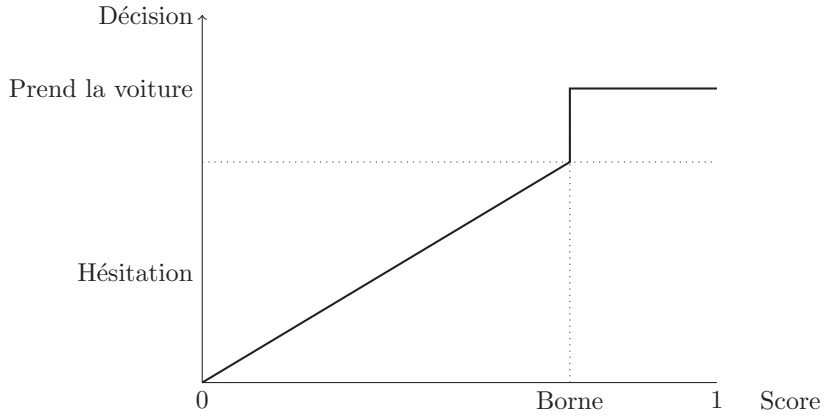


FIGURE 7.4 – Représentation du fonctionnement de choix d'alternative pour l'agent humain en fonction de son score de satisfaction. Lorsque le score est supérieur à la borne, l'agent prend automatiquement son véhicule et cherche un emplacement. Lorsque le score est inférieur à la borne, l'agent considère une alternative à la voiture et calcule sa probabilité de prendre cette alternative.

La troisième question liée au calibrage de la simulation concerne la taille de la séquence de simulation. Ce calibrage permet de gérer le compromis entre laisser l'agent humain évoluer vers un nouveau comportement, ou choisir d'adapter la politique au niveau macroscopique. À cet effet, la taille de la séquence définit à quelle fréquence est appliquée chaque action politique. Imaginons dans ce scénario que le décideur politique fasse le choix d'une séquence de 5 simulations (ce qui correspond aussi à une fenêtre de décision de taille 5), avec un horizon de 30 séquences (soit 30 actions environnementales effectuées). Les agents politiques locaux perçoivent donc l'environnement toutes les cinq simulations, et appliquent alors une action sur l'environnement. Pour simplifier les expérimentations, nous considérons que les agents humains ne sont sensibles qu'aux tarifs pratiqués pour le stationnement. À chaque nouvelle séquence de simulations, il y a donc soit une modification tarifaire, soit une modification du nombre d'emplacements, soit aucun changement. Ainsi, il est nécessaire de simuler 5 fois le comportement des agents pour un même prix d'emplacement avant que les agents politiques n'enregistrent une transition et apprennent des modifications sur l'environnement et sur le nombre de places occupées à la fin de la cinquième simulation. Cela renforce également le fait qu'il est nécessaire de bien choisir le *tick* de simulation pour éviter que l'accumulation d'expériences soit trop importante.

En observant le résultat des simulations, le décideur répond à ses différents questionnements et calibre de

manière empirique le gestionnaire de simulations pour permettre un meilleur couplage de dynamiques. Il est possible que le décideur ait besoin de disposer des connaissances du concepteur pour correctement calibrer le gestionnaire de politiques à ses objectifs. Nous pouvons imaginer qu'en effectuant plusieurs simulations, le décideur va progressivement être sensibilité aux différentes notions du gestionnaire de simulations. Dans un cadre plus complexe, la présence d'un expert est toujours envisageable. Une fois le gestionnaire de politiques configuré, le décideur dispose désormais des outils pour construire une politique de tarification d'emplacements dans une ville.

7.3 Expérimentations

Nous avons choisi de découper le scénario précédent en trois expérimentations nous permettant de valider la pertinence du gestionnaire de politiques relativement à la bonne répartition des agents politiques locaux, pour assurer une politique globale qui maximise le gain du décideur (tableau 7.1). Les deux premières expérimentations reposent sur un environnement spécifique pour illustrer le fonctionnement d'une des deux actions de contrôle : fusion ou séparation des clusters. La troisième expérimentation présente un cadre plus complexe où l'agent de contrôle peut être amené à appliquer plusieurs actions de contrôle.

Expérimentation	Actions de contrôle testées	Nb agents de contrôle initial	Nb agents locaux
1	Fusion	1	17
2	Séparation	2	17
3	Fusion, Séparation	2	32

TABLE 7.1 – Les trois expérimentations sont construites autour du même scénario où la fonction objectif est de maximiser le gain

7.3.1 Paramètres du scénario

Pour compléter le simulateur, le décideur politique doit paramétrer la personnalité des agents de la couche microscopique et la couche macroscopique avec les méta-paramètres du CDQN.

Pour les besoins de ce scénario, les personnalités des agents sont construites de manière à exhiber des comportements différents.

Comme déjà mentionné, pour nos expérimentations, les agents humains sont sensibles uniquement aux tarifs, représentés par la tableau 7.2. Ainsi, les agents humains *normaux* vont accepter un tarif de 2€ alors que les agents humains *économes* ne stationneront jamais pour ce tarif.

Les profils de populations sont décrits par la tableau 7.3.

Le CDQN dispose d'autre part d'un ensemble de méta-paramètres, pouvant être calibrés par le décideur politique ou un expert accompagnateur en fonction du scénario.

Les méta-paramètres sont ici détaillés pour chaque scénario envisagé.
un seul paramètre est spécifié pour l'action de fusion des agents de contrôle :

Profil	Prix ($)		
	Meilleur	Pire	Poids
Normal	[0 :2.0]	[3.0 :4]	1
Économe	[0 :1.0]	[2.0 :4]	1

TABLE 7.2 – L'évaluation par les deux profils d'agents humains des prix des emplacements. Dans le cas où le prix se trouve dans le meilleur intervalle, l'utilité de l'agent pour s'y garer est de 1 alors qu'elle est de 0 dans le pire intervalle. L'utilité évolue linéairement entre les deux intervalles.

	Profils (%)	
Pop	Nor	Écon
p_1	100	0
p_2	0	100

TABLE 7.3 – Les deux populations p_1 et p_2 aux personnalités différentes utilisées pour les exemples de fusion et de séparation

— **Nombre de séquences similaires** : les séquences similaires sont des séquences d'actions non-ordonnées permettant à des clusters d'atteindre le plus haut gain identifié. Deux agents de contrôle peuvent choisir d'appliquer l'action de fusion (section 6.2.6) après le nombre spécifié d'itérations ayant des séquences similaires. Un nombre de 1 implique que les agents fusionnent dès qu'ils obtiennent une séquence similaire alors qu'un nombre plus élevé entraîne une meilleure stabilité des propositions effectuées par les agents locaux.

Les paramètres liés à l'action de séparation des agents de contrôle sont :

— **Taille de la zone interdite** : le score de confiance varie entre -1 et 1. La taille de la zone interdite détermine la stabilité attendue par la simulation pour appliquer l'action de séparation. Le décideur peut avoir une zone proche de 0 (stabilité faible) jusqu'à avoir ses bornes autour de -1 et 1, ce qui garantit que la solution est obligatoirement stable avant d'appliquer l'action de séparation ;

— **Taille du tampon** : elle correspond au nombre d'itérations pendant lesquelles le score de confiance ne doit pas entrer dans la zone interdite. Une taille de 1 signifie qu'à l'instant où les scores de confiance des agents locaux d'un cluster sont hors de la zone interdite, l'agent de contrôle effectue une action de séparation, alors qu'une taille plus importante va impliquer d'atteindre une certaine stabilité du score de confiance ;

— **Incrément du score de confiance** : le score de confiance varie initialement par incrément de 0.1 mais il est possible de déterminer un incrément plus grand ou plus faible, ainsi qu'une asymétrie entre les incréments positifs et négatifs.

Les paramètres liés aux actions de conservation ou d'annulation des actions de contrôle de fusion et de séparation :

— **Nombre de fois où un gain supérieur après action est atteint** : un compteur est initialisé lors

de l'expérimentation d'une action de contrôle et est incrémenté à chaque fois qu'un gain supérieur ou égal (strictement supérieur dans le cas d'une séparation) est atteint. Pendant une même simulation, un gain supérieur peut être atteint plusieurs fois. L'action de contrôle sera validée si un gain supérieur a été atteint un certain nombre de fois. Dans le cas contraire, les clusters reprennent leur configuration initiale. Il est donc nécessaire de choisir une valeur du nombre de fois où un gain supérieur est atteint, qui soit strictement inférieure au nombre d'itérations, puisque cela impliquerait par exemple qu'un gain maximal serait atteint dès le début de la simulation suivante ;

— **Nombre maximal d'itérations pour tester une action de contrôle** : il est aussi possible d'envisager un nombre d'itérations différent pour l'action de `Fusion` et `Separation`.

Il est important de fixer correctement ces deux paramètres pour obtenir un résultat pertinent : à titre d'exemple, un gain agrégé atteint un nombre de fois assez faible sur un grand nombre d'itérations ne garantit pas nécessairement une meilleure distribution des agents locaux dans les clusters. Le paramétrage de ces paramètres se fait actuellement de manière empirique.

Comme expliqué dans le chapitre sur CDQN, chaque agent local dispose de son propre réseau de neurones. Tous les réseaux de neurones sont entrainés en utilisant l'algorithme ADAM [130] et une taille de *mini-batches* (sous-ensembles) de 64 échantillons. Le *discount factor* est de 0.95. Durant l'apprentissage, le choix des actions est fait en suivant une politique ϵ-greedy où ϵ décroit de 1.0 à 0.01 (ϵ_{min}) à chaque fois que les agents proposent une action. L'agent de contrôle va donc forcer l'exploration en choisissant l'action la plus suggérée avec une probabilité de $1 - \epsilon$ et impose une action aléatoire avec une probabilité de ϵ. La mise à jour de ϵ est égale à $\epsilon' = \epsilon \times \epsilon_{decay}$ avec $\epsilon_{decay} = 0.995$. La taille de l'échantillon ainsi que les paramètres de l'ϵ-greedy (ϵ, ϵ_{decay}, ϵ_{min}) peuvent également être fixés par l'utilisateur de CDQN.

7.3.2 Mise en place de la fusion et de la séparation

Avant de construire notre scénario, les actions de contrôle du CDQN, c'est-à-dire la fusion et la séparation (expérimentation 1 et 2 respectivement), sont employées sur des exemples simplifiés.

Ces deux expérimentations permettent de comprendre et valider chaque action de contrôle séparément.

Considérons différents quartiers avec des personnalités d'agents variables (représentées par la couleur du quartier dans la figure 7.5 et la figure 7.6).

Pour valider l'action de fusion, le premier exemple utilise deux quartiers q_1 et q_2 avec des agents possédant une unique personnalité p_1 (figure 7.5). Ici l'agent de contrôle dispose uniquement de l'action de contrôle `Fusion` et de ne rien faire.

Le second exemple sert à valider l'action de `Separation` en considérant cette fois deux personnalités différentes. Cette expérimentation considère deux quartiers et deux personnalités différentes : le premier (resp. second) quartier q_1 (resp. q_2) possède les agents avec la personnalité p_1 (resp. p_2) (figure 7.6).

L'agent de contrôle d_1^c dispose des actions `Separation` et ne rien faire.

Les deux exemples sont implémentés avec Repast Simphony (figure 7.7) pour deux quartiers de Los Angeles. Le quartier nord (q_1) dispose de 9 fronts de rue (pour un total de 72 emplacements disponibles), soit 9 agents politiques locaux ; le quartier sud (q_2) dispose de 8 fronts de rue (pour un total de 64 emplacements

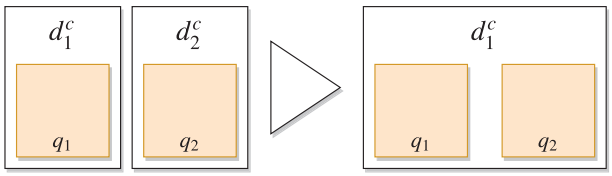


FIGURE 7.5 – Description d'une action de fusion avec un état initial disposant de deux clusters alors que les deux quartiers q_1 et q_2 ont des agents aux personnalités identiques. L'objectif pour ce scénario est d'avoir une fusion des clusters d_1^c et d_2^c en un cluster d_1^c pour valider l'objectif de minimisation du nombre de clusters.

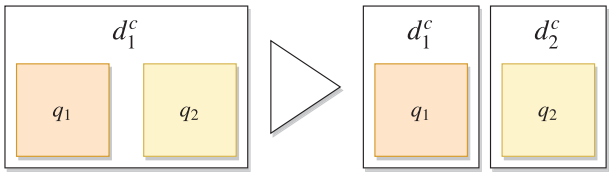


FIGURE 7.6 – Description d'une action de `Separation` avec un état initial omposé d'un seul cluster. L'objectif pour ce scénario est d'avoir une séparation du cluster d_1^c en deux clusters d_1^c et d_2^c.

disponibles). Dans le cadre de nos expérimentations, qui se limitent à l'évolution des clusters, les quartiers sont séparés : un agent apparaissant sur q_1 ne peut pas aller à q_2 puisqu'il n'existe pas de route joignant les deux quartiers. q_1 peut accueillir 90 agents humains et q_2 80.

FIGURE 7.7 – Implémentation des scénarios pour la fusion et la séparation avec Repast Simphony et Open Street Map comme fond de carte. Les polygones jaunes représentent les lieux de travail S_{bat}, les carrés verts représentent les emplacements de stationnement S_{emp}.

7.4 Résultats

Cette section introduit les résultats obtenus avec l'algorithme CDQN sur les expérimentations de fusion uniquement (1) et de séparation uniquement (2) dans un premier temps puis avec l'expérimentation complète (3) dans un second temps.

7.4.1 Validation des actions de `Fusion` et de `Separation` séparément

Pour évaluer ces deux expérimentations, les résultats sont comparés avec des Independant Deep Q-Learners [231] (IDQL) (*Agents apprenants indépendants profonds*). Ils sont paramétrés avec le même réseau que les agents locaux. Le fonctionnement des IDQL revient à avoir un CDQN sans agents de contrôle.

Pour les IDQL, le gain cumulé global correspond à la somme des gains locaux obtenus par les agents indépendants à la fin d'une séquence de simulations pendant une série de séquences. C'est équivalent à la somme des gains des agents de contrôle dans le cadre du CDQN.

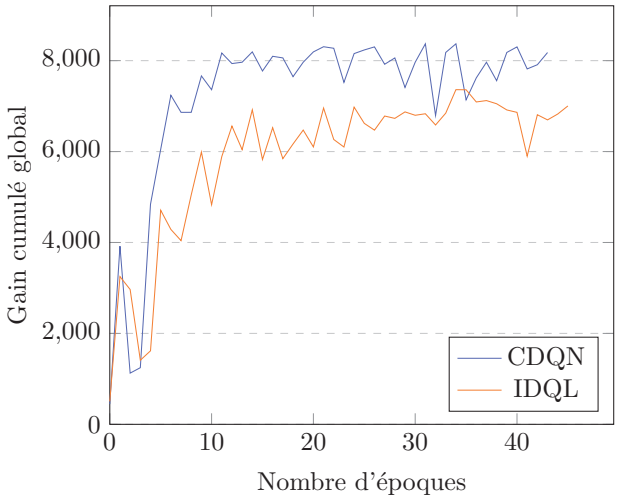


FIGURE 7.8 – Gain cumulé global pour l'action de `Fusion` (expérimentation 1)

La figure 7.8 montre que le CDQN obtient un meilleur gain global cumulé que les IDQL à l'aide de la configuration initiale des clusters. Après un certain nombre d'itérations, les agents locaux proposent des actions qui améliorent le gain global. Quand les deux agents de contrôle ont un nombre de séquences d'actions identiques égales à la valeur du méta-paramètre spécifiée, ils fusionnent (ici à l'épisode 5). Cette configuration de cluster est conservée après l'évaluation de l'action de fusion, ce qui révèle que l'action est pertinente. En étudiant la politique appliquée sur ce scénario, nous constatons que le meilleur gain est obtenu en proposant d'appliquer le même prix pour tous les emplacements, mais que le prix dépend de la population utilisée (2€ pour p_1 et 1€ pour p_2). L'analyse fine du graphique montre aussi qu'en réduisant l'impact de la non-stationnarité, notamment grâce aux agents locaux d'un même cluster qui effectuent la même action, la

vitesse d'apprentissage est accrue, et permet d'atteindre plus rapidement une politique individuelle optimale que dans le cas d'agents indépendants avec IDQL.

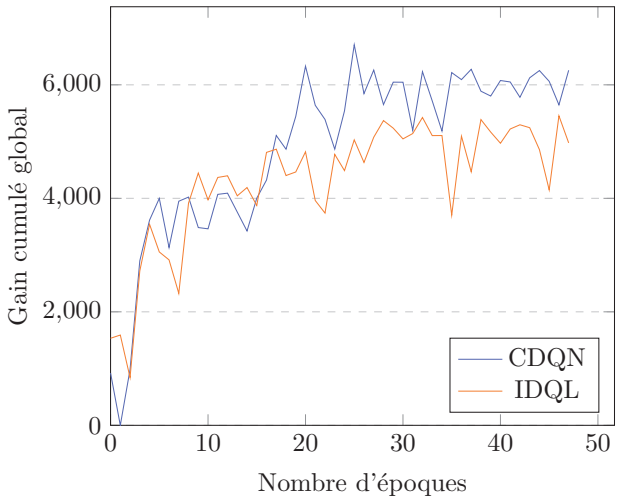


FIGURE 7.9 – Gain cumulé global pour l'action de `Separation` (expérimentation 2)

Au début de cet exemple, CDQN et IDQL explorent l'environnement (figure 7.9). Avant 15 époques, le cluster identifie un groupe d'agents locaux qui contribue davantage à l'amélioration du gain global et il modifie les scores de confiance en conséquence. Ce phénomène est visible avec l'évolution du score de confiance (figure 7.10).

Dans cette configuration, les IDQL obtiennent de meilleures performances que le CDQN dans les 15 premières époques car chaque agent indépendant choisit ses propres actions alors que seule la moitié des agents du CDQN voient l'action qu'ils ont suggérée être effectuée. Dans le cadre du CDQN, certains agents locaux apprennent que les actions appliquées par leur agent de contrôle d_1^c n'améliorent pas leur gain local alors que d'autres agents locaux ont des actions augmentant leur gain local. La particularité de ce scénario est d'avoir les deux personnalités p_1 et p_2 gérées par les agents politiques du même cluster. Le gain d'un agent local présent dans le quartier q_1 est maximal lorsque tous les emplacements sont disponibles et que le tarif est de 2€. Alors qu'un agent local présent dans le quartier q_2 obtient un gain maximal lorsque le tarif est de 1€. La personnalité p_1 étant représentée en majorité dans cet exemple, le gain global est plus important quand la politique du cluster est d'avoir des emplacements à 2€ pour lesquels seul la personnalité p_1 accepte de stationner. Par conséquent, les agents locaux proposant des actions amenant les tarifs autour de 2€ auront leur score de confiance augmenté, alors que les agents locaux proposant de réduire les tarifs à 1€ auront leur score de confiance diminué. Un écart va alors se créer entre le score de confiance des agents du quartier q_1 et ceux du quartier q_2.

Lorsque les conditions nécessaires pour effectuer une séparation sont réunies (liées à la taille de la zone interdite et au nombre d'itérations sans franchir la zone interdite), l'agent de contrôle conserve les agents

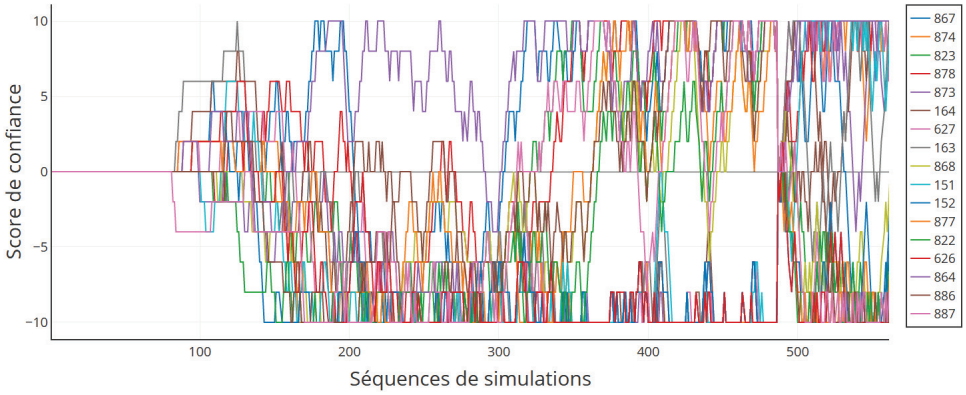


FIGURE 7.10 – Évolution du score de confiance sur les 500 premières séquences de simulations (soit environ 15 époques). Lors des 80 premières séquences, les agents locaux explorent et proposent des actions aléatoires, ce qui ne modifie pas leur score de confiance. Au cours de l'apprentissage, certains agents améliorent le gain global, ce qui augmente leur score de confiance et certains agents le réduisent, baissant ainsi leur score de confiance. Vers la séquence 480, l'agent de contrôle possède des scores de confiances stables et est en mesure d'effectuer une séparation, visible par la remise à 0 du score de confiance de chaque agent local présent dans le cluster.

locaux avec un score de confiance positif et crée un nouvel agent de contrôle avec le reste des agents locaux. À partir de cette séparation, les agents de contrôle peuvent désormais appliquer des actions environnementales différentes à chaque itération. L'impact de l'action de séparation à la quinzième époque est immédiat sur le gain global puisque les agents locaux sont désormais tous en mesure de proposer des actions pertinentes pour leur cluster. Ainsi, l'algorithme de CDQN obtient de meilleurs résultats que les IDQL grâce à l'apprentissage effectué sur un cluster plus homogène. La somme des gains de chaque agent de contrôle est supérieure au gain de l'agent de contrôle avant l'application de l'action de séparation et, par conséquent, l'action est considérée comme pertinente. Les agents locaux présents dans les clusters suggèrent des actions augmentant leur score de confiance et, après un certain nombre d'itérations, le système se stabilise. La politique urbaine résultante de cet exemple est le découpage de deux régions (ici, les deux quartiers) avec un quartier où les tarifs sont de 2€ (correspondant au tarif maximal que les agents avec la personnalité p_1 sont prêt à payer) et un quartier où les tarifs sont à 1€(tarif maximal pour les agents avec la personnalité p_2).

Observons maintenant comme la combinaison des deux actions de contrôle permet de produire une politique urbaine complète.

7.4.2 Scénario complet

Considérons un scénario où l'environnement contient 4 quartiers q_1, q_2, q_3 et q_4, les deux précédentes personnalités p_1 et p_2 et deux agents de contrôle (figure 7.11).

Les quartiers q_1 et q_2 ont la même structure que dans les deux exemples précédents. Le quartier q_3 dispose de 5 fronts de rue (pour un total de 57 emplacements disponibles) et peut accueillir 70 agents humains et le quartier q_4 dispose de 10 fronts de rue (pour un total de 81 emplacements disponibles) et peut accueillir

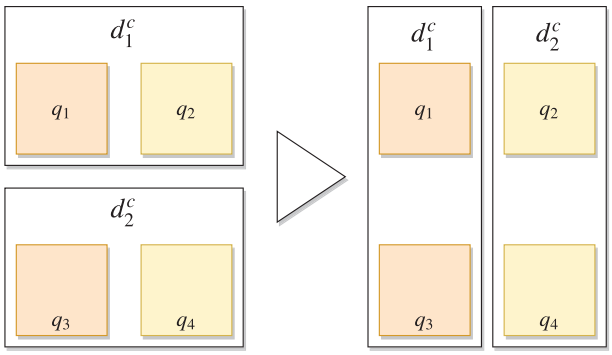


FIGURE 7.11 – Combinaison de la fusion et de la séparation où l'état initial dispose de deux clusters chacun ayant deux quartiers de populations différentes p_1 et p_2. L'objectif pour ce scénario est de correctement découper les clusters afin que les clusters se retrouvent chacun avec deux quartiers de personnalités identiques. Cela permet d'une part de réduire le nombre de clusters et d'augmenter le gain global de la simulation.

100 agents humains. Ce scénario contient un total de 32 agents locaux et 274 emplacements disponibles (figure 7.12).

FIGURE 7.12 – Les 4 quartiers du scénario

L'algorithme de CDQN commence par une exploration de l'environnement avant de fusionner (figure 7.13) vers l'époque 10. La démarche pour la séparation est similaire à celle du l'exemple de séparation précédent où seul une partie des agents locaux permettent d'augmenter le gain global, ce qui n'empêche pas les autres agents politiques d'accumuler de l'expérience. Le saut à l'époque 15 correspond à l'agent de contrôle du CDQN qui se sépare en deux agents de contrôle, permettant d'augmenter le gain cumulé global. Les agents indépendants obtiennent un gain cumulé global plus faible que celui du CDQN dans cette configuration, car cette dernière requiert plus de coordination entre les agents que dans les scénarios précédents, et les actions augmentent peu le gain cumulé global. La politique urbaine proposée dans cette configuration est

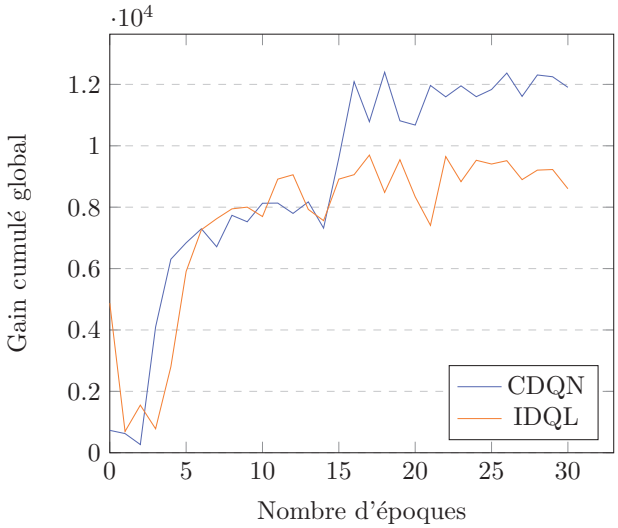


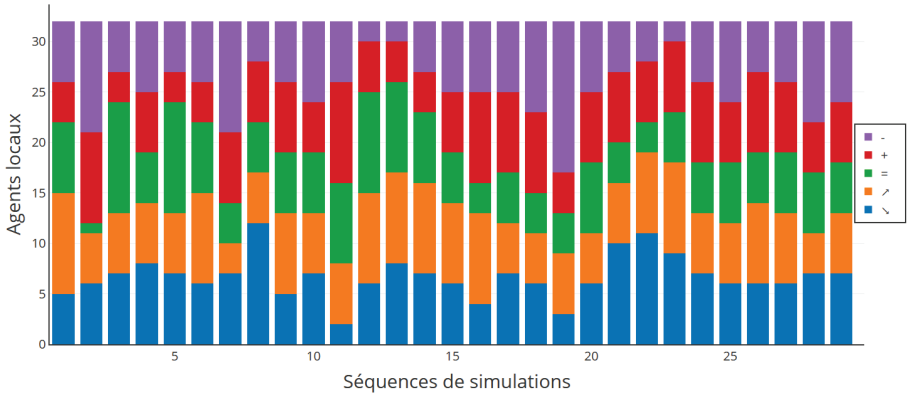
FIGURE 7.13 – Gain cumulé global pour le scénario avec les actions de contrôle `Fusion` et `Separation` (expérimentation 3)

un découpage de l'environnement où un agent de contrôle d_1^c possède les quartiers q_1 et q_3 et un agent de contrôle d_2^c possède les quartiers q_2 et q_4 avec des tarifs correspondant aux personnalités des quartiers, soit 2€ pour les fronts de rue du cluster d_1^c et 1€ pour les fronts de rue du cluster d_2^c.

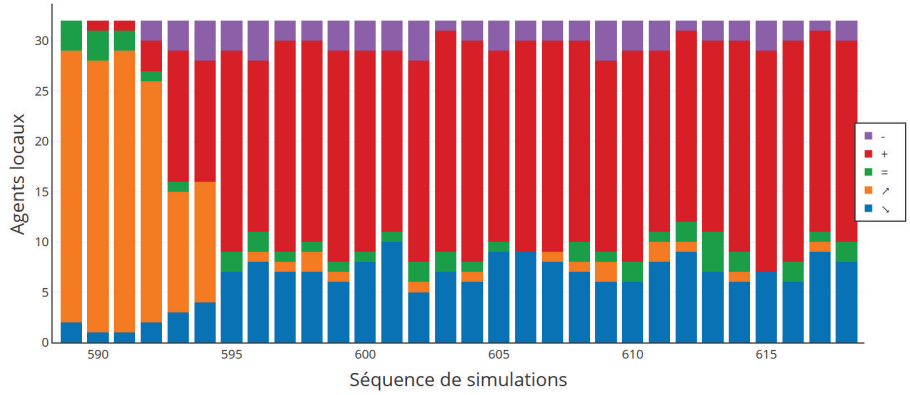
Il est possible de voir comment les agents locaux apprennent en consultant les actions suggérées à la fin de chaque séquence de simulations en fonction de leurs perceptions (figure 7.14). Au début de la simulation, les agents locaux apprennent et choisissent aléatoirement leurs actions (figure 7.14a). L'agent de contrôle applique également des actions choisies aléatoirement avant de prendre l'action la plus suggérée par ses agents locaux. Après chaque épisode (une trentaine de séquences de simulations), l'environnement réinitialise les tarifs et les emplacements disponibles afin de forcer les agents locaux à être confrontés à de nouveaux états et apprendre la meilleure politique urbaine. La figure 7.14b représente un état initial où les prix sont bas (0.5€). Ici, les agents locaux ont suffisamment appris pour tous proposer l'action ↗ permettant d'augmenter les prix pour maximiser leurs gains locaux et le gain global. Les actions proposées ensuite n'ont pas d'impact sur l'environnement, en effet, ils proposent d'augmenter le nombre d'emplacements, qui est déjà au maximum, ce qui revient à ne pas modifier l'environnement.

7.5 Discussion et perspectives

Les expérimentations réalisées dans ce chapitre montre comment la combinaison de l'approche CDQN avec le modèle SmartGov permet de produire des politiques urbaines à partir de personnalités existantes. Ces expérimentations décrivent un contexte simplifié où l'aspect co-construction est limité à la construction



(a) Démarrage de la simulation



(b) Après une fusion et une séparation

FIGURE 7.14 – Les actions suggérées par les agents locaux à leur agent de contrôle à la fin de chaque séquence de simulations. (a) représente le début de la simulation lorsque les agents explorent, les actions suggérées ici sont déterminées aléatoirement. (b) correspond à un épisode où les agents ont appris les spécificités de l'environnement. Aux itérations 589 à 591, les apprenants suggèrent ensemble d'augmenter les prix. Le système se stabilise lorsque les agents locaux suggèrent en majorité une action n'ayant pas d'impact sur la simulation (équivalent à ne rien faire).

initiale, par le décideur, de la couche microscopique. Les deux premiers scénarios utilisent un nombre restreint d'actions de contrôle pour valider leur utilité dans l'amélioration des configurations de clusters. Le troisième scénario combine les différentes actions de contrôle pour montrer l'intérêt du CDQN, par rapport à l'IDQL, pour permettre un apprentissage décentralisé dans un cadre non-stationnaire. Dans cette section, nous nous intéressons aux différentes composantes de SmartGov, notamment la co-construction des politiques urbaines et de l'apport du CDQN comme méthode d'apprentissage dans notre contexte.

7.5.1 Validation de la co-construction de politiques urbaines

L'objectif de ce travail de thèse était de proposer une architecture générique pour la co-construction de politiques urbaines. Cependant, les expérimentations que nous avons mises en place au long de ce chapitre ne permettent pas de valider cet aspect. Dans cette section, nous nous intéressons à ce que représente, pour nous, la co-construction présente dans cette thèse et notre positionnement sur cette problématique dans l'ensemble de ce travail de thèse.

La validation de la co-construction nécessite plusieurs étapes distinctes :

— Dans un premier temps, la co-construction doit être définie de manière formelle afin de cadrer les interactions qui auront lieu entre un outil d'aide à la décision, ici le modèle générique SmartGov, et les décideurs politiques. Dans la section 4.8 et la section 4.6, nous avons décrit d'un point de vue théorique les interactions qui seraient possibles et la manière dont celles-ci seraient réalisées. Ce modèle formel doit ensuite être validé pour garantir que la co-construction est envisageable. Cependant, valider un modèle formel générique requiert d'en posséder une instance concrète.

— Ainsi, dans un second temps, il est nécessaire de produire cette instance du modèle formel pour disposer d'un support sur lequel le décideur pourra interagir et plus tard co-construire des politiques. Nous avons développé une instance de Smartgov, qui a pu être utilisée par plusieurs stagiaires provenant de masters en informatique. Ces stagiaires possédaient cependant un important bagage en informatique, et il nous resterait donc à vérifier que notre outil peut être utilisé par des décideurs sans connaissances dans ce domaine.

— Enfin, dans un troisième temps, il faut solliciter des décideurs sensibles à ces problématiques de co-construction. Il est ensuite requis d'établir un protocole expérimental pour observer s'il leur est possible de se servir de l'outil sans assistance et si cet outil correspond bien à leurs besoins.

Pour valider cet aspect de co-construction, nous sommes confrontés à une différence entre l'aspect théorique et fondamental du modèle formel tel que présenté au chapitre 4, et l'aspect ingénierie de l'instance. En effet, pour valider notre modèle SmartGov, nous avons besoin de passer par une instance concrète. Cependant, la mise en place d'une instance implique des contraintes d'ingénierie, qui sont liées par exemple au choix de la plateforme ou au langage de programmation utilisé. Par conséquent, ces contraintes peuvent représenter des limites à la validation du modèle et donc à la validation de la co-construction.

La vision idéale que nous avons de l'instance SmartGov est un outil où la construction de l'environnement est simplifiée de manière à ce que le décideur puisse construire n'importe quel environnement facilement, c'est-à-dire sans connaissances expertes préalables. L'avantage de l'instance est donc de réaliser une passerelle entre une implémentation bas niveau d'un code informatique et les entrées que le décideur peut fournir pour construire son environnement. Dans l'expérimentation que nous proposons, l'environnement est constitué d'un réseau routier que le décideur peut facilement extraire d'Open Street Map et le fournir à Smart Gov, il dispose ainsi d'un graphe représentant le réseau routier sur lequel les agents microscopiques vont évoluer. Le décideur doit ensuite décrire les emplacements de stationnement. Même s'il dispose d'un fichier avec les attributs de ces emplacements, comme les coordonnées géographiques et leur ID, il est toujours nécessaire de transcrire ce fichier en structures exploitables par l'instance. Ainsi, même si une partie du code est déjà

réalisé, et que le système détecte et traite les informations géographiques par exemple, il reste toujours des attributs qui requièrent un développement spécifique. C'est une tâche que le décideur, sans connaissances en informatiques, ne peut pas faire seul et pour laquelle il a besoin d'un intermédiaire, un développeur par exemple. De la même manière, la construction de l'automate, décrit de manière à pouvoir se déplacer et à interagir dans l'environnement avec un ensemble de descripteurs, requiert encore des connaissances expertes. Même si le décideur peut facilement envisager les états, la compréhension des interactions sous-jacentes n'est pas directement évidente.

Par conséquent, la validation de SmartGov en tant qu'outil d'aide à la décision est une tâche encore complexe, et il est difficile de se positionner sur ce point. Le CDQN est conçu comme un algorithme autonome, capable de s'adapter à n'importe quelle représentation de l'environnement. Il ne requiert donc a priori aucune connaissance de la part du décideur, qui peut directement disposer d'une couche composée d'agents apprenants indépendants. Cependant, l'ajustement des méta-paramètres nécessite des connaissances acquises empiriquement en manipulant le système. En effet, l'impact de ces méta-paramètres influent sur la vitesse de convergence, d'apprentissage et la qualité de la solution produite. La démonstration de la co-construction dans le cadre de cette thèse porte donc uniquement sur le premier aspect sur les trois envisagés (co-construction avant la simulation pour construire l'environnement ; pendant la simulation pour effectuer des modifications ; après la simulation pour modifier la politique construite), les deux autres ayant été considérés formellement mais pas représentés dans l'instance de SmartGov. Par conséquent, la co-construction, bien qu'au cœur de ce travail de thèse, est uniquement présentée de manière générique en imaginant les interactions que le décideur aurait avec SmartGov et qui ont été exprimées dans aux section 4.6, section 4.4.3 et section 7.2. Approfondir l'aspect co-construction est par conséquent une de nos perspectives de thèse. Pour cela, il nous faudrait valider (i) le fait que le décideur puisse utiliser facilement l'instance Smartgov sans connaissances a priori (seconde étape, voir ci-dessus) et (ii) un protocole expérimental permettant de tester la co-construction. Ces deux étapes nous permettraient ainsi de valider le modèle de co-construction générique présenté à la section 4.8.

7.5.2 Algorithme Clustered Deep Q-Network

Les expérimentations proposées dans ce chapitre portent davantage sur la validation du CDQN en tant que modèle (pour doter le gestionnaire de politiques d'une intelligence artificielle distribuée pour la résolution de politique) que la validation du modèle SmartGov. Les trois expérimentations de ce chapitre permettent une résolution, en se comparant à l'IDQL, des verrous identifiés dans le chapitre précédent. Le premier verrou est la non-stationnarité qui impacte négativement la réutilisation d'expériences pour l'apprentissage par renforcement profond. L'action unique que les apprenants d'un même cluster appliquent diminue l'impact de la non-stationnarité, comme nous pouvons le voir sur les résultats des expérimentations. Le gain cumulé global est atteint plus rapidement et est plus élevé que lorsque les agents sont impactés par un apprentissage concurrent. Le second verrou, celui de l'identification de la contribution de chaque agent, est résolu avec le score de confiance qui est basé sur la différence entre la proposition de l'action et l'action appliquée. L'agent de contrôle, sans connaitre directement la contribution individuelle de l'agent, peut identifier les différences dans l'apprentissage de chaque agent avec les actions suggérées. Le score de confiance, utilisé

pour effectuer l'action de contrôle `Separation`, permet également d'atteindre un nouveau gain global plus élevé qu'avant la séparation. Cette évolution du gain maximal obtenu, également connu sous le nom de problème de la cible mouvante, est également adressé avec l'algorithme CDQN. En réduisant l'impact de la non-stationnarité, notamment grâce aux agents locaux d'un même cluster qui effectuent la même action, la vitesse d'apprentissage est accrue et permet d'atteindre plus rapidement une politique individuelle optimale que dans le cas d'agents indépendants.

Il serait cependant intéressant de confronter l'algorithme CDQN à d'autres approches de la littérature pour comparer la vitesse à laquelle l'état optimal est atteint, ou la stabilité de la solution proposée.

Distribution initiale des clusters

Le CDQN, utilisé dans le cadre de SmartGov, doit s'adapter à n'importe quel type d'environnement. Cependant, nous pouvons envisager des cas extrêmes où certaines configurations seront plus lentes que d'autres à converger. Le décideur impacte donc l'apprentissage initial avec un découpage expert, comme réalisé dans les trois expérimentations que nous proposons dans ce chapitre. En effet, dans le cas où chaque cluster dispose d'un unique agent apprenant, nous retrouvons une configuration identique à celle de l'IDQL qui, comme observé dans la partie résultats, apprend moins efficacement qu'avec une configuration initiale plus pertinente. Il serait intéressant d'observer dans ce cas comment les agents de contrôle utilisent leurs actions pour identifier une configuration pertinente.

Afin de prouver la robustesse de notre approche, il serait intéressant d'observer les cas extrêmes suivants (dans lesquels le CDQN serait susceptible d'être moins performant) :

- un cas où chaque agent de contrôle possède un unique agent apprenant (similaire à l'IDQL) ;
- un cas où le découpage initial est aléatoire, et le nombre d'apprenants dans chaque cluster l'est également ;
- similaire au cas précédent, il est également possible d'envisager un découpage initial aléatoire où les clusters ne sont pas des zones géographiques contigües (contrairement au découpage en zones adjacentes que nous avons utilisé, car cela diminuerait la pertinence des clusters).

La simulation avec SmartGov est plus longue que d'autres applications classiques de DQN comme les jeux Atari par exemple, et la taille de la mémoire à donc de l'importance. Les travaux de Mnih *et al.* [168] considèrent 3000 expériences stockées en mémoire avant de supprimer les plus anciennes expériences, ce qui est relativement rapide. Dans le cadre de SmartGov, 3000 expériences correspondent à 3000 séquences, soit plusieurs jours de simulations. Dans notre modèle, la suppression des plus mauvaises expériences n'est donc pas effectuée avant un certain temps, ce qui pourrait impacter négativement le CDQN.

Les choix réalisés pour le CDQN sont d'utiliser deux approches simples de la littérature. Tout d'abord, nous avons utilisé l'algorithme de Deep Q-Network, qui est l'algorithme le plus simple de l'apprentissage par renforcement profond (qui fournit de très bons résultats). D'autre part, nous avons mis en œuvre une réutilisation naïve d'expériences, par opposition à une meilleure gestion de la mémoire qui supprimerait plus rapidement les mauvaises expériences. Nous montrons donc que même nos choix d'algorithmes simples (DQN), nous obtenons de bons résultats avec l'utilisation d'une méthode multi-niveaux, et nous adressons ainsi des verrous de la communauté. Il est tout à fait possible de conserver l'idée du CDQN en changeant

l'algorithme d'apprentissage et le choix de la réutilisation d'expérience (taille plus petite de la mémoire, suppression dynamique de mauvaises expériences par exemple).

Le découpage permis par le CDQN permet donc de se détacher de certains découpages existants, trop sectaires ou trop historiques, qui n'ont plus de sens avec le récent étalement urbain ou l'évolution des quartiers.

7.5.3 Temps de simulation

Nous considérons deux aspects dans le temps de simulation. D'une part le temps nécessaire pour effectuer une simulation et de l'autre le temps requis pour produire une politique urbaine.

Dans le premier cas, le temps de simulation d'un scénario dépend du nombre d'agents de la couche microscopique, de leur modèle cognitif et de la taille de l'environnement. Bien évidemment plus l'environnement est complexe, plus le temps pour générer un tick de simulation sera grand. Cela a donc un impact direct sur l'apprentissage des agents puisque l'acquisition d'expérience est plus longue. Par contre le nombre d'agents politiques locaux, en lien avec les structures de l'environnement, a peu d'impact sur le temps de simulation et le découpage en clusters permet en final de gérer aisément un grand nombre d'agents locaux. Cependant, plus l'espace d'actions des agents locaux et plus l'espace d'états est grand, plus l'apprentissage prendra du temps, notamment parce que le nombre d'états a un impact sur le temps nécessaire à l'exploration.

Dans la version actuelle de SmartGov, le décideur politique est donc limité par la complexité des environnements qu'il peut produire puisque ceux-ci peuvent prendre trop de temps à simuler. Le décideur peut décider d'agréger son modèle pour simplifier l'environnement qu'il souhaite représenter. Cependant, l'outil doit permettre au décideur de représenter sa vision de la zone urbaine en limitant les hypothèses de modélisations qu'il doit faire. Plusieurs solutions sont envisagées pour résoudre les problèmes de temps de calcul en simulation : la programmation sur cartes graphiques (*General-Purpose Computation on Graphics Hardware*), la distribution de la simulation sur des clusters de calcul (parallélisme de la simulation), sauvegarde de morceaux de simulations pour les réutiliser, etc. Dans le second cas, le temps requis pour produire la politique repose en partie sur la bonne configuration des meta-paramètres ou encore la description itérative de l'environnement en fonction des nouveaux besoins par exemple.

Des extensions sur l'instance de SmartGov sont actuellement en cours pour permettre de gérer plusieurs vitesses de simulations différentes. Il est également possible d'enregistrer des simulations pour pouvoir les rejouer par la suite, dans un objectif de démonstration par exemple.

7.6 Conclusion

Ce chapitre est un exemple d'usage de SmartGov où le décideur interagit avec l'outil pour construire des politiques urbaines. Ici, l'interaction est représentée par la construction de l'environnement et des agents microscopiques par le décideur. Le scénario décrit dans ce chapitre expérimentation est une des nombreuses applications de SmartGov. Au cours de cette thèse, plusieurs stagiaires ont utilisé l'outil SmartGov pour construire des environnements, pour tester des simulations microscopiques de déplacements [199] ou encore

étudier l'impact de la régulation des véhicules de transport de marchandises dans le centre-ville sur la pollution locale, et de construire des politiques urbaines en conséquence [215].

L'expérimentation que nous présentons considère que les emplacements d'un front de rue sont connectés puisqu'ils transmettent des informations à leur front de rue et obtiennent des informations sur la satisfaction des usagers. Avec la SmartCity et la disponibilité des données sur les usagers, il sera possible d'envisager des emplacements de stationnement dynamique en voirie (par l'intermédiaire d'un marquage au sol par exemple) ; des tarifications ajustables en continu pour les places inoccupées (incitation des usagers à venir se garer par exemple) ; un parc d'emplacement de taille ajustable selon les places offertes par des particuliers (entreprises ou régies immobilières donnant accès à des parkings privés contre d'autres services) ; un guidage dans la zone urbaine vers des emplacements libres avec des applications dédiées.

L'avis des usagers pourrait ainsi être intégré de manière continue, afin par exemple de diminuer les tarifications trop fluctuantes si celles-ci ont un impact négatif sur les usagers.

La prochaine étape dans le développement de SmartGov serait de tester l'outil avec des décideurs politiques et observer si les politiques peuvent être appliquées sur une zone urbaine réelle. Nous envisageons également de considérer des scénarios de plus grande envergure pour observer l'impact de la croissance du nombre d'agents microscopiques et d'agents politiques sur les performances, et ainsi vérifier la robustesse du passage à l'échelle. Enfin, les travaux futurs portent sur le développement de l'interaction entre le décideur et SmartGov pour tester la conception en continu de politiques urbaines avec la possibilité pour le décideur de réajuster les politiques proposées par le système dans un mécanisme d'apprentissage réciproque, le décideur communiquant ses connaissances au système, SmartGov découvrant certains ajustements judicieux des politiques.

Chapitre 8

Conclusion et perspectives

8.1 Conclusion

La Smart City est à nos portes et la volonté d'intégrer intelligemment la quantité de données maintenant disponible : sur l'infrastructure ; les nouveaux usages qui en sont faits ; et leurs impacts sur la zone urbaine, est un objectif des décideurs depuis déjà plusieurs années. Avec la mise à disposition de données parfois en temps réel, il sera nécessaire d'adapter dynamiquement les politiques urbaines pour répondre au mieux aux difficultés rencontrées, qui risquent d'aller croissantes avec l'afflux régulier de populations dans les métropoles, tel qu'envisagé par les spécialistes. Il est ainsi facilement envisageable que d'ici quelques années, tous les acteurs de la ville, et notamment le citoyen, seront sollicités pour participer à la construction de politiques urbaines, où leurs besoins et leurs retours seront considérés pour réorienter les politiques en construction ou déjà mises en œuvre. C'est pourquoi le décideur politique, confronté à l'apport massif de toutes ces données, a besoin d'outils facilitant l'intégration et la prise en compte de ces retours. Face à cette problématique, le travail de thèse avait pour objectif de présenter une preuve de concept d'un outil d'aide à la décision autonome permettant de produire des politiques urbaines dans la ville de demain.

Pour construire un outil d'aide à la décision adapté, nous avons envisagé une architecture à deux niveaux, indépendants, et nous avons décrit de manière formelle d'une part comment le décideur pouvait exploiter l'outil et d'autre part comment les couches interagissent entre elles. Une force du modèle est son caractère générique : l'infrastructure ciblée provient de n'importe quelles données issues de plateformes collaboratives (par exemple Open Street Map), et les politiques urbaines sont définies selon un modèle générique assez facilement instanciable. Une fois configurée et instanciée, notre architecture propose un système de simulations avec des représentations visuelles aisément ajustables par les acteurs.

Dans ce mémoire de thèse, nous avons donc présenté SmartGov, la première contribution de ce travail, sous la forme d'une architecture générique multi-agents multi-niveaux, dédiée à la co-construction dynamique de politiques urbaines, dans le cadre de la Smart City de demain.

L'architecture SmartGov décrit formellement l'ensemble des composantes nécessaire pour mettre en place

deux simulations multi-agents représentant des systèmes complexes réalistes. Nous pouvons envisager que notre d'outil permettra aussi d'améliorer la transparence de la construction de politiques urbaines, avec des orientations politiques rendues visibles par la définition des attentes et des objectifs fournis en entrée. La première simulation, appelée couche microscopique, comprend l'environnement et ses composantes (structures, perceptions, actions disponibles) ainsi que les agents qui y évoluent et interagissent avec l'environnement. La seconde simulation, appelée couche macroscopique, représente l'aspect décisionnel lié à la construction de politiques, par l'intermédiaire d'un système d'apprentissage capable de s'adapter aux différents environnements de façon dynamique. L'une des particularités de SmartGov est le couplage de dynamiques entre ces deux couches permettant la co-construction des politiques urbaines à partir des données de la ville. Aucune connaissance initiale n'est nécessaire, le système est capable d'apprendre à partir de nouveaux environnements inconnus. Toutefois l'apprentissage est plus rapide lorsque le décideur donne des informations initiales pertinentes, par exemple sur le découpage géographique des agents politiques en clusters.

Pour illustrer comment l'outil de co-construction de politiques urbaines peut être exploité par les décideurs, nous avons fourni le processus complet d'instanciation du modèle SmartGov avec un scénario lié à la construction d'une politique tarifaire d'emplacements de stationnement.

Notre objectif était de doter l'outil de la capacité à prendre des actions politiques de manière autonome et pouvoir ainsi s'adapter à n'importe quelle évolution de la zone urbaine. Nous avons dans ce but décrit notre seconde contribution, l'algorithme *Clustered Deep Q-Network* (CDQN) qui propose un modèle d'apprentissage autonome et adaptatif, indépendant de SmartGov, par renforcement multi-agents profond, qui par une répartition pertinente des agents politiques sur le périmètre d'étude, permet d'affiner la régulation de la politique par rapport à un objectif recherché. Cet algorithme d'apprentissage conduit à un processus décisionnel où différentes variantes des politiques urbaines sont suggérées au décideur, en fonction des données disponibles de la Smart City et sur lesquelles le décideur a la possibilité d'ajuster ou de valider certaines actions, dans une boucle de co-construction dynamique.

Le premier niveau de la couche décisionnelle se compose d'agents apprenants, qui sont distribués sur la zone urbaine, qui n'ont pas de moyen de communication entre eux, et qui apprennent les spécificités locales de l'environnement. Ce découpage à des zones restreintes permet un apprentissage rapide, les espaces d'exploration étant limité. Le second niveau permet de réaliser un découpage géographique de la politique urbaine en introduisant des clusters regroupant des agents du premier niveau en suivant une configuration maximisant les objectifs des décideurs. Le CDQN repose sur le modèle formel des Factored Dec-POMDPs et décrit l'interaction entre les agents apprenants et leur cluster. Cette contribution est également un apport aux techniques d'apprentissage par renforcement multi-agents avec une manière de coordonner des agents sans communication entre eux et de permettre de réduire l'impact des environnements non-stationnaires sur la réutilisation d'expériences. Avec l'usage de récompenses individuelles pour les apprenants et une attribution d'un score de confiance, le CDQN construit des configurations de clusters pertinentes et s'intéresse à l'identification de la contribution des agents sur les performances du système dans un environnement décentralisé.

Ce travail prospectif est une preuve de concept d'une approche de couplage de dynamiques pour la co-

construction de politiques urbaines dans des systèmes complexes réalistes intégrant un modèle d'apprentissage autonome et adaptatif. L'architecture SmartGov est implémentée en utilisant la description formelle SmartGov ainsi que l'algorithme CDQN pour la couche macroscopique. Ces deux contributions ont donc été combinées pour comparer la co-construction de politiques urbaines sur un exemple de tarification d'emplacements de stationnement. Notre expérimentation a permis de montrer que CDQN permettait de mieux ajuster la configuration des agents politiques sur le périmètre d'étude, en fonction d'environnements spécifiques.

Dans cet exemple, nous n'avons pas illustré la participation du décideur aux actions suggérées par le système. Mais dans une utilisation plus complète et réaliste, et bien que SmartGov exploite ce module d'apprentissage autonome et adaptatif, les politiques co-construites resteront validées par le décideur politique qui garde la main sur la politique (valider ou pas une politique suggérée), et qui peut ajuster en dynamique ses attentes ainsi que sa fonction objectif.

8.2 Perspectives

Nous envisageons trois perspectives portant sur le modèle SmartGov :

1. **Co-construction et échanges avec le décideur :** la co-construction est formalisée dans notre outil et peut intervenir à trois moments distincts dans la boucle de conception d'une politique urbaine : au tout début, pour élaborer la couche microscopique, avec la simulation de la zone urbaine ; lors de la conception de politique, pour autoriser ou interdire une action préconisée par le système, ou modifier les structures ; après la proposition d'une politique, pour évaluer ou modifier la séquence d'actions composant cette politique ou pour la tester à nouveau. Dans le cadre de cette thèse, seul l'aspect co-construction en entrée de la simulation a été présenté. Les deux autres aspects de la co-construction n'ont pas été éprouvés par les expérimentations proposées. Une perspective serait donc d'éprouver l'instance du modèle SmartGov avec une co-construction pendant la simulation, avec la manipulation en temps réel des structures de l'environnement, et observer l'impact sur la politique ainsi construite.

2. **Visualisation améliorée :** L'architecture SmartGov est générique et permet aux décideurs de s'approprier un outil pour co-construire des politiques urbaines à partir des données rendues disponibles par la Smart City. Par conséquent, une perspective serait de disposer d'interfaces homme-machine permettant une meilleure visualisation et faciliter la transmission des résultats avec le décideur ou les usagers.

3. **Aide fonctionnelle proposée au décideur :** Dans le cadre de cette thèse, même si le modèle s'adresse au décideur politique, un intermédiaire disposant de connaissances rudimentaires en informatique est un plus pour la création et l'implémentation du comportement des agents de la couche microscopique par exemple. Le calibrage des paramètres pour construire l'environnement peut être délicat pour le décideur. Disposer d'un outil simplifié pour la construction des environnements et des populations microscopiques permettrait de réduire encore le besoin d'un intermédiaire dans le processus de co-construction.

4. **Validation sur un exemple réel :** la validation du modèle SmartGov nécessite d'éprouver des politiques réelles. Par conséquent, une des perspectives pour le futur de SmartGov est de simuler une zone urbaine complète avec des données disponibles pour d'une part simuler une première politique urbaine qui sera appliqué dans la réalité et d'autre part de la faire évoluer avec le retour des usagers. Nous avons déjà pour cela reçu plusieurs sollicitations de villes (Saint Etienne, Clermont-Ferrand), et c'est pourquoi nous avons demandé le déploiement d'une plateforme au niveau du laboratoire, qui permettrait ces différentes déclinaisons de l'outil, avec certainement quantité de retours intéressants.

Nous envisageons aussi plusieurs perspectives portant sur l'algorithme CDQN :

1. **Implémentation de méta-actions :** les agents de contrôle choisissent une action parmi celles suggérées par leurs agents locaux et ceux-ci l'appliquent. L'application d'une action unique à un avantage sur la non-stationnarité, cependant il pourrait être envisagé que les agents locaux utilisent des variations de l'action imposée par leur agent de contrôle. L'utilisation d'actions locales doit cependant être étudiée de manière à clairement identifier l'effet sur l'apprentissage et sur les autres agents. En effet les actions seront moins régulièrement sélectionnées.

2. **Compétition et différentes fonctions objectifs :** dans les expérimentations du chapitre précédent, la fonction objectif est identique pour chaque agent politique (qu'il soit global ou local). Il est possible d'envisager des agents de contrôle ayant des fonctions objectif différentes. Un même cluster regrouperait alors que des agents locaux dotés d'objectifs identiques, supervisés par l'agent politique partageant également la même fonction objectif. Cette perspective soulève le risque de non-convergence, en effet des actions différentes seront effectuées sur des ressources communes, et la perspective doit être creusée pour savoir si elle est pertinente.

3. **Hypothèse sur les séquences d'actions :** les agents de contrôle fusionnent lorsqu'ils identifient des séquences d'actions équivalentes. Pour permettre l'identification de ces séquences, une hypothèse a été faite sur le fonctionnement des actions et notamment sur l'idée que l'ordre des actions importe peu sur l'état obtenu. Cette hypothèse nécessite une phase de validation.

4. **Apprentissage de la couche de contrôle :** la couche de contrôle des agents locaux est un ensemble de règles de gestion *ad-hoc* permettant de produire des politiques urbaines sans avoir à modifier la couche macroscopique. Une perspective est de permettre aux agents de contrôle d'apprendre des configurations, d'identifier les contributions pertinentes et d'ainsi améliorer le choix de l'action suggérée. L'objectif serait d'accélérer l'exploration des configurations par les agents de contrôle.

5. **Preuve de convergence du CDQN :** la dernière perspective envisagée sur l'algorithme de CDQN est l'étude formelle de la convergence des agents locaux et globaux lors de l'apprentissage et la garantie que l'ensemble des solutions proposées correspondent à l'ensemble des politiques optimales.

Ce travail est un pas de plus dans la direction d'outils d'aide à la décision intelligents. Le décideur politique, et éventuellement les usagers dans le futur, pourront utiliser ces outils pour construire et suggérer des politiques urbaines dans un écosystème connecté holistique. L'intelligence artificielle sera alors utilisée pour enrichir les perspectives des concepteurs en proposant de nouvelles solutions ou de nouvelles relations entre les nombreuses composantes de la ville de demain. Avec l'émergence de ces approches autonomes et auto-adaptatives, il est important de sensibiliser, dès maintenant, les citoyens au bon usage des nouvelles technologies plutôt que de les effrayer. L'intelligence artificielle devrait être au service des usagers et des décideurs et non les remplacer.

Chapitre 9

Glossaire

A_p^m Ensemble fini des actions disponibles sur les structures que l'agent politique perçoit (page 67)

A_Σ Ensemble des actions qu'un automate considère (pages 51, 56)

$A_\mathcal{H}$ Ensemble fini d'actions politiques disponibles pour le gestionnaire de politiques (pages 66, 68)

E Entrée de la politique (pages 70, 72, 75)

F Ensemble des états finaux de l'automate (page 50)

G Graphe de l'environnement (pages 47, 64, 69)

H L'*horizon de temps* (pages 69–71)

I Ensemble fini de perceptions (pages 47, 69)

I_Σ Ensemble des perceptions pertinentes pour un automate (page 51)

$I_\mathcal{H}$ Ensemble fini de perceptions de l'environnement de la couche usagers (pages 66–68)

N Ensemble d'agents microscopiques concernés par les actions politiques (page 69)

N_m Ensemble fini d'agents microscopiques (pages 47, 72, 73)

N_p Ensemble fini d'agents politiques (pages 65–67)

P^n Personnalité d'un agent humain n (pages 50, 52)

$R_\mathcal{H}$ Ensemble fini de représentations d'environnement construit à partir de l'agrégation de perceptions de $I_\mathcal{H}$ (pages 66, 67, 71)

S Ensemble fini d'états de l'automate (page 50)

T Ensemble fini de types de structures (page 47)

Σ Vecteur d'entrées décrivant les conditions des changement d'états (pages 50, 51)

δ Fonction de transition entre les états de l'automate (page 50)

γ Facteur d'atténuation (page 108)

$\hat{\mathcal{A}}$ Ensemble fini des actions jointes (page 108)

$\hat{\mathcal{D}}$ Ensemble fini des agents dans le CDQN (page 108)

$\hat{\mathcal{O}}$ Ensemble fini des observations jointes (page 108)

$\hat{\mathcal{S}}$ Ensemble fini des états de l'environnement (page 108)

$\mathcal{A}^c$ Ensemble fini des actions de contrôle disponibles pour les agents de contrôle (page 108)

$\mathcal{A}^l$ Ensemble fini des actions environnementales disponibles pour les agents apprenants (pages 108, 110, 123)

$\mathcal{C}$ Ensemble des configurations possibles des clusters (figure 6.2) (pages 110, 120)

$\mathcal{D}^c$ Ensemble fini des agents de contrôle (pages 108, 110)

$\mathcal{D}^l$ Ensemble fini des agents apprenants (pages 108, 109, 123)

$\mathcal{E}$ Environnement (pages 46, 47, 68, 72)

$\mathcal{F}$ Fonction objectif du CDQN pour calculer le gain agrégé des agents de contrôles et l'attribution de la récompense des apprenants (pages 108, 112, 120)

$\mathcal{H}$ L'environnement de la couche macroscopique également appelé *gestionnaire de politiques* (pages 65, 66, 68, 72, 73)

$\mathcal{O}$ Des objectifs spécifiques du décideur politique (fonction objectif, contraintes, besoins, etc.) (pages 70–73, 75)

$\mathcal{O}^c$ Ensemble fini des observations jointes des agents de contrôle (page 108)

$\mathcal{O}^l$ Ensemble fini des observations jointes des agents apprenants (page 108)

$\mathcal{P}$ Une politique urbaine (pages 70–73)

$\mathcal{S}$ Ensemble fini de structures (pages 47, 69, 72)

$\mathcal{S}^c$ Ensemble fini des états des agents de contrôle (page 108)

$\mathcal{S}^l$ Ensemble fini des espaces factorisés des apprenants (page 108)

$\mathcal{S}_p^m$ Ensemble fini de structures que l'agent politique perçoit (page 67)

$\mathcal{S}_\Sigma$ Ensemble des structures qu'un automate considère (page 51)

ϕ L'*indicateur de performance* (pages 69–71, 73)

ψ Le *périmètre environnemental* (pages 69, 71)

ρ_p^m Fonction de décision pour choisir l'action correspondant au vecteur de perceptions fourni par l'environnement (pages 67, 68)

σ Fonction de score évaluant un ensemble de perceptions liées à une action spécifique de l'environnement (pages 54–56, 59)

τ Fonction de score de confiance (page 108)

φ Le *domaine d'action* (pages 69, 71)

ς Fonction de satisfaction représentant le gain personnel d'effectuer une action (pages 54, 55, 59)

ζ	Analyse de l'environnement (pages 70–72)
a_h^n	Un agent humain (pages 50, 52, 59)
a_m^n	Un agent microscopique (page 49)
a_p^m	Un agent politique m (pages 67, 71)
c_i	Caractéristique associées à la perception i (page 52)
d_j^c	Un agent de contrôle ou contrôleur j (pages 110, 111, 114, 115, 117, 119, 120)
d_i^l	Un agent apprenant local ou apprenant i (pages 108–114, 122, 124)
e_0	État initial de l'automate (pages 50, 61)
u_i	Fonction d'utilité lié à la perception i (page 52)
w_i	Poids attribué à une perception i (page 52)
CU	Couche Usagers (page 46)
GDP	Gestionnaire de Politiques (pages 65, 73)

Chapitre 10

Bibliographie

[1] Henk Aarts, Bas Verplanken, and Ad Van Knippenberg. Habit and information use in travel mode choices. *Acta Psychologica*, 96(1-2) :1–14, 1997. 63

[2] Adrian K. Agogino and Kagan Tumer. Unifying temporal and structural credit assignment problems. *Proceedings of 17th International Conference on Autonomous Agents and Multiagent Systems*, pages 980–987, 2004. 112

[3] E. Albaek. Knowledge, interests and the many mearnings of evaluation : a developmental perspective. *Scandinavian Journal of Social Welfare*, 7(2) :94–98, 1998. 16

[4] Vito Albino, Umberto Berardi, and Rosa Maria Dangelico. Smart cities : Definitions, dimensions, performance, and initiatives. *Journal of urban technology*, 22(1) :3–21, 2015. 18

[5] Stefano Albrecht and Peter Stone. Multiagent learning foundations and recent trends, 2017. 98

[6] Martin Allen and Shlomo Zilberstein. Complexity of decentralized control : Special cases. *Advances in Neural Information Processing Systems*, pages 19–27, 2009. 99, 108

[7] Sam Allwinkle and Peter Cruickshank. Creating smart-er cities : An overview. *Journal of urban technology*, 18(2) :1–16, 2011. 18

[8] Robert M. Argent. An overview of model integration for environmental applications—components, frameworks and semantics. *Environmental Modelling & Software*, 19(3) :219–234, 2004. 35

[9] Kenneth J. Arrow. A difficulty in the concept of social welfare. *Journal of Political Economy*, 58(4) :328–346, 1950. 11

[10] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anthony Bharath. A Brief Survey of Deep Reinforcement Learning. *arXiv preprint arXiv :1708.05866*, 2017. 90, 91, 97

[11] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47 :235–256, 2002. 89

[12] Dipyaman Banerjee and Sandip Sen. Reaching pareto-optimality in prisoner's dilemma using conditional joint action learning. *Autonomous Agents and Multi-Agent Systems*, 15(1) :91–108, 2007. 98

[13] Steven C. Bankes. Tools and techniques for developing policies for complex and uncertain systems. In *Proceedings of the National Academy of Sciences*, volume 99, pages 7199–7200, 2002. 13

[14] Eugene Bardach. *A Practical Guide for Policy Analysis The Eightfold Path to More Effective Problem Solving Fourth Edition*. SAGE, 2012. 10

[15] Michael Batty. *Cities and complexity : understanding cities with cellular automata, agent-based models, and fractals*. The MIT press, 2007. 31

[16] Michael Batty and Yichun Xie. Possible urban automata. *Environment and Planning B : Planning and Design*, 24 :175–192, 1997. 30

[17] Richard J. Beckman, Keith A. Baggerly, and Michael D. McKay. Creating synthetic baseline populations. *Transportation Research Part A : Policy and Practice*, 30(6) :415–429, 1996. 31

[18] Jennifer Belissent. The core of a smart city must be smart governance. *Forrester Research Inc., Cambridge*, 2011. 20

[19] Richard Bellman. *Dynamic Programming*. Dover Publications, Inc. Mineola, New York, 1957. 85

[20] Richard Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, pages 679–684, 1957. 84, 86

[21] Itzhak Benenson, Karel Martens, and Slava Birfir. Parkagent : An agent-based model of parking in the city. *Computers, Environment and Urban Systems*, 2008. 31

[22] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning : A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8) :1798–1828, 2013. 90, 91

[23] Daniel S Bernstein, Shlomo Zilberstein, and Neil Immerman. The complexity of decentralized control of markov decision processes. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 32–37. Morgan Kaufmann Publishers Inc., 2000. 99

[24] Donal A. Berry and Bert Fristedt. *Bandit Problems : sequential allocation of experiments*. Chapman and Hall, 1985. 89

[25] Thomas A Birkland. *An Introduction to the Policy Process : Theories, Concepts and Models of public policy making*. Routledge, 2014. 10, 15, 17

[26] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. 82

[27] Daan Bloembergen, Karl Tuyls, Daniel Hennes, and Michael Kaisers. Evolutionary dynamics of multi-agent learning : A survey. *Journal of Artificial Intelligence Research*, 53 :658 – 697, 2015. 101

[28] Eric Bonabeau. Agent-based modeling : Methods and techniques for simulating human systems. *PNAS*, 99(3), 2002. 29

[29] Enrique Bonsón, Sonia Royo, and Melinda Ratkai. Citizens' engagement on local governments' facebook sites. an empirical analysis : The impact of different media and content types in western europe. *Government Information Quarterly*, 32(1) :52–62, 2015. 21

[30] Craig Boutilier, Thomas Dean, and Steve Hanks. Decision-theoretic planning : Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 1999. 102

[31] Michael Bowling. Convergence and no-regret in multiagent learning. *Advances in neural information processing systems*, pages 209–216, 2005. 98

[32] Michael Bowling and Manuela Veloso. Rational and convergent learning in stochastic games. *International Joint Conference on Artificial Intelligence*, 2001. 106

[33] Michael Bowling and Manuela Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2) :215–250, 2002. 98

[34] Danah Boyd and Kate Crawford. Critical questions for big data : Provocations for a cultural, technological, and scholarly phenomenon. *Information, communication & society*, 15(5) :662–679, 2012. 19, 20

[35] M. Buchholz, G. Holst, and O. Musshoff. Irrigation water policy analysis using a business simulation game. *Water Resources Research*, 52 :7980–7998, 2016. 21

[36] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. A comprehensive survey of multi-agent reinforcement learning. *IEEE Transactions on Systems Man and Cybernetics – Part C : Applications and Reviews*, 38(2) :156–172, March 2008. 94, 97, 98, 101

[37] Stéphane Bura, France Guérin-Pace, Hélène Mathian, Denise Pumain, and Lena Sanders. Multiagent systems and the dynamics of a settlement system. *Geographical Analysis*, 28(2) :161–178, 1996. 35

[38] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. Multi-agent reinforcement learning : An overview. *Innovations in multi-agent systems and applications-1*, pages 183–221, 2010. 94, 106

[39] Benjamin Camus. *Environnement Multi-agent pour la Multi-modélisation et Simulation des Systèmes Complexes*. PhD thesis, Université de Lorraine, 2015. 35

[40] Andrea Caragliu, Chiara Del Bo, and Peter Nijkamp. Smart cities in europe. *Journal of urban technology*, 18(2) :65–82, 2011. 18

[41] Paolo Cardullo and Rob Kitchin. Being a 'citizen'in the smart city : up and down the scaffold of smart citizen participation in dublin, ireland. *GeoJournal*, 84(1) :1–13, 2019. 20

[42] Francisco Javier Carrillo, Tan Yigitcanlar, Blanca García, and Antti Lönnqvist. *Knowledge and the city : Concepts, applications and trends of knowledge-based urban development*. Routledge, 2014. 17

[43] Anthony R. Cassandra. A survey of pomdp applications. *Working notes of AAAI 1998 fall symposium on planning with partially observable Markov decision processes*, 1724, October 1998. 89

[44] Jean-Christophe Castella, Tran Ngoc Trung, and Stanislas Boissau. Participatory simulation of land-use changes in the northern mountains of vietnam : the combined use of an agent-based model, a role-playing game, and a geographic information system. *Ecology and Society*, 10(1), 2005. 32

[45] Christian J. E. Castle. Developing a prototype agent-based pedestrian evacuation model to explore the evacuation of king(s cross st pancras underground station. *UCL Centre for Advanced Spatial Analysis*, 2006. 31

[46] Yu-Han Chang, Tracey Ho, and Leslie Pack Kaelbling. All learning is local : Multi-agent learning in global reward games. *Advances in neural information processing systems*, pages 807–814, 2004. 106

[47] Jacob Chapman, Peer-Olaf Siebers, and Darren Robinson. Coupling multi-agent stochastic simulation of occupants with building simulation. *BSO14 building simulation and optimization, second IBPSA—England conference in association with CIBSE, UCL—University College London, London*, pages 23–24, 2014. 35

[48] Bo Chen and Harry H. Cheng. A review of the applications of agent technology in traffic and transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 11 :485–497, 2010. 29

[49] Min Chen, Shiwen Mao, and Yunhao Liu. Big data : A survey. *Mobile networks and applications*, 19(2) :171–209, 2014. 19

[50] Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. *Proceedings of the 15th National Conference on Artificial Intelligence*, pages 746–752, 1998. 98, 102

[51] Stéphane Clinchant, Christopher Dance, Tom De Ruijter, Peer Ghent, and Onno Zoeter. Using analytics to understant on-street parking : the impact of special permit use and the benefit to demand based rates over zones. *22ńd ITS World Congress*, 2015. 129

[52] Annalisa Cocchia. Smart and digital city : A systematic literature review. *Smart city*, pages 13–43, 2014. 17

[53] Jonathan Cohen, Jilles Steeve Dibangoye, and Abdel-Illah Mouaddib. Open Decentralized POMDPs. In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 977–984, New York, New York, USA, Nov 2017. IEEE. 98

[54] Vincent Conitzer and Tuomas Sandholm. Awesome : A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents. *Machine Learning*, 67(1–2) :23–43, 2007. 98

[55] Helen Couclelis. Cellular worlds : a framework for modeling micro—macro dynamics. *Environment and planning A*, 17(5) :585–596, 1985. 35

[56] Helen Couclelis. *Geographic Information Systems and Environmental Modeling*, chapter Modeling frameworks, paradigms, and approaches, pages 1–15. New York : Longman & Co, 2000. 31

[57] Helen Couclelis. *Why I no longer work with agents : A challenge for ABMs of human-environment interactions*, pages 4–7. LUCC Focus 1 Office, 2001. 33

[58] Andrew T. Crooks and Christian J.E. Castle. *The Integration of Agent-Based Modelling and Geographical Information for Geospatial Simulation*, chapter 12, pages 219–251. Springer, 2012. 34

[59] Jon P Daries, Justin Reich, Jim Waldo, Elise M Young, Jonathan Whittinghill, Andrew Dean Ho, Daniel Thomas Seaton, and Isaac Chuang. Privacy, anonymity, and big data in the social sciences. *Communications of the ACM*, 57(9), 2014. 20

[60] Daniel David, Denis Payet, Aurélie Botta, Gilles Lajoie, Sébastien Manglou, and Rémy Courdier. Un couplage de dynamiques comportementales : le modèle ds pour l'aménagement du territoire. *JFSMA*, 7 :129–138, 2007. 35

[61] Tomás De La Barra, Beatriz Pérez, and N Vera. Tranus-j : putting large models into small computers. *Environment and Planning B : Planning and Design*, 11(1) :87–101, 1984. 31

[62] Andrea De Mauro, Marco Greco, and Michele Grimaldi. What is big data ? a consensual definition and a review of key research topics. *AIP conference proceedings*, 1644(1) :97–104, 2015. 19

[63] Auriol Degbelo, Carlos Granell, Sergio Trilles, Devanjan Bhattacharya, Sven Casteleyn, and Christian Kray. Opening up smart cities : citizen-centric challenges and opportunities from giscience. *ISPRS International Journal of Geo-Information*, 5(2) :16, 2016. 20

[64] Yves Demazeau. From interactions to collective behaviour in agent-based systems. *Proceedings of the 1st. European Conference on Cognitive Science*, 1995. 29

[65] J. Dewey. *Public and its problems*. Henry Holt and Company, 1927. 3

[66] Edsger Wybe Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1 :269–271, 1959. 64, 130

[67] Kurt Dresner and Peter Stone. Multiagent traffic management : A reservation-based intersection control mechanism. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 530–537, 2004. 31

[68] Alexis Drogoul, Edouard Amouroux, Philippe Caillou, Benoit Gaudou, Arnaud Grignard, Nicolas Marilleau, Patrick Taillandier, Maroussia Vavasseur, Duc-An Vo, and Jean-Daniel Zucker. Gama : multi-level and complex environment for agent-based models and simulations. *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 1361–1362, 2013. 34

[69] Cécile Duchêne and Guillaume Touya. Emergence de zones conflits dans deux modèles de généralisation cartographique multi-agents. *JFSMA*, pages 33–42, 2010. 36

[70] Ed Durfee and Shlomo Zilberstein. *Multiagent Planning, Control, and Execution*, chapter 11, pages 485–545. MIT Press, second edition, 2013. 99

[71] Patrick d'Aquino, Christophe Le Page, Francois Bousquet, Alassane Bah, et al. A novel mediating participatory modelling : the 'self-design'process to accompany collective decision making. *International journal of agricultural resources, governance and ecology*, 2(1) :59–74, 2002. 20

[72] Patrick A.M. Ehlert and Leon J.M. Rothkrantz. Microscopic traffic simulation with reactive driving agents. *2001 IEEE Intelligent Transportation Systems Conference Proceedings*, 2001. 31

[73] Robert Englemore and Anthony Morgan. *Blackboard Systems ; Edited by Robert Engelmore, Tony Morgan (the Insight Series in Artificial Intell.* Addison-Wesley Longman Publishing Co., Inc., 1988. 28

[74] J. Doyne Farmer and Duncan Foley. The economy needs agent-based modelling. *Nature*, 460 :685–686, 2009. 23

[75] Jacques Ferber. *Les systèmes Multi-Agents*, volume 1. InterEditions, 1995. 26, 28, 29, 36

[76] Jacques Ferber. Les systèmes multi-agents : un aperçu général. *Technique et Science Informatiques*, 16(8) :979–1012, 1997. 29

[77] Jakob Foerster, Nantas Nardelli, Gregory Farquhar, Triantafyllos Afouras, Philip. H. S. Torr, Pushmeet Kohli, and Shimon Whiteson. Stabilising Experience Replay for Deep Multi-Agent Reinforcement Learning. *arXiv preprint arXiv :1702.08887*, 2017. 96

[78] Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. Learning to Communicate to Solve Riddles with Deep Distributed Recurrent Q-Networks. *arXiv preprint arXiv :1602.02672*, 2016. 96

[79] Pierre Fosset, Isabelle Andre-Poyaud, Arnaud Banos, Elise Beck, Sonia Chardonnel, Alexis Conesa, Christophe Lang, Thomas Leysens, Nicolas Marilleau, Arnaud Piombini, and Thomas Thévenin. Exploring intra-urban accessibility and impacts of pollution policies with an agent-based simulation platform : Gamirod. *Systems*, 2016. 23

[80] Nicolas Gaud, Stéphane Galland, and Abderrafiâa Koukam. Towards a multilevel simulation approach based on holonic multiagent systems. *Tenth International Conference on Computer Modeling and Simulation (uksim 2008)*, pages 180–185, 2008. 34

[81] Nicolas Gaud, Franck Gechter, Stéphane Galland, and Abderrafiâa Koukam. Holonic multiagent multilevel simulation application to real-time pedestrians simulation in urban environment. *IJCAI*, pages 1275–1280, 2007. 34

[82] Sumit Ghosh. On the concept of dynamic multi-level simulation. *19th Annual Symposium on Simulation*, pages 201–205, 1986. 34

[83] David V Gibson, George Kozmetsky, and Raymond W Smilor. *The technopolis phenomenon : Smart cities, fast systems, global networks.* Rowman & Littlefield, 1992. 17

[84] Rudolf Giffinger, Christian Fertner, Hans Kramar, Evert Meijers, et al. City-ranking of european medium-sized cities. *Cent. Reg. Sci. Vienna UT*, pages 1–12, 2007. 18

[85] Rudolf Giffinger and Haindlmaier Gudrun. Smart cities ranking : an effective instrument for the positioning of the cities ? *ACE : architecture, city and environment*, 4(12) :7–26, 2010. 20

[86] P. G. Gipps. A behavioural car-following model for computer simulation. *Transportation Research Part B : Methodological*, 15(2) :105–111, 1981. 64, 131

[87] Pierre Yves Glorennec. Reinforcement learning : An overview. *Proceedings European Symposium on Intelligent Techniques (ESIT-00), Aachen, Germany*, pages 14–15, 2000. 90

[88] Yiwei Gong and Marijn Janssen. From policy implementation to business process management : Principles for creating flexibility and agility. *Government Information Quaterly*, 29 :S61–S71, 2012. 21

[89] Amy Greenwald, Keith Hall, and Roberto Serrano. Correlated q-learning. *ICML*, 3 :242–249, 2003. 98

[90] Volker Grimm, Eloy Revilla, Uta Berger, Florian Jeltsch, Wolf M Mooij, Steven F Railsback, Hans-Hermann Thulke, Jacob Weiner, Thorsten Wiegand, and Donald L DeAngelis. Pattern-oriented modeling of agent-based complex systems : lessons from ecology. *Science*, 310(5750) :987–991, 2005. 29

[91] Eric A Hansen, Daniel S Bernstein, and Shlomo Zilberstein. Dynamic programming for partially observable stochastic games. *AAAI*, 4 :709–715, 2004. 98

[92] Kirkl Harland, Alison Heppenstall, Dianna Smith, and Mark Birkin. Creating realistic synthetic populations at varying spatial scales : A comparative critique of population synthesis techniques. *Journal of Artificial Societies and Social Simulation*, 15(1), 2012. 32

[93] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2) :100 – 107, July 1968. 64, 130

[94] Sergiu Hart and Andreu Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68(5) :1127–1150, 2000. 98

[95] Sergiu Hart and Andreu Mas-Colell. A reinforcement procedure leading to correlated equilibrium. *Economics Essays*, pages 181–200, 2001. 98

[96] Hiromitsu Hattori, Yuu Nakajima, and Toru Ishida. Learning from humans : Agent modeling with individual human behaviors. *IEEE Transactions on Systems, Man, and Cybernetics-Part A : Systems and Humans*, 41(1) :1–9, 2011. 32

[97] Brian W. Head. Three lenses of evidence-based policy. *The Australian Journal of Pulbic Administration*, 67(1) :1–11, 2007. 17

[98] Pablo Hernandez-Leal, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote. A survey of learning in multiagent environments : Dealing with non-stationarity. *arXiv preprint arXiv :1707.09183*, 2017. 95

[99] Ralph Hertwig and Andreas Ortmann. Experimental practices in economics : A methodological challenge for psychologists ? *Behavioral and Brain Sciences*, 24 :383–451, 2001. 21

[100] Stephane Hess and John W. Polak. An analysis of parkin behaviour using discrete choice models calibrated on sp datasets. *44th Congress of the Euopean Regional Science Association*, 2004. 130

[101] Tony Hey and Anne E Trefethen. Cyberinfrastructure for e-science. *Science*, 308(5723) :817–821, 2005. 19

[102] Heather C Hill. Understanding implementation : Street-level bureaucrats' resources for reform. *Journal of Public Administration Research and Theory*, 13(3) :265–282, 2003. 17

[103] Robin M. Hogarth. Beyond discrete biases : Functional and dysfunctional aspects of judgemental heuristics. *Psychological Bulletin*, 1981. 11

[104] Brian W Hogwood, Lewis A Gunn, and Sean Archibald. *Policy analysis for the real world*, volume 69. Oxford University Press Oxford, 1984. 17

[105] Robert G. Hollands. Will the real smart city please stand up ? *City*, 12(3) :303–320, dec 2008. 17, 18

[106] Robert G Hollands. Critical interventions into the corporate smart city. *Cambridge Journal of Regions, Economy and Society*, 8(1) :61–77, 2015. 17

[107] Gesa Sophie Holst and Oliver Musshoff. Policy impact analysis of penalty and reward scenarios to promote flowering cover crops using a business simulation game. *Agri-Food and Rural Innovations for Healthier Societies*, 2014. 13, 21

[108] Christopher Hood. The risk game and the blame game. *Government and Opposition*, 37(1) :15–37, 2002. 16

[109] Ronald A. Howard. *Dynamic programming and Markov Processes*. Technology Press of Massachusetts Institute of Technology, 1960. 86

[110] Junling Hu and Michael P Wellman. Nash q-learning for general-sum stochastic games. *Journal of machine learning research*, 4(Nov) :1039–1069, 2003. 98

[111] John D Hunt, David S Kriger, and Eric J Miller. Current operational urban land-use–transport modelling frameworks : A review. *Transport reviews*, 25(3) :329–376, 2005. 31

[112] Helen Ingram. *"Policy Failure : An Issue Deserving Analysis."*. Sage Publications Beverly Hills, 01 1980. 16

[113] Toru Ishida, Yuu Nakajima, Yojei Murakami, and Hideyuki Nakanishi. Augmented experiment : Participatory design with multiagent simulation. *IJCAI*, 7 :1341–1346, 2007. 32

[114] Peter Jackson. *Introduction to Expert Systems*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1998. 23

[115] Werner Jann and Kai Wegrich. *Theories of the policy cycle*, chapter 4, pages 43–62. CRC Press, 2006. 16

[116] Katleen Janssen. The influence of the psi directive on open government data : An overview of recent developments. *Government Information Quarterly*, 28(4) :446–456, 2011. 18

[117] Marijn Janssen, Yannis Charalabidis, and Anneke Zuiderwijk. Benefits, adoption barriers and myths of open data and open government. *Information systems management*, 29(4) :258–268, 2012. 19

[118] Nicholas R. Jennings. An agent-based approach for building complex software systems. *Communications of the ACM*, 44(4) :35–41, 2001. 29

[119] Nicholas R. Jennings, Katia Sycara, and Michael Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems*, 1 :7 – 38, 1998. 23, 28

[120] Erik W Johnston and Derek L Hansen. Design lessons for smart governance infrastructures. *Transforming American governance : Rebooting the public square*, pages 197–212, 2011. 20

[121] Bryan D. Jones. *Reconceiving Decision-Making in Democratic Politics : Attention, Choice, and Public Policy*. The University of Chicago Press, 1994. 11

[122] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2) :99–134, 1998. 83, 89

[123] Daniel Kahneman, Jack L. Knetsch, and Richard H. Thaler. Anomalies the endowment effect, loss aversion and status quo bias. *Journal of Economic Perspectives*, 5(1) :193–206, Winter 1991. 13

[124] Daniel Kahneman and Amos Tversky. Prospect theory : An analysis of decision under risk. *Econometrica*, 47(2) :263–292, March 1979. 13

[125] Spiros Kapetanakis and Daniel Kudenko. Reinforcement learning of coordination in cooperative multi-agent systems. *AAAI/IAAI*, 2002 :326–331, 2002. 101

[126] William G Kennedy. Modelling human behaviour in agent-based models. *Agent-based models of geographical systems*, pages 167–179, 2012. 31

[127] Nawsher Khan, Ibrar Yaqoob, Ibrahim Abaker Targio Hashem, Zakira Inayat, Mahmoud Ali, Waleed Kamaleldin, Muhammad Alam, Muhammad Shiraz, and Abdullah Gani. Big data : survey, technologies, opportunities, and challenges. *The Scientific World Journal*, 2014, 2014. 19

[128] Charles H Kieffer. Citizen empowerment : A developmental perspective. *Prevention in human services*, 3(2-3) :9–36, 1984. 20

[129] Elmar Kiesling, Markus Günther, Christian Stummer, and Lea M. Wakolbinger. Agent-based simulation of innovation diffusion : a review. *CEJOR*, 20 :183–230, 2012. 23

[130] Diederik P. Kingma and Jimmy Lei Ba. ADAM : A Method for Stochastic Optimization. *International conference on machine learning*, 2015. 143

[131] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13) :3521–3526, 2017. 92

[132] Rob Kitchin. The ethics of smart cities and urban science. *Philosophical Transactions of the Royal Society A : Mathematical, Physical and Engineering Sciences*, 374(2083) :20160115, 2016. 20

[133] Reinout Kleinhans, Maarten Van Ham, and Jennifer Evans-Cowley. Using social media and mobile technologies to foster engagement and self-organization in participatory urban planning and neighbourhood governance. *Planning Practice & Research*, 30(3), 2015. 21

[134] Frank H. Knight. *Risk, uncertainty and profit*. Courier Corporation, 1921. 11

[135] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics : A survey. *The International Journal of Robotics Research*, 32(11) :1238–1274, 2013. 82

[136] Nicos Komninos. *Intelligent cities and globalisation of innovation networks*. Routledge, 2008. 17

[137] Tobias Krueger, Trevor Page, Klaus Hubacek, Laurence Smith, and Kevin Hiscock. The role of expert opinion in environmental modelling. *Environmental Modelling & Software*, 36 :4 – 18, 2012. 20

[138] Guillaume Lample and Devendra Singh Chaplot. Playing FPS Games with Deep Reinforcement Learning. *Association for the Advancement of Artificial Intelligence*, pages 2140–2146, 2017. 93

[139] Gerard F Laniak, Gabriel Olchin, Jonathan Goodall, Alexey Voinov, Mary Hill, Pierre Glynn, Gene Whelan, Gary Geller, Nigel Quinn, Michiel Blind, Scott Peckham, Sim Reaney, Noha Gaber, Robert Kennedy, and Andrew Hughes. Integrated environmental modeling : A vision and roadmap for the future. *Environmental Modelling & Software*, 39 :3–23, 2013. 20, 21

[140] Martin Lauer and Martin Riedmiller. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. *In Proceedings of the Seventeenth International Conference on Machine Learning*, 2000. 101

[141] Guillaume J Laurent, Laëtitia Matignon, Le Fort-Piat, et al. The world of independent learners is not markovian. *International Journal of Knowledge-based and Intelligent Engineering Systems*, 15(1) :55–64, 2011. 106

[142] Christophe Le Page, Didier Bazile, Nicolas Becu, Pierre Bommel, François Bousquet, Michel Etienne, Raphael Mathevet, Véronique Souchere, Guy Trébuil, and Jacques Weber. *Agent-based modelling and simulation applied to environmental management*, chapter 19, pages 499–540. Springer, 2013. 23

[143] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553) :436, 2015. 90

[144] Honglak Lee, Peter Pham, Yan Largman, and Andrew Y Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. *Advances in neural information processing systems*, pages 1096–1104, 2009. 82

[145] Joel Z. Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. Multi-agent reinforcement learning in sequential social dilemmas. *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 464–473, 2016. 96

[146] Robert J. Lempert, Steven W. Popper, and Steven C. Bankes. *Shaping the Next One Hundred Years : New Methods for Quantitative, Long-Term Policy Analysis*. RAND, 2003. 11, 12, 13

[147] Soumaya Ben Letaifa. How to strategize smart cities : Revealing the smart model. *Journal of Business Research*, 68(7) :1414–1419, 2015. 18

[148] Long-Ji Lin. *Reinforcement Learning for Robots Using Neural Networks*. PhD thesis, Carnegie Mellon University, 1993. 92

[149] Michael Lipsky. *Street-level bureaucracy : Dilemmas of the individual in public services*. New York : Russell Sage Foundation, 1980. 17

[150] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. *Machine learning proceedings*, pages 157–163, 1994. 98

[151] Michael L Littman. Value-function reinforcement learning in markov games. *Cognitive Systems Research*, 2(1) :55–66, 2001. 102

[152] Yan Liu. *Modelling Urban Development with Geographical Information Systems and Cellular Automata*. CRC Press, 2008. 30

[153] Paul A. Longley, Michael F. Goodchild, David J. Maguire, and David W. Rhind. *Geographic Information Science and Systems*. Wiley, 4 edition, 2015. 34

[154] James MacQueen et al. Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 1(14) :281–297, 1967. 82

[155] James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, and Angela H Byers. *Big data : The next frontier for innovation, competition, and productivity*. McKinsey Global Institute, 2011. 19

[156] James G. March and Herbert A. Simon. *Organizations*. Wiley, 1958. 11

[157] V. A. W. J. Marchau, Warren E. Walker, and G. P. van Wee. Dynamic adaptive transport policies for handling deep uncertainty. *Technological Forecasting and Social Change*, 77(6) :940–950, July 2010. 13

[158] Maria-Lluïsa Marsal-Llacuna, Joan Colomer-Llinàs, and Joaquim Meléndez-Frigola. Lessons in urban monitoring taken from sustainable and livable cities to better address the smart cities initiative. *Technological Forecasting and Social Change*, 90 :611–622, 2015. 18

[159] Laëtitia Matignon. *Synthèse d'agents adaptatifs et coopératifs par apprentissage par renforcement. Application à la commande d'un système distribué de micromanipulation.* PhD thesis, Université de Franche-Comté, 2008. 87

[160] Laetitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative markov games : a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(1) :1–31, 2012. 101

[161] Robin Matthews, Nigel Gilbert, Alan Roach, Gary Polhill, and Nick Gotts. Agent-based land-use models : a review of applications. *Landscape Ecology*, 22(10) :1447–1459, 2007. 23, 31

[162] Peter J May. Policy learning and failure. *Journal of public policy*, 12(4) :331–354, 1992. 17

[163] Igor S. Mayer. The gaming of policy and the politics of gaming : A review. *Simulation & Gaming*, 40(6) :825–862, 2009. 21

[164] John McCarthy. Some expert systems need common sense. *Annals of the New York Academy of Sciences*, 426(1) :129–137, 1984. 23

[165] Allan McConnell. Policy success, policy failure and grey areas in-between. *Journal of Public Policy*, 30(10) :345–362, 2010. 16

[166] Nelson Minar, Roger Burkhart, Chris Langton, Manor Askenazi, et al. The swarm simulation system : A toolkit for building multi-agent simulations. *Technical report, Swarm Development Group*, 1996. 34

[167] Tom M. Mitchell. *Machine Learning*. McGraw HIll, 1997. 82

[168] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv :1312.5602*, 2013. 83, 91, 93, 94, 96, 153

[169] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-Level Control Through Deep Reinforcement Learning. *Nature*, 518(7540) :529, 2015. 91, 93

[170] Luca Mora, Roberto Bolici, and Mark Deakin. The first two decades of smart-city research : A bibliometric analysis. *Journal of Urban Technology*, 24(1) :3–27, 2017. 18

[171] Gildas Morvan. Multi-level agent-based modeling-bibliography. *CoRR, vol. abs/1205.0561*, 2012. 34, 36

[172] Jean Pierre Müller. *Des systèmes autonomes aux systèmes multi-agents : Interaction, émergence et systèmes complexes*. PhD thesis, UM2, 2002. 31

[173] Beniamino Murgante and Giuseppe Borruso. Smart cities in a smart world. *Future City Architecture for Optimal Living*, pages 13–35, 2015. 18, 21

[174] Kevin P. Murphy. A survey of pomdp solution techniques. *environment*, 2000. 89

[175] Kirill Müller and Kay W. Axhausen. Population synthesis for microsimulation : State of the art. *ETH Zürich, Institut für Verkehrsplanung, Transporttechnik, Strassen-und Eisenbahnbau (IVT)*, 2010. 32

[176] Taewoo Nam and Theresa A Pardo. Conceptualizing smart city with dimensions of technology, people, and institutions. *Proceedings of the 12th annual international digital government research conference : digital government innovation in challenging times*, pages 282–291, 2011. 17, 18

[177] Thanh Thi Nguyen, Ngoc Duy Nguyen, and Saeid Nahavandi. Deep reinforcement learning for multi-agent systems : A review of challenges, solutions and applications. *CoRR*, abs/1812.11794, 2018. 95, 97

[178] Donald D. Norman. New views of information processing : Implications for intelligent decision support systems. *NATO Advanced Study Institute on Intelligent Decision Support in Process Environments*, pages 123–136, 1985. 11

[179] Michael J. North, T.R. Howe, Nicholson T. Collier, and J.R. Vos. The repast simphony runtime system. In C.M. Macal, M.J. North, and D. Sallach, editors, *Proceedings of the Agent2005 Conference on Generative Social Processes, Models, and Mechanisms*, 2005. 129

[180] James J Odell, H Van Dyke Parunak, and Mitchell Fleischer. The role of roles in designing effective agent organizations. *International Workshop on Software Engineering for Large-Scale Multi-agent Systems*, pages 27–38, 2002. 28

[181] Frans A Oliehoek and Christopher Amato. *A concise introduction to decentralized POMDPs*. Springer, 2016. 98, 99

[182] Frans A. Oliehoek, Matthijs T.J. Spaan, Shimon Whiteson, and Nikos Vlassis. Exploiting locality of interaction in factored dec-pomdps. *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems*, 2008. 99, 108

[183] Frans A Oliehoek, Arnoud Visser, et al. A hierarchical model for decentralized fighting of large scale urban fires. *Proc. of the AAMAS'06 Workshop on Hierarchical Autonomous Agents and Multi-Agent Systems (H-AAMAS)*, pages 14–21, 2006. 98

[184] Frans A. Oliehoek, Shimon Whiteson, and Matthijs T.J. Spaan. Approximate solutions for factored dec-pomdps with many agents. *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, May 2013. 99

[185] Eugénio Oliveira, Klaus Fischer, and Olga Stepankova. Multi-agent systems : which research for which applications. *Robotics and Autonomous Systems*, 27 :91–106, 1999. 29

[186] Shayegan Omidshafiei, Jason Pazis, Christopher Amato, Jonathan P. How, and John Vian. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. *Proceedings of the 34th International Conference on Machine Learning*, 2017. 96

[187] Guillermo Owen. *Game Theory*. Emerald Group Publishing Limited, 1995. 98

[188] Jessen Page, Darren Robinson, Nicolas Morel, and J-L Scartezzini. A generalised stochastic model for the simulation of occupant presence. *Energy and buildings*, 40(2) :83–98, 2008. 35

[189] Simon Pageaud, Véronique Deslandres, Vassilissa Lehoux, and Salima Hassas. Co-Construction of Adaptive Public Policies using SmartGov. *29th International Conference on Tools with Artificial Intelligence*, 2017. 41

[190] Simon Pageaud, Véronique Deslandres, Vassilissa Lehoux, and Salima Hassas. Couplage de simulations multi-agents pour la conception de politiques urbaines. *26èmes Journées Francophones sur les Systèmes Multi-Agents*, 2018. 41

[191] Simon Pageaud, Véronique Deslandres, Vassilissa Lehoux, and Salima Hassas. Application du clustered deep q-network aux politiques tarifaires. *Conférence Nationale sur les Applications Pratiques de l'Intelligence Artificielle (APIA)*, 2019. 108

[192] Simon Pageaud, Véronique Deslandres, Vassilissa Lehoux, and Salima Hassas. Multiagent learning and coordination with clustered deep q-network. *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019)*, page 3, 2019. 108

[193] Liviu Panait and Sean Luke. Cooperative multi-agent learning : The state of the art. *Autonomous Agents and Multi-Agent Systems*, 2005. 98

[194] Thomas Paris, Laurent Ciarletta, and Vincent Chevrier. Designing co-simulation with multi-agent tools : a case study with netlogo. *15th European Conference on Multi-Agent Systems*, pages 253–267, 2017. 35

[195] Dawn C. Parker, Steven M. Manson, Marco A. Janssen, Matthew J. Hoffmann, and Peter Deadman. Multi-agent systems for the simulation of land-use and land-cover change : A review. *Annals of the association of American Geographers*, 93(2) :314–337, 2002. 23

[196] Derek Partridge. The scope and limitations of first generation expert systems. *Future Generation Computer Systems*, 3(1) :1–10, 1987. 23

[197] H. Van Dyke Parunak. Applications of distributed artificial intelligence in industry. *Foundations of distributed artificial intelligence*, 2, 1996. 29

[198] Alenka Poplin. Playful public participation in urban planning : A case study for online serious games. *Computers, Environment and Urban Systems*, 36, 2012. 21

[199] Manon Predhumeau. Mobilité intelligente et gouvernance participative. Master's thesis, Institut Supérieur d'Informatique, de Modélisation et de leurs Applications, 2017. 154

[200] Jeffrey L Pressman and Aaron Wildavsky. *Implementation : How great expectations in Washington are dashed in Oakland ; Or, why it's amazing that federal programs work at all, this being a saga of the Economic Development Administration as told by two sympathetic observers who seek to build morals on a foundation*. Univ of California Press, 1984. 17

[201] David R Pritchard and Eric J Miller. Advances in agent population synthesis and application in an integrated land use and transportation model. *Transportation Research Board 88th Annual Meeting*, 2009. 32

[202] Denise Pumain. Pour une théorie évolutive des villes. *L'Espace géographique*, pages 119–134, 1997. 14

[203] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix : Monotonic value function factorisation for deep multi-agent reinforcement learning. *35th International Conference on Machine Learning*, 2018. 96, 102

[204] C Ratti and A Townsend. The social nexus. the best way to harness a city's potential for creativity and innovation is to jack people into the network and get out of the way. *Scientific American, New York*, 2011. 19

[205] Craig W. Reynolds. Flocks, herds, and schools : A distributed behavioral model. *ACM SIGGRAPH computer graphics*, 21(4) :25–34, August 1987. 29

[206] Rosaldo J.F. Rossetti, Joao Emilio Almeida, Zafeiris Kokkinogenis, and Joel Gonçalves. Playing transportation seriously : Applications of serious games to artificial transportation systems. *Intelligent Transportation Systems*, 2013. 21

[207] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3) :1–4, 1986. 91

[208] Stuart Russel and Peter Norvig. *Artificial Intelligence : A Modern Approach, 3rd Edition*. Pearson, 2010. 27, 28, 82, 83

[209] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. *Learning for Text Categorization : Papers from the 1998 workshop*, 62 :98–105, 1998. 82

[210] Arthur L. Samuel. Somes studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3) :210–229, 1959. 82

[211] Ian Sanderson. Evaluation, policy learning and evidence-based policy making. *Public Administration*, 80(1) :1–22, 2002. 16, 17

[212] Hans Schaffers, Nicos Komninos, Marc Pallot, Brigitte Trousse, Michael Nilsson, and Alvaro Oliveira. Smart cities and the future internet : Towards cooperation frameworks for open innovation. *The Future Internet Assembly*, pages 431–446, 2011. 17

[213] Thomas C. Schelling. Dynamic models of segregation. *Journal of Mathematical Sociology*, 1 :143–186, 1971. 30

[214] Anne Schneider and Helen Ingram. Behavioral assumptions of policy tools. *The Journal of Politics*, 52(2) :510–529, May 1990. 13

[215] Manon Seppecher. Modélisation multi agents pour l'évaluation d'une politique de zone de circulation restreinte sur la ville de lyon. Master's thesis, École Centrale de Nantes, 2018. 155

[216] David Servat, Edith Perrier, Jean-Pierre Treuil, and Alexis Drogoul. When agents emerge from agents : Introducing multi-scale viewpoints in multi-agent simulations. *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pages 183–198, 1998. 34, 36

[217] Shvetank Shah, Andrew Horne, and Jaime Capellá. Good data won't guarantee good decisions. *Harvard Business Review*, 90(4), 2012. 19

[218] Yoram Shiftan and Rachel Burd-Eden. Modeling response to parking policy. *Transportation Research Record*, 1765(1) :27–34, 2001. 139

[219] Yoav Shoham, Rob Powers, and Trond Grenager. If multi-agent learning is the answer, what is the question ? *Artificial Intelligence*, 171(7) :365–377, 2007. 98

[220] Donald C Shoup. *The high cost of free parking*, volume 17. Sage Publications Sage CA : Thousand Oaks, CA, 2011. 132

[221] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676) :354, 2017. 83, 91, 93

[222] Herbert A. Simon. *Administrative Behavior : a Study of Decision-Making Processes in Administrative Organization*. Macmillan, 1947. 11

[223] Herbert A. Simon. Human nature in politics : The dialogue of psychology with political science. *The American Political Science Review*, 79(2) :293–304, June 1985. 11

[224] Michael Sipser. *Introduction to the Theory of Computation*. Thomson Course Technology, 2006. 50

[225] Uthayasankar Sivarajah, Muhammad Mustafa Kamal, Zahir Irani, and Vishanth Weerakkody. Critical analysis of big data challenges and analytical methods. *Journal of Business Research*, 70 :263–286, 2017. 20

[226] Richard D Smallwood and Edward J Sondik. The optimal control of partially observable markov processes over a finite horizon. *Operations research*, 21(5) :1071–1088, 1973. 88

[227] Daniel J Solove. A taxonomy of privacy. *U. Pa. L. Rev.*, 154 :477, 2005. 20

[228] James Surowiecki. *The Wisdom of Crowds*. Anchor, 2005. 21

[229] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1) :9–44, 1988. 87

[230] Richard S. Sutton and Andrew G. Barto. *Renforcement Learning : An Introduction*. MIT Press, 1998. 83, 86, 89, 90

[231] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. Multiagent Cooperation and Competition with Deep Reinforcement Learning. *PloS ONE*, 12(4), April 2017. 94, 96, 145

[232] Ming Tan. Multi-agent reinforcement learning : Independent vs. cooperative agents. *Proceedings of the tenth international conference on machine learning*, pages 330–337, 1993. 101

[233] Lili Tong, Audrey Serna, Simon Pageaud, Sébastien George, and Aurélien Tabard. It's not how you stand, it's how you move : F-formations and collaboration dynamics in a mobile learning game. *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, pages 318–329, 2016. 21

[234] Claude F Touzet. Neural reinforcement learning for behaviour synthesis. *Robotics and Autonomous Systems*, 22(3-4) :251–281, 1997. 90

[235] Chun-Wei Tsai, Chin-Feng Lai, Han-Chieh Chao, and Athanasios V Vasilakos. Big data analytics : a survey. *Journal of Big data*, 2(1) :21, 2015. 19

[236] John N. Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine Learning*, 16 :185–202, 1994. 87

[237] Alexis Tsoukiàs. De la théorie de la décision à l'aide de la décision. *D. Bouyssou, D. Dubois, M. Pirlot, & H. Prade, Concepts et méthodes pour l'aide à la décision*, 2006. 11

[238] Judith Tsouvalis and Claire Waterton. Building 'participation'upon critique : The loweswater care project, cumbria, uk. *Environmental Modelling & Software*, 36 :111–121, 2012. 20

[239] Karl Tuyls and Kagan Tumer. *Multiagent Learning*, chapter 10, pages 423–483. MIT Press, second edition, 2013. 95, 96

[240] Karl Tuyls and Gerhard Weiss. Multiagent learning : Basics, challenges, and prospects. *AI Magazine*, 33(3) :41, 09 2012. 83, 95, 103

[241] Tom R Tyler. *Why people obey the law*. Princeton University Press, 2006. 13

[242] Marjan Van den Belt. *Mediated modeling : a system dynamics approach to environmental consensus building*. Island press, 2004. 20

[243] Hado van Hasselt, Arthur Guez, Matteo Hessel, Volodymyr Mnih, and David Silver. Learning Values Across many Orders of Magnitude. *Advances in Neural Information Processing Systems*, 2016. 93, 112

[244] Katja Verbeeck, Ann Nowé, and Karl Tuyls. Coordinated exploration in multi-agent reinforcement learning : an application to load-balancing. *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 1105–1106, 2005. 96

[245] Bas Verplanken, Henk Aarts, AD Van Knippenberg, and Anja Moonen. Habit versus planned behaviour : A field experiment. *British journal of social psychology*, 37(1) :111–128, 1998. 63

[246] José M. Vidal and Edmund H. Durfee. The moving target function problem in multi-agent learning. *International Conference on Multi-Agent Systems*, 1998. 97

[247] Nikos Vlassis. A concise introduction to multiagent systems and distributed artificial intelligence. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 1(1) :1–71, 2007. 102

[248] Alexey Voinov, Nagesh Kolagani, Michael Keith McCall, Pierre Glynn, Marit Ellen Kragt, Frank Ostermann, Suzanne Alise Pierce, and Palaniappan Ramu. Modelling with stakeholders - next generation. *Environmental Modelling and Software*, 2016. 21, 22

[249] John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944. 11

[250] Warren E. Walker. Policy analysis : A systematic approach to supporting policymaking in the public sector. *Journal of Multi-Criteria Decision Analysis*, 9 :11–27, 2000. 3

[251] Warren E. Walker, S. Adnan Rahman, and Jonathan Cave. Adaptive policies, policy analysis, and policy-making. *European Journal of Operational Research*, pages 282–289, 2001. 13

[252] Rashid A. Waraich and Kay W. Axhausen. An agent-based parking choice model. *Transportation Research Record : Journal of the Transportation Research Board*, 2319 :39–46, 2012. 130

[253] Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards*. King's College, Cambridge, 1989. 87

[254] Danny Weyns, H Van Dyke Parunak, Fabien Michel, Tom Holvoet, and Jacques Ferber. Environments for multiagent systems state-of-the-art and research challenges. *International Workshop on Environments for Multi-Agent Systems*, pages 1–47, 2004. 28, 29

[255] Louise G White. Approaches to land use policy. *Journal of the American Planning Association*, 45(1) :62–71, 1979. 20

[256] Shimon Whiteson, Brian Tanner, Matthew E Taylor, and Peter Stone. Protecting against evaluation overfitting in empirical reinforcement learning. *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 120–127, 2011. 84

[257] Marco Wiering, Jelle van Veenen, Jilles Vreeken, and Arne Koopman. Intelligent traffic light control. *Technical Report UU-CS-2004-029, University Utrecht*, pages 1151–1158, 2004. 31

[258] Michael Wooldridge and Nicholas R. Jennings. Intelligent agents : theory and practice. *The Knowledge Engineering Review*, 1995. 28

[259] Onno Zoeter, Christopher Dance, Stéphane Clinchant, and Jean-Marc Andreoli. New algorithms for parking demand management and a city scale deployment. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1819–1828, 2014. 129

[260] Cliff Zukin, Scott Keeter, Molly Andolina, Krista Jenkins, and Michael X Delli Carpini. *A new engagement ? : Political participation, civic life, and the changing American citizen*. Oxford University Press, 2006. 15