Optimisation avancée pour la recherche et la composition des itinéraires comodaux au profit des clients de transport

Zhanjun Wang

N° d'ordre : 286

Ecole Centrale de Lille

THESE

Présentée en vue d'obtenir le grade de

DOCTEUR

En

Spécialité : Automatique, Génie Informatique, Traitement du Signal et Image

Par

Zhanjun WANG

DOCTORAT DELIVRE PAR L'ECOLE CENTRALE DE LILLE

Optimisation avancée pour la recherche et la composition des itinéraires comodaux au profit des clients de transport

Soutenue publiquement le 02 décembre 2015 devant le jury d'examen :

Président :	Lionel AMODEO, Professeur, Université de Technologie de Troyes
Rapporteur :	Olivier GRUNDER, MCU HDR, UTBM
Rapporteur :	Xiaolan XIE, Professeur, ENSMSE (Mines Saint-Etienne)
Directeur :	Slim Hammadi, Professeur, Ecole Centrale de Lille
Co-Directeur :	Khaled Mesghouni, MCU HDR, Ecole Centrale de Lille

Thèse préparée dans le laboratoire CRIStAL UMR 9189 à l'Ecole Centrale de Lille
Ecole Doctorale SPI 072 (EC Lille)
PRES Université Lille Nord-de-France

Remerciements

Je tiens tout d'abord à remercier profondément le Professeur Slim Hammadi pour m'avoir accueilli et d'avoir été mon directeur de thèse, ses grandes qualités humaines, ses conseils tout au long des trois années qu'a duré cette thèse, son encouragement ainsi que son savoir et de son expertise m'ont permet de mener à bien ce travail.

J'exprime sincèrement ma gratitude à mon co-directeur de cette thèse M. Khaled Mesghouni, Maître de Conférences (HDR) à l'École Centrale de Lille, pour m'avoir soutenu continuellement et m'avoir expliqué dans les moindres détails les subtilités de ce sujet de recherche. Ses connaissances scientifiques et ses qualités humaines sont très présentes tout au long de l'élaboration de ce travail de recherche.

Je remercie particulièrement M. Olivier GRUNDER et le Professeur Xiaolan XIE d'avoir accepté d'être rapporteurs et consacré un temps important à la lecture de cette thèse avec les remarques précieuses.

Je remercie également le Professeur Lionel Amodeo, de m'avoir honoré en acceptant d'être examinateur de cette thèse.

J'adresse des remerciements particuliers au personnel de CRIStAL et de l'Ecole Centrale de Lille pour l'ambiance détendue.

Je remercie mes amis les plus anciens comme les plus récents pour leur soutien moral qui m'a aidé à résister surtout les moments difficiles durant toutes ces années.

Table des matières

Table des figures

Liste des tableaux

Abréviations

GES	**G**az à **E**ffet de **S**erre
SI	**S**ystèmes d'**I**nformation
SIAM	**S**ystème d'**I**nformation d'**A**ide à la **M**obilité
SMA	**S**ystème **M**ulti-**A**gents
FIFO	**F**irst **I**n, **F**irst **O**ut
CH	**C**ontraction **H**ierarchies
IAD	**I**ntelligences **A**rtificielles **D**istribuées
AGs	**A**lgorithmes **G**énétiques
SGBD	**S**ystème de **G**estion de **B**ase de **D**onnées
JADE	**J**ava **A**gent **D**evelopment **E**nvironment

Notations

Définitions des graphes

G	Graphe
V	Ensemble de nœuds de G
A	Ensemble d'arcs de G
$\|V\|$	Cardinal de V
$\|A\|$	Cardinal de A
$c(u, v)$	Poids d'arc (u, v)
O	Nœud d'origine
D	Nœud de destination
Q	Queue de priorité pour l'Algorithme Dijkstra
$l(O, D)$	Distance entre O et D
$c(u, v, t)$	Coût d'arc (u, v) pour un départ de u à v à l'instant t
$G_m = \{V_m, A_m\}$	Graphe unimodal de mode m
$G_M = \{V_M, A_M, M\}$	Graphe multimodal avec l'ensemble de modes M
$G_{Tr} = \{V_{Tr}, A_{Tr}, C\}$	Graphe de transfert
C	Ensemble de composants du graphe de transfert
H_G	Graphe hiérarchisé de G
$\mathscr{M}_k$	Module de niveau k
M_f	Nœud remplaçant pour la recherche avec l'étiquetage vers avant
M_b	Nœud remplaçant pour la recherche avec l'étiquetage vers arrière

Problème d'affectation

(a, b)	Section de route allant de a à b

I	Ensemble des requêtes concernées
$\mathscr{V}$	Ensemble des véhicules concernés
C^j	Capacité de véhicule V_j
x_i^j	Variable de décision d'affectation entre I_i et V_j
T_d^j	L'instant où le véhicule V_j est sur le nœud d
L^j	Le nombre de passagers dans le véhicule V_j à l'issue de l'affectation
l^j	Le nombre de passagers dans le véhicule V_j au début de l'affectation
$[\alpha_d, \beta_d]$	La fenêtre horaire pour le nœud d
M	Matrice d'affectation

Algorithmes évolutionnistes

P_c	Probabilité de croisement
P_m	Probabilité de mutation

Modèle de transit

t_s	Temps de sortie
t_c	Temps de correspondance
t_a	Temps d'accès
t_{tr}	Temps de transit
(m_f, m_t, v_f, v_t, c)	Connexion de transit
$\gamma_{(i,j)}^{(k,l)}$	Transit de v_i^k à v_j^l

Coalition

$A = \{a_1, a_2, \ldots, a_n\}$	Ensemble des agents
$K = \{k_1, k_2, \ldots, k_m\}$	Ensemble des tâches
$s_{a_i} = \{s_{a_i}^1, s_{a_i}^2, \ldots, s_{a_i}^p\}$	Ensemble des ressources d'agent a_i

Ce travail est dédié

À mes chers parents
À mon épouse

Introduction Générale

Contexte et Problématique

L'augmentation de l'attractivité des différents pôles économiques autours des métropoles engendre une croissance de la demande de transport, celle-ci a conduit inévitablement à une croissance dans l'utilisation des véhicules personnels, ceci a pour effet, des nuisances sonores, sociales et environnementales. Afin de répondre favorablement à cette demande de transport de plus en plus croissante toute en limitant l'impact de ces déplacements sur l'environnement, il faut être en mesure de promouvoir une mobilité durable en proposant des alternatives aux modes traditionnels (utilisation seule du véhicule individuel) et de pouvoir également modifier les habitudes des usagers. Partant du constat que, le transport public pourrait être insuffisant pour répondre à l'ensemble de la demande, il est impératif de faire coopérer plusieurs modes de transport. L'objectif est donc de faire collaborer les modes de transports publics avec d'autres modes comme le covoiturage et/ou l'autopartage afin de proposer aux usagers un itinéraire cohérent en fonction de leur demande, ceci permettrait d'offrir un choix de déplacement à chacun, en fonction de critères environnementaux, économiques, sociaux ou autres ; c'est ce qu'on appelle déplacement comodal.

Vu la quantité limitée de véhicules partagés et des places disponibles, il est important de pouvoir planifier les itinéraires qui seront proposés aux usagers, ceci fait émerger pour nous un sous problème qui concerne l'affectation des ressources limitées. Dans cette optique, nous nous concentrons sur la mise en place d'un système d'information d'aide au déplacement des voyageurs en intégrant le transport en commun, le covoiturage et l'autopartage. De plus, notre travail s'intéressera aux modèles de données du réseau de transport ainsi qu'aux algorithmes de planification d'itinéraire basés sur ces derniers.

Objectifs et Contributions

Le système que nous allons développer devrait pouvoir gérer une multitude de requêtes simultanées sollicitant un réseau de transport aux moyens limités, nous devons alors, résoudre le problème d'une manière globale, il est nécessaire de prendre en compte conjointement les deux aspects importants dans ce type de problème ; la comodalité et

l'optimisation multiobjectif. Cependant, dans le cas où nous avons une seule requête, un algorithme exact est appliqué pour résoudre le problème ainsi posé.

Du point de vue modèle et algorithme, il subsiste des soucis qu'il convient de lever notamment, pour la partie planification d'itinéraires dans un réseau de transport intégrant les transports en commun couplés avec les modes émergents tels que l'autopartage et le covoiturage, dans ce qui suit nous exposons un certain nombre de verrous scientifiques et notre manière de les résoudre :

— **Modélisation du réseau multimodal**. Les données brutes disponibles sur les réseaux de transports ne peuvent être utilisées pour résoudre le problème de la planification d'itinéraires. Il faut absolument mettre en place toute une procédure spécifique au modèle du réseau afin d'obtenir la structure de données appropriée qui permettra à l'algorithme de planification de les utiliser d'une manière correcte. Malheureusement, la transposition des algorithmes fonctionnant sur les réseaux unimodaux ne peuvent être utilisés pour le cas qui nous intéresse ; réseau multimodal en cause, la complexité et l'hétérogénéité de ce dernier. L'algorithme de la planification d'itinéraire et la modélisation du réseau dépendants l'un de l'autre.
— **Intégration de données et systèmes hétérogènes**. La planification d'un itinéraire multimodal nécessite des données venant de plusieurs opérateurs ou autorités de transport différents, il est donc obligatoire afin de pouvoir proposer des solutions d'itinéraires multimodaux d'accéder en temps réel aux différentes données réelles de chaque réseau.
— **Perturbation du réseau et gestion des ressources disponibles**. Pour un système réel couvrant une grande agglomération, les requêtes peuvent être multiples et simultanées. Dans un réseau où les ressources de transport sont limitées surtout en cas de perturbation, l'allocation optimale des offres limitées aux demandes est aussi un défi qu'il faut surmonter.

Dans cette thèse, nous nous sommes attelés à proposer des solutions pour le problème de planification collaborative d'itinéraire. L'approche de résolution que nous proposons fait émerger plusieurs aspects de recherche que nous résumons ci-dessous.

— **Modélisation du réseau de transport multimodal**. Dans cette optique, nous devons considérer les nœuds de transit dans lesquels les voyageurs seront invités éventuellement à changer de mode de transport. Ces nœuds jouent un rôle très important dans cette modélisation et sont considérés comme des éléments centraux pour la constitution d'itinéraire multimodal.
— **Formalisation et résolution du problème de la planification d'itinéraire multimodal**. Sur la base du modèle de réseau multimodal et de la structure de données, il faudra formuler le problème en précisant ses entrées et ses sorties. Les

méthodes visant à résoudre le problème défini doivent être compatibles avec le modèle et la structure du réseau et doivent fournir des solutions viables.

— **Formalisation et résolution du problème de la planification d'itinéraire multimodal en cas de perturbation du réseau**. Il est évident qu'un réseau de transport est soumis aux aléas de fonctionnement au quotidien (pannes, imprévus, retard, etc.), notre approche devra prendre en compte cet aspect temps réel de fonctionnement concernant les perturbations possible sur le réseau. Il faut pouvoir répondre à la question : Comment satisfaire au mieux les demandes des utilisateurs avec une offre limitée en ressource de transport ? Nous devons formuler le problème d'allocation avant de proposer les approches adaptées pour sa résolution. Le problème multimodal nécessite une étape supplémentaire permettant de relier l'ensemble des tronçons formant un trajet complet. Il est alors nécessaire de prendre en compte chaque perturbation dans chaque tronçon afin de proposer une solution optimale pour la demande en fonction de l'offre disponible et ceux pour chaque tronçon de route.

— **Une alliance entre SMA et optimisation**. Comme l'organisation des données des différents modes de transport intégrés ne sont pas centralisée, l'approche basée sur les Systèmes Multi-Agents (SMA) est adoptée pour ce problème d'optimisation distribuée. Toujours dans le but de l'élaboration d'un système temps réel, performant et compétitif à grande échelle, nous nous concentrons sur l'alliance des SMA et de l'optimisation. La mise en place de cette alliance permet d'intégrer le sens d'optimisation au sein des agents par les conciliations de leurs comportements pour la recherche des solutions optimales. En outre, la coalition des agents au sein des SMA, qui représentent les tronçons de route, relie les différents services de transport pour assurer un trajet complet.

— **Implémentation et évaluation de l'approche proposée**. Les fondements théoriques peuvent aussi être vérifiés par la voie expérimentale. La mise en pratique des approches proposées permet de résoudre les problèmes réels et de fournir des scénarios qui seront appliqués et qui permettront leurs évaluations.

Organisation du mémoire

Ce rapport de thèse se divise en cinq chapitres qui sont résumés ci-dessous.

Chapitre 1 : Mode de transport et comodalité, contexte et présentation du problème. Nous commençons par présenter les transports en général en France, puis les modes de transport en soulignant la notion de véhicule partagé et son intégration

au réseau multimodal. Par la suite, nous présentons un état de l'art des systèmes d'information de transport. Ensuite, nous définissons la problématique traitée ainsi que la stratégie de résolution.

Chapitre 2 : Graphe, système multi-agents et optimisation ; au service d'un système de transport comodal distribué et efficace. Le deuxième chapitre présente les fondements théoriques de la problématique traitée. Nous commençons par présenter la théorie des graphes utilisée pour la modalisation, les algorithmes de recherche d'itinéraire, ainsi que les techniques d'accélération du calcul. Nous nous intéressons également aux approches multi-agents pour lesquels nous présentons un état de l'art. En fin, nous mettons l'accent sur l'alliance entre optimisation et systèmes multi-agents.

Chapitre 3 : Planification d'itinéraire et problème d'affectation au sein d'un système de transport comodal. La modélisation du réseau de transport comodal est présentée en premier lieu. L'algorithme de planification d'itinéraire est développé en utilisant des heuristiques manipulant de fenêtres horaires. Il s'en suit, la résolution à l'aide des algorithmes évolutionnistes du problème d'affection des voyageurs aux véhicules sur les sections de route concernées, ce dernier est formulé comme un problème d'optimisation multiobjectif.

Chapitre 4 : Architecture multi-agents et mise en œuvre des coalitions d'agents pour la composition des itinéraires. Dans la quatrième partie, nous présentons dans un premier temps l'architecture du système proposé ; à savoir un système basé sur les multi-agents. Dans un second temps, les agents composant cette architecture sont définis avec des détails sur leurs comportements. Puis, après avoir obtenu les résultats d'affectation représentés par les agents communicants, nous établissons le modèle de la coalition de ces entités avec les protocoles et les interactions pour reformuler les itinéraires.

Chapitre 5 : Implémentation et déploiement de la solution proposée. Le cinquième et dernier chapitre concerne l'implémentation et le déploiement de l'approche proposée, les architectures logicielles avec leurs outils utilisés sont présentées. Les différents scénarios d'application sont également exposés.

Conclusion générale. Nous finissons ce rapport de thèse en présentant un bilan de nos travaux de recherche et nous développerons quelques pistes pour nos travaux futurs en guise de perspectives.

Chapitre I

Mode de transport et comodalité, contexte et présentation du problème

Sommaire

I.1 Introduction

Dans ce chapitre, nous présentons le problème de transport comodal utilisant également les véhicules partagés au même titre que le transport en commun. Nous donnons un panorama du transport en France et les moyens déployés, nous expliquons également la notion de véhicules partagés : le covoiturage et l'autopartage. Un état de l'art concernant les systèmes existants est réalisé. Nous finissons par décrire et formuler notre problème ainsi que le plan de résolution mis en place.

I.2 Les transports en France

La France dispose d'un réseau de transport assez dense, regroupant le réseau routier, ferroviaire, métropolitain, maritime, etc. L'utilisation de véhicule personnel est plébiscitée, en effet, 60% des déplacements urbains se font avec ce mode de transport. L'impact environnemental ainsi que l'engorgement du réseau routier sont devenus une préoccupation majeure pour les autorités, qui essayent de réduire, ou voir, mieux utiliser ce moyen de transport.

I.2.1 Impacts écologiques

Le secteur de transport occupe le triste record du premier émetteur de gaz à effet de serre (GES), avec 27% des émissions de CO_2, loin devant l'agriculture (18,2%), de l'industrie manufacturière et construction (13,0%) ou encore l'industrie de l'énergie (11,7%). En l'espace de deux décennies (de 1990 à 2012), les émissions de GES dans le domaine des transports en France ont augmenté de 9%. [1]

Si on analyse les émissions de GES concernant le transport, on constate que 21,5% des émissions sont imputables au transport de marchandises utilisant les poids lourds et seulement 3,5% sont dues au transport aérien. Il en découle que le plus gros responsable est le véhicule personnel avec 53,1% des émissions. Ramené à l'échelle nationale et sur l'ensemble des causes, le véhicule personnel contribue à hauteur de 20% du total des émissions des GES (statistiques établies en 2012).

1. Service de l'Observation et des Statistiques (SOeS) du commissariat Général au Développement Durable (CGDD)

I.2.2 Évolution des transports

Le déplacement des voyageurs en France, comme en Europe, ne cesse de croître. En effet, la distance parcourue pour les trajets quotidiens ou fréquents (par exemple : déplacements domicile-travail) et les trajets touristiques ont considérablement augmenté. La voiture particulière a vu son utilisation augmentée et encouragée par le pouvoir public depuis des décennies, quand bien même que l'acquisition et l'entretien d'un véhicule privé coûte de plus en plus cher. Selon l'étude de l'INSEE [2], la possession annuelle d'un véhicule personnel coûte 4 273 euros en 2004, et l'utilisation des routes coûte 26 milliards d'euros [3]. La Figure I.1 montre l'évolution des voyageurs en France. La proportion en véhicules privés se stabilise à 83% alors que 985 milliards de voyageurs-kilomètres (l'unité "voyageur-kilomètre" équivalente au transport d'un voyageur sur un kilomètre) sont réalisés en 2012 avec une augmentation de 1,3% par an depuis 1990.

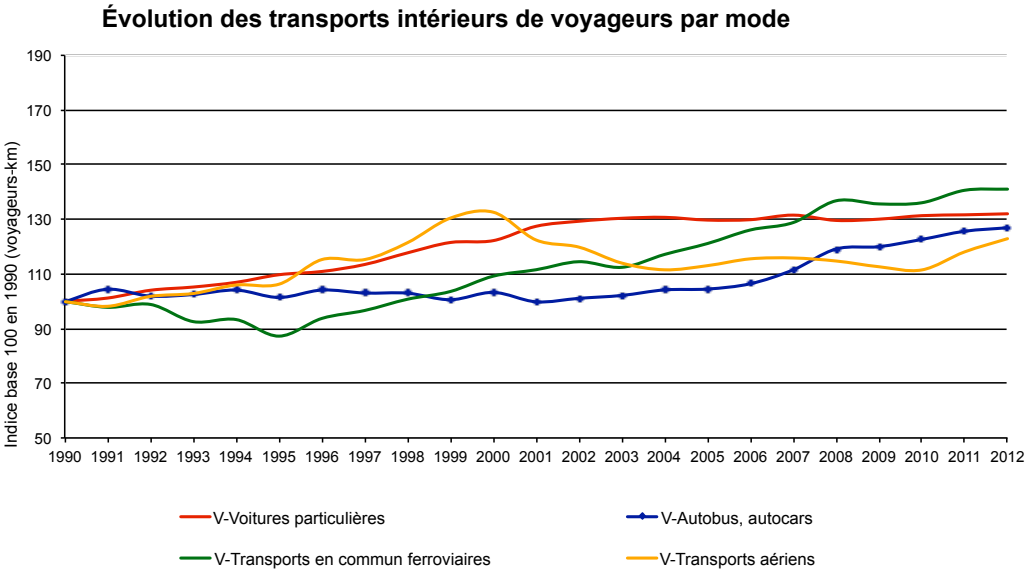


FIGURE I.1: Évolution des transports intérieurs de voyageurs par mode [4]

Cette excessive utilisation du véhicule personnel a inévitablement conduit à une consommation importante de l'énergie dont 97% est d'origine fossile (produits pétroliers) [Commission, 2009].

2. http://www.insee.fr/fr/ffc/docs_ffc/ip1039.pdf
3. http://www.rprudhomme.com/resources/2009+co$C3$BBts+de+la+route+$28Transports$29.pdf
4. http://www.statistiques.developpement-durable.gouv.fr/indicateurs-indices/f/2096/0/modes-transports.html

I.2.3 Vers une mobilité durable

L'impact environnemental de notre manière de se déplacer n'est plus à ignorer, les autorités publiques sont de plus en plus sensibles et encouragent les initiatives innovantes dont l'objectif est de réduire cet effet négatif sur l'environnement. Parmi celles qui nous intéressent, ceux qui consistent à développer des plateformes logicielles adaptées afin d'offrir des solutions plus économiques et plus souples. Certaines plateformes intègrent des services effectués par différents exploitants de transport. L'offre permet sur des critères variés, de planifier des itinéraires et de les proposer aux usagers afin d'offrir des solutions de déplacement dans le cadre d'une mobilité durable.

Parmi les objectifs de la stratégie nationale du développement durable [5], à l'échelle nationale, est de réduire les déplacements contraints et de développer des systèmes innovants répondant aux besoins de performances économiques, écologiques et de cohésion sociale.

Dans nos travaux, nous traitons le problème de la planification d'itinéraire en proposant une coalition de plusieurs modes : les transports en commun et les véhicules partagés comme le covoiturage et l'autopartage.

I.3 Différents types de modes de transport

Les moyens de transport peuvent être classés selon différents critères : urbain ou extra-urbain, selon le type de fonctionnement ; privé ou public, selon leurs niveaux et types de service, ou encore, selon le type de véhicule utilisé. L'hybridation des différents moyens de transport pour effectuer un déplacement est favorisée en vue de profiter de la complémentarité, de la flexibilité et de souplesse de chaque moyen.

I.3.1 Différents moyens de transport

Les transports publics sont proposés par la collectivité et ils offrent des services ouverts au grand public. Les informations le concernant sont disponibles et accessibles par tous. Pour le service public régulier de transport de personnes, les itinéraires, les arrêts, et les horaires (tableaux de marche) sont publiés avant sa mise en service. Ce type englobe les transports urbains comme le bus et le métro, les services extra-urbains comme les trains régionaux et/ou express.

5. `http://www.developpement-durable.gouv.fr/IMG/spipwwwmedad/pdf/LPS-Bilan_SNDD-Vfinale002_cle2a3dc9.pdf` et `http://www.developpement-durable.gouv.fr/IMG/pdf/SNDD-3.pdf`

De l'autre coté, il existe des moyens de transport privés utilisés par un individu pour son propre compte. Grâce à ses nombreux avantages (simplicité et rapidité), la voiture privée est devenue une nécessité dans la vie quotidienne et souvent privilégiée pour les déplacements personnels.

En plus des transports en commun et des voitures privées, de nouvelles utilisations de la voiture, basées sur la notion de véhicules partagés, telles que le covoiturage et l'autopartage ont vu le jour. Ces moyens de transport sont considérés comme complémentaire aux modes précédemment cités.

I.3.2 Différents modes

Selon [Van Nes, 2002], le terme *mode* peut avoir divers sens selon le contexte d'usage. Au niveau du service, une distinction entre le mode privé et le mode public peut être identifiée, le premier peut être représenté par le covoiturage, l'autopartage ou encore le vélo, et le second correspond au train, bus ou encore le métro. Notre sujet de thèse concerne les services de transport, le terme "mode" pour nous, sera associé au type de véhicule. Pour les services publics de transport, les caractéristiques des véhicules comme la vitesse, l'accessibilité et le coût sont fortement liés au réseau qu'il soit urbain, régional et/ou national. Pour le transport privé, le réseau peut se diviser en urbain et extra-urbain. La Figure I.2 montre les distinctions entre les différents modes concernés dans cette thèse.

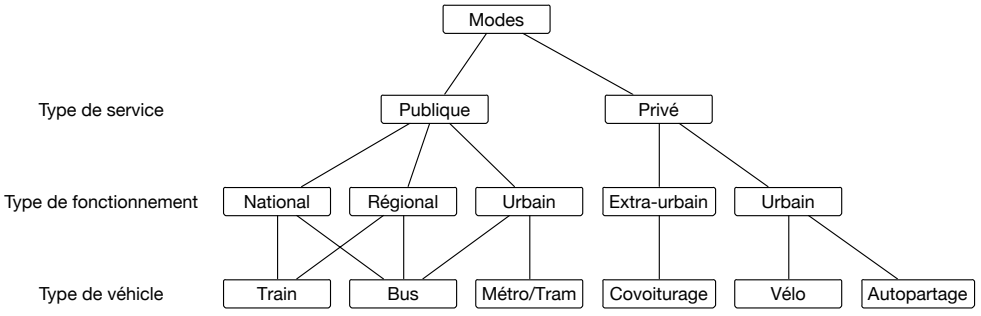


FIGURE I.2: Différents types de modes (modifié de [Van Nes, 2002]).

I.3.2.1 Transport monomodal

Le déplacement réalisé en transport monomodal s'effectue en utilisant un seul mode de transport tout au long du trajet. Même s'il ne s'agit qu'un seul type spécifique de véhicule, le transfert peut avoir lieu avec ce moyen de transport (par exemple Métro-Métro, Train-Train).

I.3.2.2 Transport multimodal

La multimodalité concerne toute offre de transport faisant usage de plusieurs moyens de transport pour un déplacement d'une origine vers une destination. Comme il n'existe pas de réel consensus concernant la définition exacte, certaines chercheurs préfèrent employer le terme *intermodal* au lieu de multimodal. Un itinéraire multimodal est donc un itinéraire combinant différents modes de transport tels que le métro, le train, le bus, le vélo, ou encore, la marche à pied. Différentes combinaisons de plusieurs moyens peuvent coexister pour effectuer un trajet selon des critères imposés. L'itinéraire ainsi construit sera proposé au décideur (par exemple le voyageur) qu'il pourra accepter ou non. La difficulté de construire un tel itinéraire vient de l'obligation de la prise en compte des différents modes de transport pour la planification de ce dernier, sans pour autant oublier les contraintes temporelles représentées par les fenêtres de temps pour le départ et/ou pour l'arrivée qui rajoutent une couche de difficulté au problème.

I.3.2.3 Transport comodal

Étant défini comme "la combinaison optimale des différents modes sur la chaîne de transport ", il correspond à "l'utilisation intelligente et efficace des différents modes de transport afin de profiter des avantages de l'un et de l'autre ", ce terme de transport comodal est introduit en 2006 par la commission européenne [officiel de l'Union européenne, 2007]. Il s'agit donc d'un mode particulier adoptant le transport sous toutes ses formes en vue de présenter un service flexible offrant un maximum de souplesse à l'usager. Dans ce contexte, les combinaisons réunissant les moyens de transport en commun (Train, Métro, Bus) et privés (véhicule personnel) sont envisagées. En opposition de l'intermodalité (multimodalité) qui incite à utiliser plutôt les transports publics donnant lieu de ce fait, à une sorte de concurrence entre les services de transport se basant sur l'une ou l'autre des offres collectives ou privée [of the european communities, 2001]. La comodalité vient par ailleurs contrer cette rivalité pour imposer et instaurer un certain équilibre mettant en exergue la complémentarité entre les modes de transports collectifs et privés [Gille, 2006]. Selon [Giannopoulos, 2008], l'approche comodale consiste à développer des infrastructures et agir avec des actions qui assurent une combinaison optimale des différents modes de transport. Les systèmes de transports comodaux visent à assurer et à fournir un service de qualité garantissant son optimalité économique et environnementale.

I.4 Notion de véhicule partagé

Une pratique pour réduire le nombre de voitures en circulation est de rendre les moyens de transport initialement conçus pour une utilisation individuelle, collectifs mettant en évidence la notion de partage de véhicule (services de covoiturage, autopartage...). Les systèmes basés sur cette notion ont vu le jour suite au développement de plusieurs travaux dans ce domaine de transport. Nous citons essentiellement deux catégories selon leur fonctionnement :

- **Autopartage** : un véhicule de ce type est partagé par période de temps par des utilisateurs différents à chaque fois.
- **Covoiturage** : un véhicule est partagé pendant un même intervalle de temps par plusieurs personnes pour effectuer une partie ou la totalité d'un trajet commun.

I.4.1 Autopartage

L'autopartage ("Carsharing" ou "car clubs" à l'anglaise) est un service de mobilité proposé soit par une société ou par une agence publique. Ce système nécessite un adhésion préalable au service, les clients peuvent effectuer des réservations avant chaque prise de véhicule qui est mis à disposition [Clavel et al., 2008]. Au lieu de disposer d'une voiture personnelle et d'un garage ou une place de stationnement privée, les membres d'un service d'autopartage partagent ainsi l'usage d'un parc de véhicules.

I.4.1.1 Mode d'emploi d'autopartage

L'autopartage est un système de transport personnel accessible à toute personne possédant un permis de conduire. Les abonnés au service font une réservation obligatoire soit par téléphone, ou par Internet, ou par tout autre moyen avant de se rendre aux emplacements prévus pour récupérer le véhicule. Plus précisément, l'ensemble des véhicules de la flotte est disponible à une prise autonome, l'utilisateur possédant une carte à puce délivrée lors de l'abonnement au service et permettant soit d'ouvrir un boitier contenant les clés ou déverrouiller directement le véhicule via un ordinateur embarqué [Clavel et al., 2008].

Concernant la tarification de ce service, outre les frais d'abonnement, les véhicules en libre-service sont à des tarifs attractifs basés sur l'utilisation réelle et incluant le carburant, l'assurance et la maintenance. Concrètement, le tarif dépend de la durée d'utilisation et de la distance parcourue. De plus, ce mode de transport permet aux clients de bénéficier de la gratuité des parkings. Dans la plupart des cas, il est plus avantageux

pour les individus d'utiliser cette formule qu'acquérir un véhicule personnel (prix de véhicule, assurance, carburant,...).

I.4.1.2 Systèmes existants d'autopartage

Crée en 1948 à Zurich (Suisse), le SEFAGE est le premier club à but non lucratif avoir pratiqué l'autopartage, il s'agit d'un club de conducteurs dans lequel les membres cotisés pour acquérir un véhicule. Après cette première expérience sans trop de suite, l'autopartage a connu un grand développement lors de ces dernières années, des systèmes analogues au premier ont vu le jour un peu partout dans le monde pour exploiter cette idée.

En France, de nombreux services d'autopartage ont été mis en place et déployés au cours de ces dernières années. Nous citons ici quelques uns comme Lilas à Lille, Autolib' à Paris, citizprovence en Provence, Mobilib à Toulouse. Le service d'Autolib' de Paris, qui existe depuis trois ans, connaît aujourd'hui un incontestable succès. Disposant de 2600 voitures électriques stationnées dans 900 parkings dédiés, ce service d'autopartage est très utilisé, il compte plus de 66000 abonnés actifs qui réalisent 14000 locations quotidiennes pour rouler 9 kilomètres pendant 36 minutes. Au niveau mondial, divers systèmes existent, nous citons par exemple, CityCarClub en Grande-Bretagne, Communauto au Canada et mobility qui occupe le plus grand marché en Suisse avec plus de 800000 abonnés et 10000 véhicules à travers 470 villes, et en fin, ZipCar aux États-Unis qui est le premier acteur mondial.

I.4.1.3 Utilisation combinée avec le transport en commun

Pour obtenir une mobilité combinée plus étendue et plus flexible, il est important de développer de fortes liaisons entre les opérateurs de transport en commun et ceux d'autopartage. Localement, les opérateurs du système d'autopartage donnent la possibilité de combiner les abonnements à leurs produits et ceux des transports en commun. Ces abonnements combinés, comme celui de Lilas, qui offre une réduction de 50% sur l'abonnement si l'utilisateur a déjà un abonnement transport en commun. La mise en place de cette pratique dans les villes disposant d'un service d'autopartage encourage l'utilisation de ce service émergeant.

À l'échelle nationale, pour un trajet qui se fait principalement en train, le dernier tronçon entre la gare et la destination finale peut être effectué en autopartage avec le service de réservation d'un véhicule proche de la gare, en cas de perturbations ou de la non

déserte de la destination finale par les transports en commun. Cette utilisation combinée transport en commun et autopartage favorise l'utilisation des transports publics, permet d'augmenter la multimodalité, réalise un gain économique (en évitant les coûts d'acquisition et d'entretien d'un véhicule rarement utilisé) et un gain environnemental (réduction de la pollution et des GES).

I.4.2 Covoiturage

L'idée du mode de transport basé sur le concept de partage de véhicule privé, appelé covoiturage (carpooling ou ride-sharing en anglais), est née dans les années 50 [Fagin and Williams, 1983] [Teal, 1987]. Depuis, ce mode de mobilité n'a cessé d'évoluer car il offre un coût d'utilisation réduit (partage des frais), et permet également de réduire l'impact des déplacements individuels sur l'environnement (réduction de la pollution).

I.4.2.1 Mode d'emploi de covoiturage

Dans [Dailey et al., 1999], un trajet est défini comme un seul voyage d'un lieu géographique à un autre. La demande de chaque voyageur comprend donc deux points : l'origine et la destination. Pour un trajet de covoiturage, il faut au moins deux participants qui partage un même véhicule pour tout ou parti d'un trajet. Un trajet de covoiturage réussi nécessite donc un point "collecte" (pick-up, le lieu où le participant est récupéré) et un point de "dépôt" (drop-off, le lieu où le participant descend).

Définition : Participant de covoiturage : un voyageur qui n'est pas le chauffeur de véhicule sur le trajet de covoiturage qu'il effectue. Il cherche des offres de trajet.

Définition : Chauffeur de covoiturage : un voyageur qui est le chauffeur du véhicule sur le trajet qu'il propose pour le covoiturage. Il fournit les offres de trajet.

Selon son mode et régime, le covoiturage se divise en deux modes : organisé et non organisé (spontané).

Le covoiturage non-organisé concerne souvent les individus qui se connaissent, comme les collègues ou les amis. Ce type de covoiturage existe depuis longtemps. Les personnes sans relation directes peuvent également l'utiliser, on parle alors de covoiturage ad-hoc. À cause du manque de communication efficace, ce type de covoiturage n'est pas populaire.

Le covoiturage organisé concerne les entités intermédiaires (par exemple les agences) qui effectuent la tâche de *matching* entre les participants et les chauffeurs. Cette entité de mise en relation entre les chauffeurs et les participants peut prendre plusieurs formes :

associations et collectivités (un cadre privilégié dédié au covoiturage), centre de mobilité (structure qui rassemble les informations sur les offres de transport), entreprises ou administrations et les sites Internet de covoiturage (pour les particuliers). Grâce à ces entités intermédiaires, ce type de covoiturage possède un grand potentiel.

Définition : matching de covoiturage : La mise en relation entre le participant et le chauffeur par un intermédiaire.

Dans [Furuhata et al., 2013], 4 modèles de matching sont classifiés : le covoiturage identique, le covoiturage inclus, le covoiturage partiel et le covoiturage avec détour, comme illustre la Figure I.3.

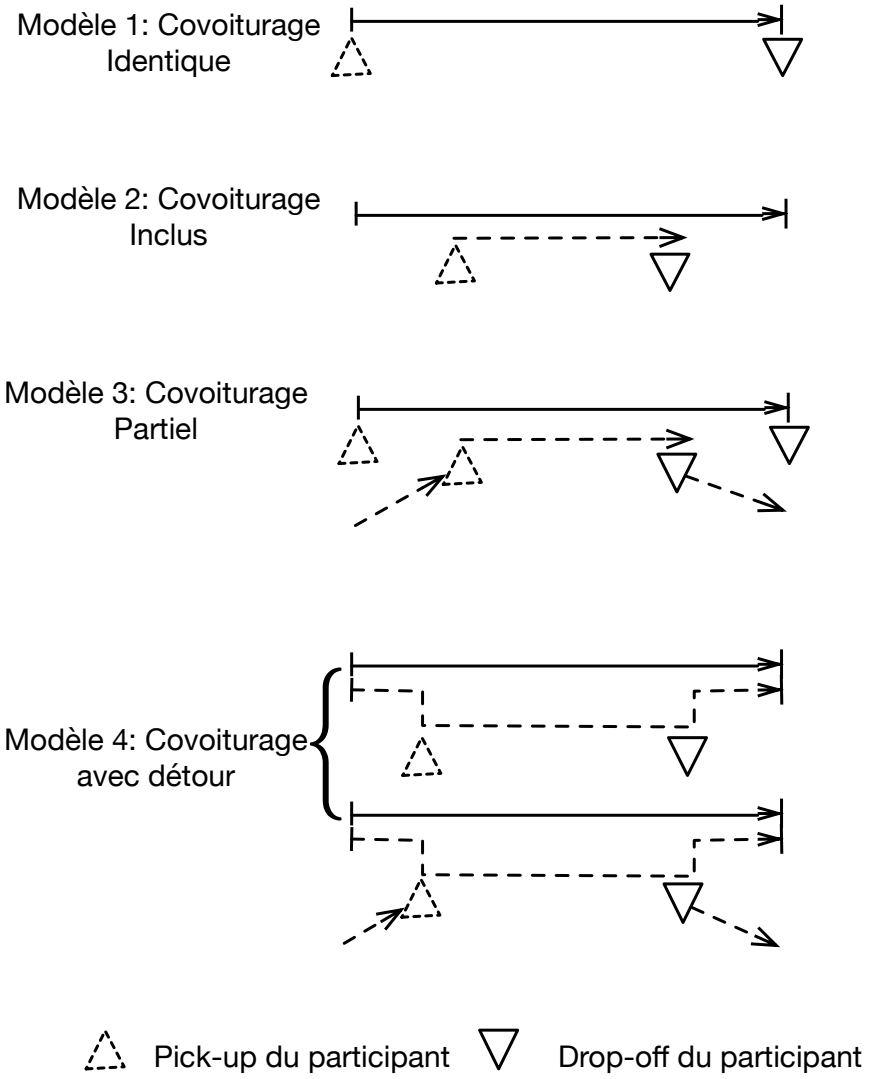


FIGURE I.3: Différents modèles de matching de covoiturage

Ces 4 modèles illustrés dans la Figure I.3 sont décrits comme suivant :

— **Modèle 1 (covoiturage identique)** : Le chauffeur et le participant ont des points de départ et d'arrivée identiques.
— **Modèle 2 (covoiturage inclus)** : Les points de départ et d'arrivée du participant sont sur le trajet initial du chauffeur.
— **Modèle 3 (covoiturage partiel)** : Les points de départ et d'arrivée du participant ne sont pas sur le trajet initial du chauffeur. Par contre, les points de pick-up et drop-off sont sur ce même trajet.
— **Modèle 4 (covoiturage avec détour)** : Les points de pick-up et drop-off ne sont pas sur le trajet initial du chauffeur.

I.4.2.2 Systèmes existants de covoiturage

Selon les gestions de matching de covoiturage, ce type de service se divise en deux catégories : le covoiturage dynamique en temps-réel et le covoiturage statique.

— **Covoiturage dynamique en temps-réel** [Agatz et al., 2012] [Deakin et al., 2010] [Amey, 2010] : ce type de covoiturage permet un matching automatique entre le chauffeur et le participant quelques minutes avant le départ. C'est une nouvelle tendance dans le domaine de covoiturage pour gérer les offres et les demandes et actualiser le matching sur une toute petite fenêtre de temps ou en temps réel. Le matching est basé sur les éléments suivants : les points de pick-up et drop-off et les heures de rendez-vous à partir des itinéraires et des horaires déclarés par le chauffeur et le participant.
Le principe de covoiturage dynamique proposé par Covivo est d'informer le chauffeur en temps réel de la position du participant potentiel ayant la même destination que lui. d'autres systèmes basés sur le même principe en temps réel existent comme GoLoco, Easy-Rider ou encore T.écovoiturage.
— **Covoiturage statique** : La majorité des approches existantes fait partie de cette catégorie. Dans le cas général, au lieu de matching sous forme automatique, l'utilisation de ce type de covoiturage nécessite une inscription préalable pour avoir accès aux différentes fonctionnalités offertes, par exemple chercher des demandes de covoiturage, et consulter les données correspondant au trajet comme le type de véhicule, le chauffeur, l'autorisation de bagage grande format, ou encore, mettre en ligne des offres de covoiturage.
Le système d'idvroom se présente sous cette forme et propose des trajets uniques ainsi que des trajets réguliers domicile-travail. Blablacar, un système de covoiturage statique, a connu un grand succès récemment.

I.4.2.3 Utilisation combinée avec le transport en commun

Pour assurer la continuité du trajet, il faut que les lieux de rendez-vous soient accessibles pour les participants et les voitures à la fois. Pour un covoiturage, il est nécessaire de bien sélectionner le lieu de drop-off pour rejoindre éventuellement les transports public.

D'une part, des panneaux dédiés au covoiturage sont installés dans certaines villes. Ces panneaux sont pris comme lieu de rendez-vous fixé entre les chauffeurs et les participants de covoiturage. Ils sont très souvent à proximité des arrêts de transport en commun, donc facilement accessibles après la descente du véhicule ou du transport en commun. Parallèlement, le stationnement temporaire est autorisé pour les véhicules qui attendent les participants.

D'autre part, il est préférable de choisir les points de pick-up et drop-off près des gares de transport en commun pour les personnes utilisant le covoiturage. C'est déjà une pratique très courante.

I.4.3 Complémentarité des moyens de transport

En ce qui concerne la flexibilité, la sécurité ou la continuité des trajets, considérer les moyens de transport chacun dans son propre contexte semble insuffisant. Les transports en commun sont préconisés pour les déplacements en ville pour des raisons de pollution et de coût. Par ailleurs, la nécessité des déplacements occasionnels hors des zones d'urbanisation et le manque de desserte régulière de ces zones, voir son absence total, imposent la voiture comme un moyen de transport essentiel. Le véhicule personnel offre à l'usager la liberté et la flexibilité souhaitée, par contre, elle induit un coût financier et environnemental.

Les transports collectifs remédient aux principaux problèmes de l'usage abusif de la voiture en créant des modes de transports économes en termes de consommation. Malgré la grande évolution, le transport en commun reste insuffisant les restrictions imposées pour son étalement géographique ainsi que les risques potentiels pouvant provoquer les perturbations sur les réseaux de transports collectifs freinent son développement.

I.5 Système d'information d'aide à la mobilité

Un Système d'Information (SI), est un ensemble organisé de ressources qui permet le regroupement, la classification, le stockage, le traitement et la diffusion de l'information sur un environnement donné.

Dans le domaine de transport, le système d'information servant à l'information des voyageurs correspond au Système d'Information d'Aide au Déplacement (SIAD). Ce dernier a pour fin d'assister les voyageurs dans leur déplacement en leur fournissant les informations dont ils ont besoin.

I.5.1 Quelques systèmes existants

Les opérateurs locaux de transports en commun fournissent leurs outils séparément. Les grandes entreprises comme Google proposent également des offres de service de planification d'itinéraire.

Nous allons examiner les services de transport en commun à Paris (France) : ratp.fr, de covoiturage en France : covoiturage.fr et Google Transit.

Lors d'une requête sur ratp.fr, les entrées sont la station, le lieu ou l'adresse de départ et d'arrivée, l'heure de départ ou d'arrivée, le jour, les préférences sur le mode et les critères. Un résultat de requête de ratp.fr se présente comme sur la Figure I.4. Les résultats montrent pour chaque itinéraire le temps de voyage, la tarification et le nombre de transfert. La durée du trajet est calculée en temps réel. La réponse à la recherche est rapide avec une bonne performance. Cependant, les résultats ne concernent que les moyens de transports de la RATP sans l'intégration des autres modes comme le covoiturage, ou le vélo en location.

FIGURE I.4: Requête de Gare du Nord à Gare de Lyon avec RATP

Le site covoiturage.fr est une plateforme qui permet de mettre en relation les chauffeurs avec les participants de covoiturage. Les résultats d'une requête se présentent comme sur la Figure I.5. Pour lancer une recherche, il faut compléter les champs suivants : lieux de départ et d'arrivée et la date. Il est possible d'ajouter des critères supplémentaires, comme la plage horaire de départ, l'expérience du chauffeur, le type de véhicule. etc. Les réponses aux requêtes peuvent être triées par heure de départ ou par coût de trajet. Il est à noter, qu'il n'est pas possible d'intégrer les autres moyens de transport avant ou après le trajet de covoiturage sur cette même plateforme.

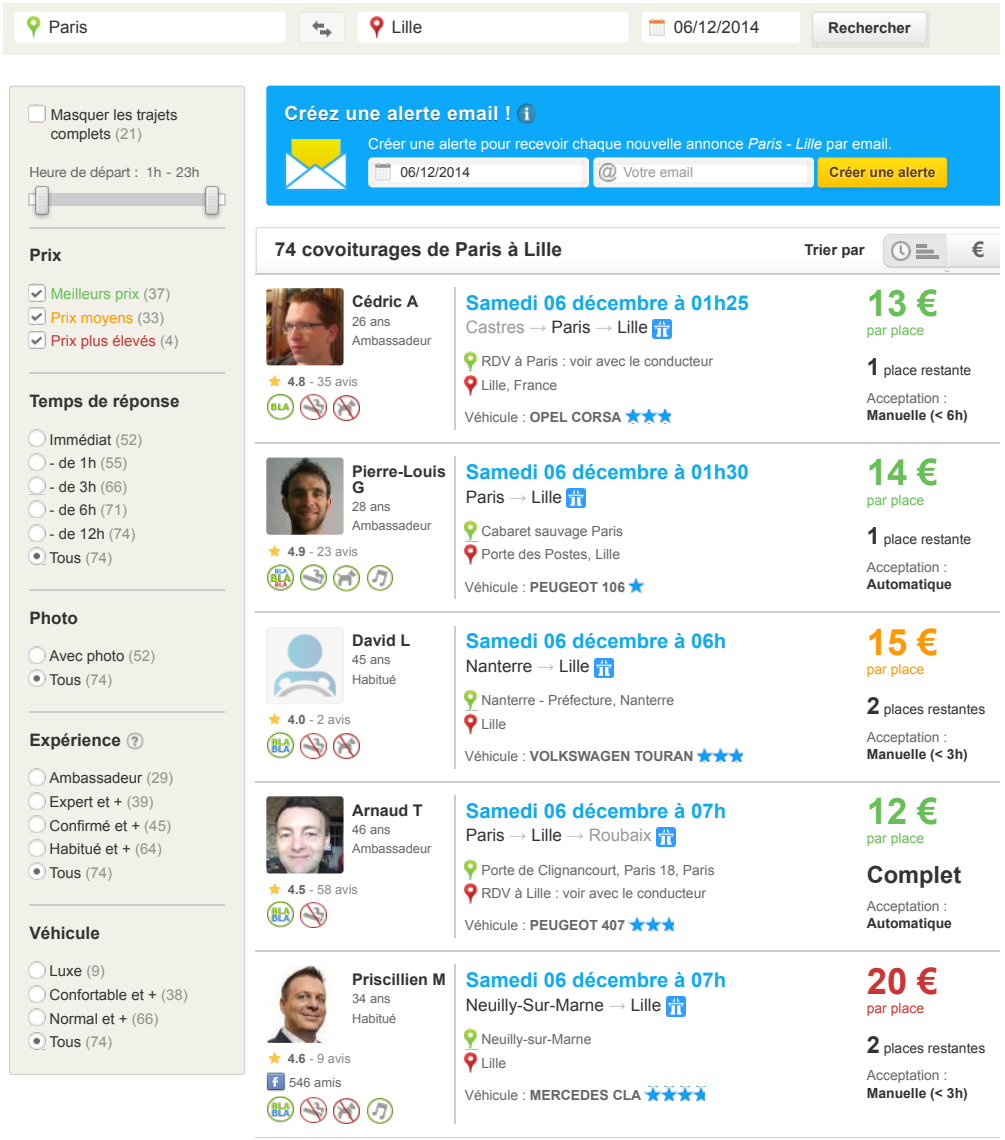
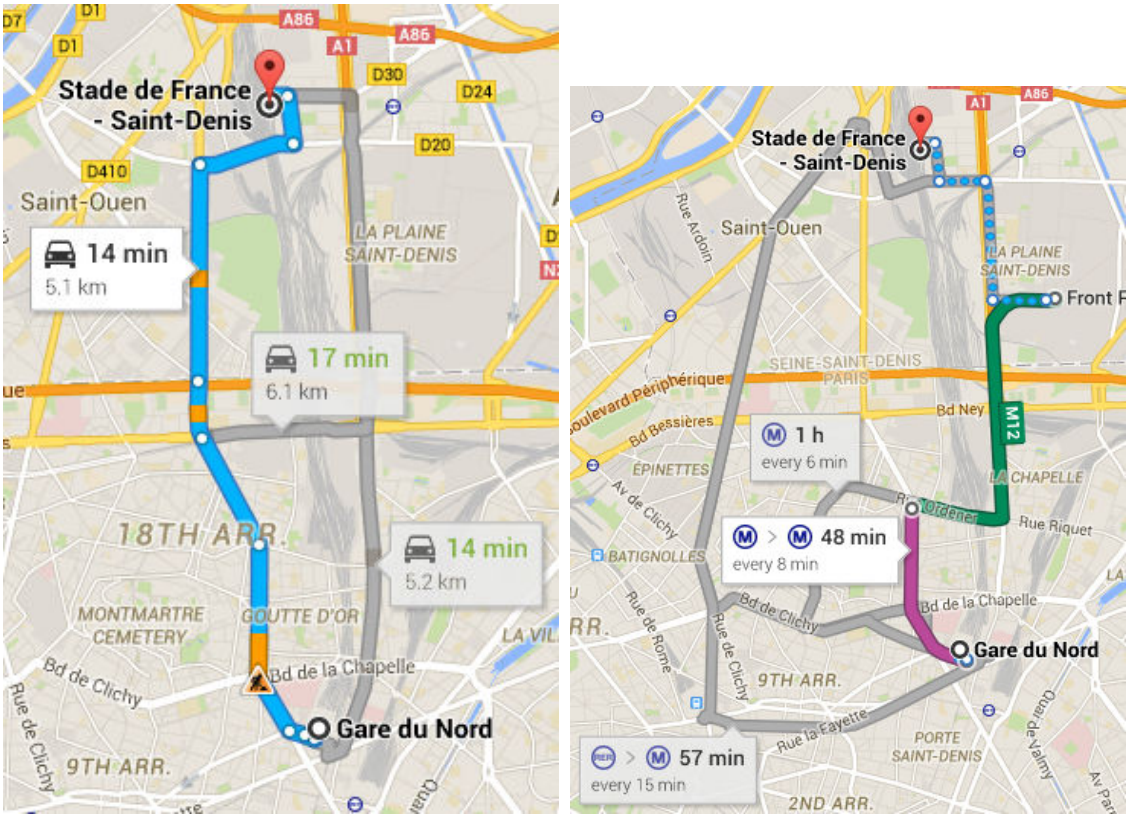


FIGURE I.5: Résultats d'une requête de covoiturage de Paris à Lille avec covoiturage.fr

Google Transit (`transit.google.fr` ou `maps.google.fr`) est un ensemble d'outils qui est disponible gratuitement dans plusieurs villes à travers le monde. Comme sur ratp.fr, les entrées de requêtes sont le lieu ou adresse de départ et d'arrivée, l'heure de départ ou d'arrivée, le jour, les préférences sur le mode de transport et sur les critères. Le

résultat de requête est visualisé comme sur la Figure I.6. Cet outil supporte à la fois les informations statiques et en temps réel du trafic public. Ce dernier concerne les éléments arrêts, routes, horaires et autres détails. Concernant la préférence de mode, le transport en commun ou le trajet avec véhicule peut être choisi, l'itinéraire de la combinaison entre ces deux modes n'est pas possible. Le véhicule partagé n'est pas non plus intégré.



(A) Requête de Gare du Nord à Stade de France avec Google Transit avec voiture privée

(B) Requête de Gare du Nord à Stade de France avec Google Transit avec transport en commun

FIGURE I.6: Requête avec Google Transit [6]

I.5.2 Système d'information d'aide à la mobilité de transport multimodal

Généralement, un système de transport multimodal correspond au système de transports en commun. Nous l'ajouterons ici pour que les véhicules partagés soient considérés lors de la planification d'itinéraire. Le covoiturage et l'autopartage sont les modes à choisir comme préférences pour les utilisateurs. Prenons en compte tous les modes, il s'agit donc de l'agrégation des systèmes des différents opérateurs pour un système d'information de transport multimodal.

6. Les images produites avec Google Maps © mapping service

Il y a deux solutions pour calculer des itinéraires au sein d'un système d'information pour le transport multimodal. La première est de rassembler les données issues de plusieurs opérateurs et leurs faire un prétraitement afin de pouvoir les utiliser pour exécuter les algorithmes de planification. Cette manière a plusieurs inconvénients dont le principal est que les données ne sont pas toutes disponibles en raison de la sécurité et de la confidentialité évoquées par les différents opérateurs. De plus, le fait que les données soient parfois dynamiques et en temps réel rend le traitement des données plus compliqué, par exemple les offres de covoiturage. La deuxième solution, calculer les itinéraires à travers les sous-systèmes intégrés par les portails qui sont des serveurs sécurisés dédiés qui renvoient les résultats sous une certaine forme. Les itinéraires sont calculés à partir de ces résultats.

I.6 La problématique à traiter

La planification d'itinéraire en transport multimodal soulève le problème de trouver les "bons" trajets à travers des réseaux de transports combinés, comprenant le transport en commun ainsi que les autres modes de transport. Un trajet où se rencontrent le transport public et les véhicules partagés peut être très attractif grâce à leur complémentarité. Actuellement, parmi les systèmes existants on ne trouve pas de systèmes où les véhicules partagés sont intégrés pour répondre aux besoins des utilisateurs. Nous décidons donc de travailler sur cet axe de recherche afin de mieux répondre aux attentes. Avec les services de transport existants, nos travaux de recherche visent à intégrer non seulement le transport en commun, mais aussi le service des véhicules partagés comprenant le covoiturage, l'autopartage ainsi que le vélo-partage (vélo en libre-service) dans l'optique de proposer une planification d'itinéraire qui vise à répondre aux requêtes avec les parcours optimaux selon les critères imposés.

Les critères pour les "bons" trajets peuvent être la minimisation du temps d'arrivée, la durée de parcours ou le coût du trajet etc. Une combinaison entre certains critères peut être aussi un critère de "bon" trajet. En effet, différents utilisateurs mettent des poids sur les critères, il faut donc donner plus de flexibilité aux utilisateurs.

Pour un scénario réel qui peut se produire au sein d'un système à grande échelle, les requêtes peuvent être multiples et simultanées. Il faut également considérer le fait que les préférences des utilisateurs ne sont pas forcément uniformes et les stocks des véhicules ou les places disponibles sont parfois limités, les solutions seront fournies en répondant aux besoins des utilisateurs et garantissant une attribution optimale des ressources de transport.

I.6.1 Une approche proposée pour résoudre le problème

La Figure I.7 montre les démarches à suivre pour le problème ciblé. Trois niveaux de planification se présentent, chacun accueille un processus d'optimisation.

Dans un premier temps, les requêtes d'itinéraires sont reçues et résolues avec l'algorithme de la planification d'itinéraire, à l'issue duquel les itinéraires sont donnés. En prenant en compte des fenêtres de temps précis au moment du lancement de requêtes, nous identifions les offres disponibles (les véhicules pour le transport public ainsi que le covoiturage, l'autopartage) sur les composants des itinéraires, donc sur les sections.

Dans le deuxième niveau, les itinéraires en sections avec les offres identifiées sont reçues de l'étape précédente. Ces offres seront affectées aux demandes concernant la même section avec un autre processus d'optimisation. Concrètement, des métaheuristiques seront mises en place pour réaliser cette étape d'optimisation.

Dans le dernier niveau, les passagers et les véhicules sont déjà mis en relation sur les tronçons de route. Afin d'avoir des itinéraires complets, nous allons recombiner ces morceaux de routes afin d'obtenir les itinéraires complets avec un processus de *coalition*.

Pour résoudre un problème d'optimisation dynamique et distribuée, nous nous permettons de profiter de l'apport des systèmes multi-agents qui ont fait leurs preuves dans ce domaine. L'une des originalité de notre travail réside dans l'union entre l'optimisation et le système multi-agents utilisé pour la résolution du problème.

I.7 Conclusion

Dans ce premier chapitre, nous avons présenté le contexte de la mobilité ainsi que la multimodalité de transport. En particulier, l'accent est mis sur la notion de véhicule partagé avec ses classifications et ses fonctionnements. La complémentarité et l'utilisation combinée potentielle de ce dernier avec le transport en commun donne une nouvelle voie pour la mobilité personnelle dans un réseau intégré. Nous nous orientons vers ce domaine pour la planification d'itinéraire.

Dans la section I.6.1, une méthode de résolution à trois étapes est proposée pour ce problème spécifique présenté ci-dessus.

Dans le chapitre suivant, nous allons présenter les généralités des graphes servant de base à la planification d'itinéraire, les algorithmes courants ainsi que les techniques d'accélération. Nous allons aussi établir le modèle de données sur lequel les algorithmes sont fondés.

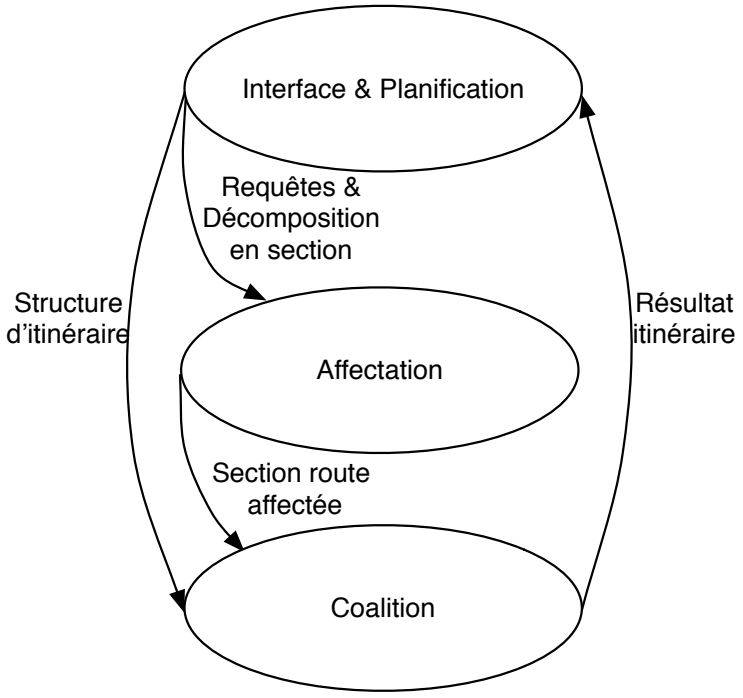


FIGURE I.7: Cadre de résolution du problème avec une procédure de trois niveaux pour les requêtes multiples simultanées

Chapitre II

Alliance entre les systèmes multi-agents et l'optimisation au profit de transport

Sommaire

II.1 Introduction

Sur la base de la problématique énoncée dans le chapitre précédent, nous présenterons dans celui-là les fondements essentiels et les techniques spécifiques pour résoudre le problème de cheminement, ainsi que les principes de la modélisation et de l'optimisation.

Ce chapitre a donc pour objectif de présenter l'état de l'art et les approches dont nous avons besoin pour résoudre le problème posé.

Nous nous concentrons fondamentalement dans ce chapitre sur les trois aspects suivants :

1. La théorie des graphes qui est l'essence de presque tous les travaux traitants de la planification d'itinéraire. Nous introduisons les algorithmes et leurs techniques d'accélération.
2. L'optimisation. À travers des concepts essentiels liés à l'optimisation, nous nous focaliserons sur le principe d'optimalité et l'optimisation multicritère ainsi que sur les métaheuristiques comme l'algorithme génétique que nous utiliserons par la suite pour résoudre le problème d'affectation.
3. Le Système Multi-Agents (SMA) qui constitue d'une abstraction en faveur de la mise en œuvre d'une optimisation distribuée. Avant de décrire son implémentation dans notre système, nous évoquerons les concepts essentiels et caractéristiques des agents, leurs capacités de communiquer et de négocier, permettant la coalition des agents auprès des protocoles spécifiques.

Ce travail s'appuie sur les éléments d'état de l'art que nous avons présenté précédemment afin de proposer un système d'information distribué de transport comodal à base d'agents communicants pour répondre efficacement aux demandes d'itinéraires et gérer les ressources de transport.

Ce chapitre se divise en trois parties : tout d'abord, nous présentons la théorie des graphes et les algorithmes de planification d'itinéraire, ainsi que les techniques auxiliaires qui permettent l'accélération de ces algorithmes. Ensuite, nous présentons l'état de l'art sur l'optimisation dans la partie suivante. La troisième partie est dédiée à la modélisation

multi-agents et ses fondements. Enfin, la dernière partie est destinée à l'alliance entre l'optimisation et SMA, mettant en exergue les avantages de notre approche.

II.2 Théorie des graphes pour la planification d'itinéraire

La théorie des graphes est considérée comme l'un des outils essentiels pour résoudre les problèmes de cheminement. Une brève introduction à la théorie des graphes et au problème du plus court chemin est exposée dans cette section.

II.2.1 Notion de la théorie des graphes

En 1736, Leonhard Euler publie un article sur le problème des sept ponts de Königsberg. Avec l'introduction d'une nouvelle structure de données appelée plus tard *graphe*, il démontre qu'il est impossible d'emprunter tous ces sept ponts une et une seule fois. Avec le développement de la théorie des graphes, cette structure de données très simple a donné naissance à de nombreuses applications dans tous les domaines liés à la notion de réseau : le transport, la planification, la télécommunication ou encore le réseau social.

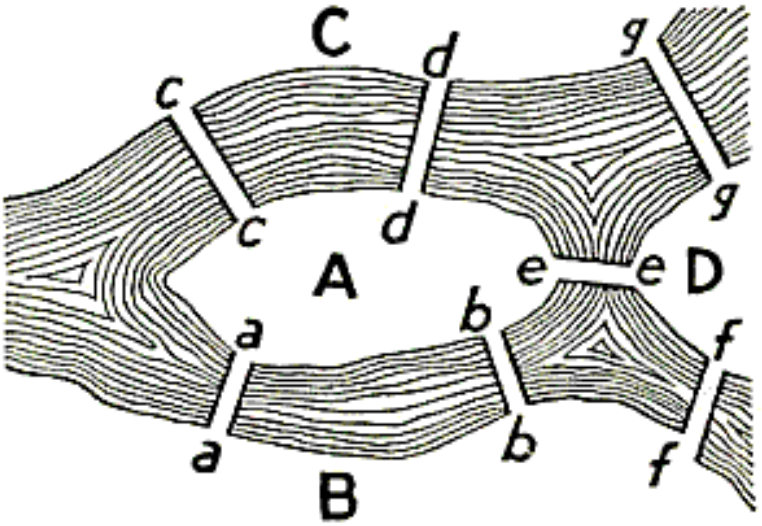


FIGURE II.1: Les sept ponts de Königsberg

Le problème du plus court chemin représente la base de tous les problèmes de planification d'itinéraire. Le plus court chemin d'un point de départ à un point d'arrivée minimisant soit la distance, soit un autre critère similaire comme par exemple le temps. Plusieurs algorithmes ont été mis au point : l'algorithme de Bellman-Ford, l'algorithme de Johnson, et le plus connu, l'algorithme de Dijkstra [Dijkstra, 1959].

Un graphe $G = (V, A)$ est un graphe avec un ensemble de nœuds noté V et un ensemble d'arcs noté A. Chaque arc $(u, v) \in A$ possède un poids non-négatif $c(u, v)$. Le coût d'un voyage d'un nœud à un autre nœud voisin peut être représenté avec ce poids. En effet, deux nœuds sont dit adjacents lorsqu'ils sont reliés par un arc. Un graphe G possédant des poids associés à chacun de ses arcs, est appelé : graphe *pondéré* (ou *valué*).

Pour un graphe pondéré G, le problème du plus court chemin est défini comme suit :

Définition. *Chemin* : Un chemin (ou itinéraire) dans un graphe est une suite ordonnée de nœuds tels que chaque deux nœuds successifs soient reliés par un arc.

Définition. *Longueur d'un chemin* : Dans un graphe pondéré, la longueur d'un chemin est la somme des poids des arcs traversés.

Pour plus de détails, nous vous invitons à consulter les documents suivants : Graph Theory [Diestel, 2005] et Introduction à l'algorithmique [Cormen et al., 1996].

II.2.2 Le problème du plus court chemin

Pour le problème du plus court chemin de type *un-à-un*, l'entrée est un graphe G, un point d'origine $O \in V$, et un point de destination $D \in V$. Le problème revient à calculer la longueur du plus court chemin de O à D dans G, représentée avec $l(O, D)$, autrement dit la distance entre O et D.

Il existe plusieurs variantes du problème *un-à-un*. Nous citons à titre d'exemple :

- Le problème *un-à-plusieurs* : ce problème consiste à calculer des distances d'un point de départ donné O à plusieurs points d'arrivées au sein du graphe.
- Le problème *plusieurs-à-un* : ce problème consiste à trouver des distances d'un ensemble de points à un point d'arrivée donné O.
- Le problème *plusieurs-à-plusieurs* : ce problème peut être formulé comme suit : pour un ensemble S de points d'origine et un ensemble T de points de destination, trouver les distances $l(s, t)$ pour $\forall s \in S$ et $\forall t \in T$. Quand $S = T = V$, le problème devient de type *tous-à-tous*.

II.2.3 Modèles de graphes

Dans le but de modéliser un réseau de transport en commun ou un réseau routier, il est nécessaire de savoir modéliser l'attente à un nœud, le poids variable en fonction du temps d'un arc. En se basant sur les tableaux horaires disponibles pour les réseaux de transports en commun, il est possible de construire un graphe $G = (V, A)$, ensuite, il

devient possible d'appliquer les algorithmes de planification d'itinéraire pour calculer le plus court chemin. Nous abordons dans cette partie trois approches de modélisation :

— *graphe condensé* ;
— *graphe dépendant du temps* ;
— *graphe temps-étendu.*

II.2.3.1 Graphe condensé

Le graphe condensé est le modèle le moins compliqué pour représenter un graphe temps-dépendent. En effet, les nœuds et les arcs ne changent pas par rapport au graphe statique. Cependant, les poids des arcs sont mis à jour comme le temps nécessaire le moins important parmi toutes les connections potentielles entre les deux nœuds liants. L'avantage de ce modèle est sa taille assez petite ainsi que l'application des techniques d'accélération pour la recherche du plus court chemin. Parallèlement, il apporte des inconvénients inévitables. En effet, les poids des arcs qui dépendent forcément des temps de départ ne peuvent pas être affichés. Par conséquent, le temps calculé entre les nœuds ne donne qu'un seuil minimum sur le temps de voyage.

II.2.3.2 Graphe dépendant du temps (*time-dependent*)

Dans le graphe dépendant du temps, les poids des arcs sont représentés par des multi-étiquettes contrairement aux graphes condensés qui le sont avec une seule. Chaque temps de départ possède sa propre étiquette. Dans ce cas les poids peuvent être indexés avec leurs temps de départ. En comparaison avec le graphe condensé, le temps de voyage est disponible pour différents départs en gardant l'avantage de la petite taille. Avec ce modèle, l'attente devient possible sur les nœuds en augmentant tout simplement les poids des arcs.

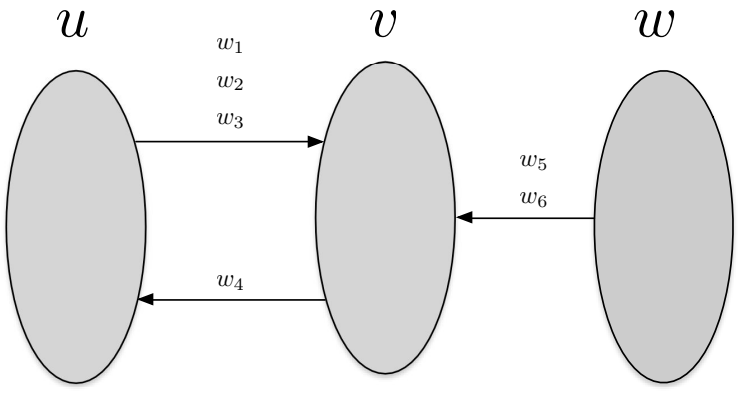


FIGURE II.2: Graphe dépendent du temps avec 3 nœuds

La Figure II.2 illustre un exemple d'un graphe dépendant du temps avec trois nœuds où les arcs possèdent une ou plusieurs étiquettes.

II.2.3.3 Graphe temps-étendu (*time-expanded*)

Dans les graphes présentés précédemment (condensé, dépendant du temps), chaque nœud correspond à une position géographique. Dans le graphe temps-étendu, les nœuds sont dédoublés et associés chacun à un instant précis. Nous considérons le fait qu'un tableau de marche d'une ligne de bus consiste en un ensemble d'événements dépendant du temps qui ont lieu sur l'axe du temps discret. Chaque départ et arrivée à différents moments est considéré comme un événement, donc un nouveau nœud est ajouté dans ce modèle de graphe pour représenter chaque événement. L'idée de ce modèle est de construire un graphe spatio-temporel (ou un graphe d'événement) [Pallottino and Scutella, 1998]. Dans ce graphe, chaque nœud est un couple d'un nœud géographique et d'un instant.

Avec le graphe temps-étendu, les événements sont reliés par des connections. Ces connections définissent les possibilités de passer d'un nœud à un autre dans l'intervalle de temps correspondant aux deux instants associés à ces deux nœuds. Dans ce cas, le graphe devient plus grand par rapport au graphe dépendant du temps, où la taille est très sensible à la fenêtre de temps considérée. Comme le nœud de la destination est divisé en plusieurs nœuds, ceci rend le problème de recherche bidirectionnelle plus compliqué.

La Figure II.3 montre un graphe temps-étendu avec 3 nœuds .

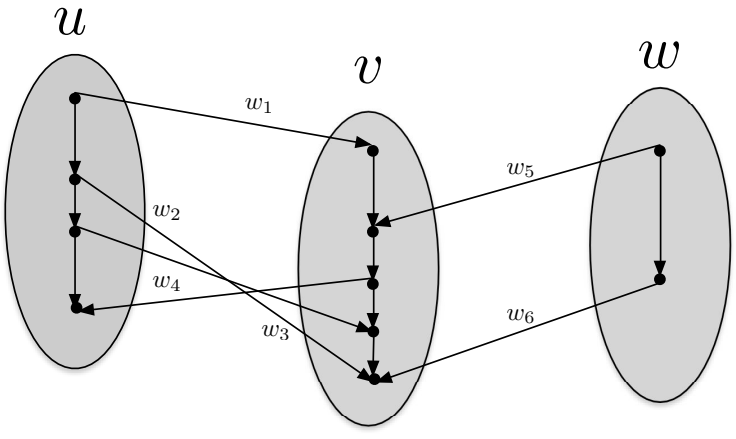


FIGURE II.3: Graphe temps-étendu avec 3 nœuds

Comme nous venons de le mentionner, la taille de ce type de graphe peut devenir de plus en plus grande, en revanche, cet inconvénient est compensé par le fait que le graphe est acyclique et possède une densité très faible.

Dans le travail de [Bast et al., 2010], ce modèle de graphe est repris pour le calcul d'itinéraires en transport en commun. Le précalcul du *transfer pattern* ayant pour but d'avoir les correspondances à utiliser entre deux arrêts indépendamment de l'heure permet d'augmenter énormément les performances. Pendant le processus de calcul de l'itinéraire, le fait que seules certaines routes sont explorées permet un calcul particulièrement rapide même avec une optimisation bi-critère. Le calcul de toutes les correspondances entre deux points nécessite trop de temps, donc les approximations légères sont appliquées avec le risques de générer des solutions sous-optimales. À titre d'exemple, Google maps a appliqué cette technique.

II.2.4 Contrainte *FIFO* ou *non-FIFO*

Pour modéliser un réseau de transport en commun ou bien un réseau routier, il faut pouvoir modéliser l'attente à un nœud avec des graphes. Pour les graphes dynamiques, nous en distinguons deux types : *FIFO* (*First In, First Out*) et *non-FIFO*. La contrainte *FIFO* implique qu'il est nécessaire d'emprunter un arc (u, v) au plus tôt pour une arrivée au plus tôt en v. Si la contrainte *FIFO* n'est pas respectée, par exemple un bus dépasse un autre entre deux arrêts, il aurait été optimal d'attendre et prendre le deuxième bus pour arriver plus tôt, c'est le cas de *non-FIFO*. Dans la suite, nous nous concentrons sur le critère de temps.

Formellement, la définition de *FIFO* pour un graphe est la suivante [Diestel, 2005] : **Définition**. *FIFO* : Un graphe $G = (V, A)$ est un graphe *FIFO* si et seulement si chaque arc (u, v) procède la propriété *FIFO*. Un arc (u, v) procède la propriété *FIFO* si et seulement si : $c(u, v, t_0) \leq t_\Delta + c(u, v, t_0 + t_\Delta)$ pour $t_\Delta \geq 0$ ou $t_1 + c(u, v, t_1) \leq t_2 + c(u, v, t_2)$ pour $t_1 \leq t_2$.

II.2.5 Algorithmes de la planification d'itinéraire

L'algorithme Dijkstra est utilisé pour trouver le plus court chemin entre deux nœuds dans un graphe pondéré non-négatif. Contrairement à l'algorithme de Dijkstra, qui ne peut être utilisé que lorsque tous les arcs ont des poids positifs ou nuls, l'algorithme de Bellman-Ford autorise la présence de certains arcs de poids négatifs. L'algorithme de Dijkstra est plus performant que celui de Bellman-Ford, il est donc à privilégier systématiquement. Ce dernier n'a pas connu de grand changement, hormis sa combinaison avec des techniques d'accélération. Actuellement, il existe de nouveaux algorithmes de planification d'itinéraire basés sur la théorie des graphes mais non Dijkstra. Ils forment l'ensemble d'algorithmes de post-Dijkstra.

II.2.5.1 Algorithme de Dijkstra

L'algorithme de Dijkstra a été proposé en 1959, il permet de déterminer le plus court chemin entre deux sommets d'un graphe pondéré dont le poids est non négatif.

L'algorithme commence par affecter à chaque nœud u un coût temporaire $l(u)$ qui est initialement 0 pour le nœud de départ et ∞ pour tous les autres nœuds du graphe. Chaque nœud a trois possibilités d'état : non-marqué, touché ou marqué. Une file de priorités avec clé $l(u)$ regroupe tous les nœuds touchés mais pas encore marqué. À chaque itération, un nœud u dans la file de priorités avec la plus petit clé $l(u)$ est supprimé. En même temps, ce même nœud u est marqué car $l(u)$ est le plus petit coût. Les arcs entre u et tous ses successeurs v sont détendus. Si la longueur du chemin via u est moins importante par rapport au coût temporaire $l(v)$, on met à jour (màj) $l(v)$ et on ajoute v dans la file de priorités. Cette mise à jour peut être une opération de type *insérer* si v est non-marqué, ou une opération de type *diminuer clé*. Le calcul se poursuit jusqu'à l'épuisement des nœuds dans la file de priorités. Le pseudo-code correspondant est illustré par l'Algorithme 1.

L'idée de base repose sur le principe de sous-optimalité. Le plus court chemin vérifie ce principe comme suit : si un nœud t est sur le plus court chemin reliant le nœud u au nœud v, alors le sous-chemin reliant le nœud u au nœud t est un plus court chemin de u à t.

Algorithm 1 Un-à-tous Dijkstra(O) (**Dijkstra**)

```
1: l ←< ∞, ..., ∞ >                          ▷ Les longueurs temporaires
2: l(O) ← 0                                  ▷ O est le nœud de départ
3: Q.update(0,O)                             ▷ la file de priorité
4: Tant que (Q ≠ ∅) répéter
5:    (·, u) ← Q.deleteMin()                 ▷ u est marqué
6:    Pour tout (u, v) ∈ A répéter
7:       Si l(u) + c(u, v) < l(v) alors      ▷ plus court ou non ?
8:          l(v) ← l(u) + c(u, v)            ▷ màj la distance temporaire
9:          Q.update(l(v),v)                 ▷ màj la file de priorité
10:      Fin si
11:   Fin pour
12: Fin Tant que
13: Sortie : Les plus courts chemins de O à tous les nœuds du graphe
```

L'exécution l'algorithme sur un graphe $G = (V, A)$ où $|V| = m$ et $|A| = n$, revient à exécuter n opérations *insérer* dans la file de priorités, n opérations *supprimer* et

m opérations *diminuer clé* dans le cas le plus défavorable. Donc l'algorithme a une complexité d'exécution $O(m + n \log n)$ avec le tas de Fibonacci [Fredman and Tarjan, 1987].

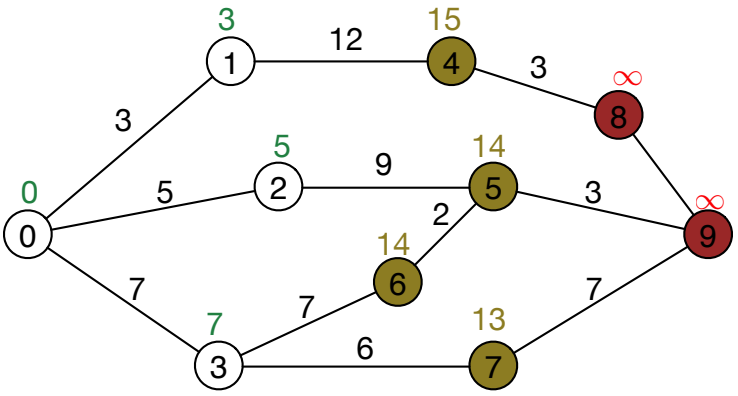


FIGURE II.4: Illustration de l'algorithme de Dijkstra. Le nœud 0 est le point de départ. Les nœuds 0, 1, 2 et 3 sont marqués, les nœuds 4, 5, 6 et 7 sont touchés et les nœuds 8 et 9 sont non-marqués.

Selon le processus de recherche, la distance temporaire est mise à jour après chaque détente et comparaison d'un arc (Ligne 8 de l'Algorithme *Dijkstra*), donc ce type d'algorithme est aussi algorithme à fixation d'étiquettes ("label-setting").

Selon les différents types de problèmes, les destinations peuvent être définies comme n'importe quel sous-ensemble de V. Une fois tous les nœuds destinataires sont marqués, l'algorithme s'arrête. Pour le problème le plus simple de type *un-à-un*, le nœud destinataire est le seul élément de ce sous-ensemble.

II.2.5.2 Algorithme de Bellman-Ford

À la différence de l'algorithme à fixation d'étiquettes, une autre approche nommée algorithme Bellman-Ford utilisée pour résoudre le problème du plus court chemin applique la correction d'étiquettes ("label correcting") [Ford, 1956] [Bellman, 1958] [Moore, 1959]. Les nœuds peuvent être exploités plusieurs fois pour mettre à jour la distance sans utiliser la file de priorités. Contrairement à l'algorithme de Dijkstra, à chaque itération, tous les arcs sont examinés même ceux qui sont adjacents au nœud le plus prometteur. L'appellation "label-correcting"provient du fait que le plus court chemin jusqu'à un nœud peut être corrigé à chaque itération. En outre, les arcs avec poids négatifs sont autorisés dans les graphes. Cet algorithme assure un temps d'exécution $O(|V||A|)$ dans le cas défavorable, mais en réalité cet algorithme possède une performance meilleure.

II.2.5.3 Algorithme objectif-dérivé

Comme une extension de l'algorithme de Dijkstra, l'algorithme A* a été proposé pour la première fois par Hart [Hart et al., 1968]. Au cours de l'exécution, une heuristique est mise en place pour estimer le coût à la destination pour chaque nœud. L'algorithme est capable de réduire l'espace de recherche si la recherche est faite dans la direction de la destination. La performance d'un A* dépend largement de la qualité de l'heuristique utilisée. Une simple et fréquente heuristique repose sur l'estimation du coût à la destination en utilisant la distance en ligne droite géographique jusqu'à la destination divisée par la vitesse maximale sur l'ensemble du réseau. [Dechter and Pearl, 1985] ont beaucoup étudié l'algorithme de la recherche *best-first* (littéralement : le meilleur en premier). Leur application dans le domaine de transport a été améliorée par d'autres travaux de recherche comme [Jagadeesh et al., 2002] [Goldberg and Harrelson, 2005] [Hahne et al., 2008].

II.2.5.4 Algorithme post-Dijkstra

La méthode la plus courante pour résoudre le problème du plus court chemin dans un réseau de transport repose sur la modélisation de ce dernier par un graphe condensé, un graphe dépendant du temps ou un graphe temps-étendu, ensuite l'application l'algorithme de Dijkstra ou ses versions variantes est utilisée.

Dans [Delling et al., 2013], une nouvelle approche vise à trouver tous les itinéraires optimaux au sens de Pareto (cf. section II.3.2.4) dans un réseau de transport en commun avec deux critères : le temps d'arrivée et le nombre de transferts. RAPTOR, l'algorithme proposé (appelé : round-based public transit router) joue seulement sur les éléments essentiels du réseau de transport : les arrêts, les lignes et les itinéraires en employant une simple structure de données.

II.2.5.5 Synthèse

Les solutions classiques pour le problème de type *un-à-plusieurs* repose sur Algorithme de Dijkstra [Dijkstra, 1959]. Au cours de l'exécution, une file de priorités est maintenue par rapport à la distance à la source. Le temps d'exécution de cet algorithme dépend de cette file de priorités. Avec l'implémentation d'un tas binaire, le temps d'exécution est $O((|V| + |A|) \log |V|)$ [Williams, 1964], qui peut être amélioré à $O(|A| + |V| \log |V|)$ avec l'implémentation du tas Fibonacci [Fredman and Tarjan, 1987].

En pratique, l'espace de recherche peut être réduit en employant la recherche bidirectionnelle qui fait exécuter une recherche en avant et une autre en arrière simultanément [Dantzig, 1998]. Dans [Pohl, 1969], on a montré avec les expérimentations que l'espace de recherche peut être réduit d'un facteur de 2. L'algorithme s'achève lors de la rencontre des espaces de recherche en *chaînage avant* (*avec l'étiquetage vers l'avant*) et en *chaînage arrière* (*avec l'étiquetage vers l'arrière*) qui contiennent un nœud sur l'itinéraire le plus court de l'origine vers la destination.

Les réseaux de transport en commun se basent souvent sur un tableau de marche. Les véhicules qui circulent sur ces réseaux opérés selon leurs horaires, sont donc basés sur un programme prédéfini. Ce dernier est constitué d'éléments comme les arrêts, les routes (les lignes de bus, métro, train etc.) ainsi que les tours qui correspondent aux itinéraires qu'utilisent les véhicules. Les tours peuvent donc être divisés en une chaîne de connections élémentaires sans arrêts intermédiaires avec une paire de départ/arrivée ainsi que le temps.

Plus 50 ans après sont apparition, les chercheurs n'ont pas trouvé un autre algorithme meilleure que celui de Dijkstra. Supposons qu'un "futur" algorithme meilleure sera possible, dans ce cas, chaque nœud et arc du graphe est visité une seule fois (et au moins une fois) ce qui donnera un temps linéaire $O(m + n)$. Même avec cette hypothèse très "audacieuse", ce dernier reste trop grand pour certains réseaux de transport. A défaut de ce "nouveau futur " algorithme, les chercheurs sont orientés vers des techniques d'assistance pour accélérer l'exécution de l'algorithme Dijkstra

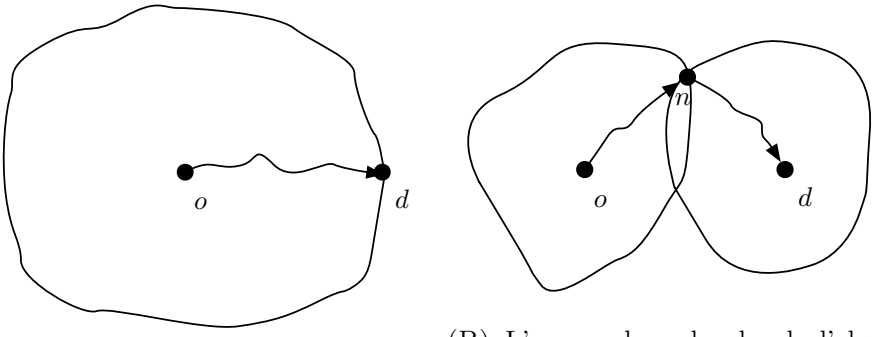
II.2.6 Techniques d'assistance à la planification d'itinéraire

Comme nous l'avons mentionné précédemment, il est possible d'apporter une aide à l'algorithme de Dijkstra afin d'améliorer ses performances en terme d'accélération de sa convergence, pour cela l'utilisation d'heuristique est nécessaire. La section qui suit donnera un descriptif et fournira des explications sur son fonctionnement pour différents problèmes

II.2.6.1 Recherche bidirectionnelle

L'idée d'appliquer une recherche bidirectionnelle sur l'algorithme Dijkstra est assez simple. Elle revient à lancer les recherches à partir des points d'origine et de destination simultanément. La recherche s'achève quand les deux *frontières de recherche* se rencontrent. En même temps, deux files de priorités sont maintenues pour le chaînage avant et le chaînage arrière. La file de priorités pour le chaînage avant est maintenue lors

d'une recherche ordinaire en sens "forward". Pour le chaînage arrière, il est nécessaire d'utiliser une autre file indépendante pour la recherche en avant sur le graphe inversé. Donc pour l'étape de détente (cf. section II.2.5.1), l'arc (u, v) est devenu (v, u). Pour chaque itération, le nœud est marqué avec la longueur temporaire *globale*. Ce terme *globale* signifie la longueur minimale après comparaison des deux files de priorité de chaînage avant et arrière. Une fois un nœud n est marqué dans les deux files de priorité, le plus court chemin de l'origine à la destination via le nœud n est obtenu. Mais ce dernier n'est pas certainement le plus court chemin entre les deux extrémités. Afin de garantir l'optimalité globale, la recherche doit continuer jusqu'à ce que la somme des minima des deux files soit plus grand que la longueur du plus court chemin obtenu.



(A) L'espace de recherche de l'algorithme de Dijkstra

(B) L'espace de recherche de l'algorithme de Dijkstra bidirectionnel

FIGURE II.5: Schéma de l'espace de recherche

La Figure II.5, montre bien que l'espace de recherche avec Dijkstra consiste en un grand "cercle" dont le "rayon" est égal à la distance de l'origine à la destination. Alors que dans le cas de la recherche bidirectionnelle, on a deux "cercles" plus petits avec le "rayon" qui représente la moitié de la distance de l'origine à la destination.

II.2.6.2 Recherche avec graphe hiérarchisé

Cette approche exige de mettre le graphe en plusieurs niveaux et ajouter les arcs supplémentaires qui sont distribués sur les différents niveaux. Seulement une petite part des arcs sont engagés pour trouver le plus court chemin de la requête. Les travaux de recherche utilisant cette technique montrent son efficacité sur les réseaux de routes et de transport en commun, ainsi que les requêtes avec les grilles horaires de lignes [Jung and Pramanik, 2002] [Schulz et al., 2002]. Dans le travail de [Bauer et al., 2013], un cadre théorique est développé pour étudier la taille de l'espace de recherche avec la recherche hiérarchisée. Les estimations sur la taille de l'espace de recherche peuvent être formulées en fonction de la taille du graphe initial.

Pour le réseau routier, l'approche hiérarchisée a pour but d'exploiter sa hiérarchie inhérente. Le réseau peut être analysé à l'aide de sa hiérarchie qui catégorise les tronçons de réseau auprès d'un certain nombre de niveaux (par exemple trois catégories : routes principales, routes secondaires et routes locales). À titre d'exemple, sur la carte de Paris et ses environs qui sont visualisés sur la Figure II.6, nous constatons qu'il existe au moins trois niveaux de route : voies blanches, voies jaunes et voies oranges. Quand la distance entre l'origine et la destination est assez importante, l'algorithme de recherche du plus court chemin s'exécute sur un réseau principal des routes importantes, au lieu de celui comprenant tous les arcs et transits. À partir des extrémités, cette heuristique définie le graphe correspondant. Même s'il n'y a pas de garantie sur l'optimalité, l'approche hiérarchisée est largement appliquée [Hliněnỳ and Moriš, 2011] [Efentakis et al., 2011]. L'algorithme *Contraction Hierarchies* (CH), proposé dans [Geisberger et al., 2008] [Geisberger et al., 2012], vise à trouver le plus court chemin d'une manière exacte. L'idée de cette approche est d'augmenter le graphe G en utilisant les raccourcis qui servent aux requêtes de longues distances pour s'échapper des nœuds les moins importants. Pour le faire, nous faisons répéter l'opération de *nœud contraction* ; afin de contracter un nœud n, ce dernier est supprimé temporairement sur G, en même temps, un raccourci est ajouté entre chaque paire de nœuds adjacents u et v si le plus court chemin de u à v est unique et passe par n. En plus, cet algorithme met les nœuds en ordre en suivant leurs *niveaux d'importance* dans l'étape préliminaire et les contracte du niveau le plus bas vers le plus haut dans la hiérarchie.

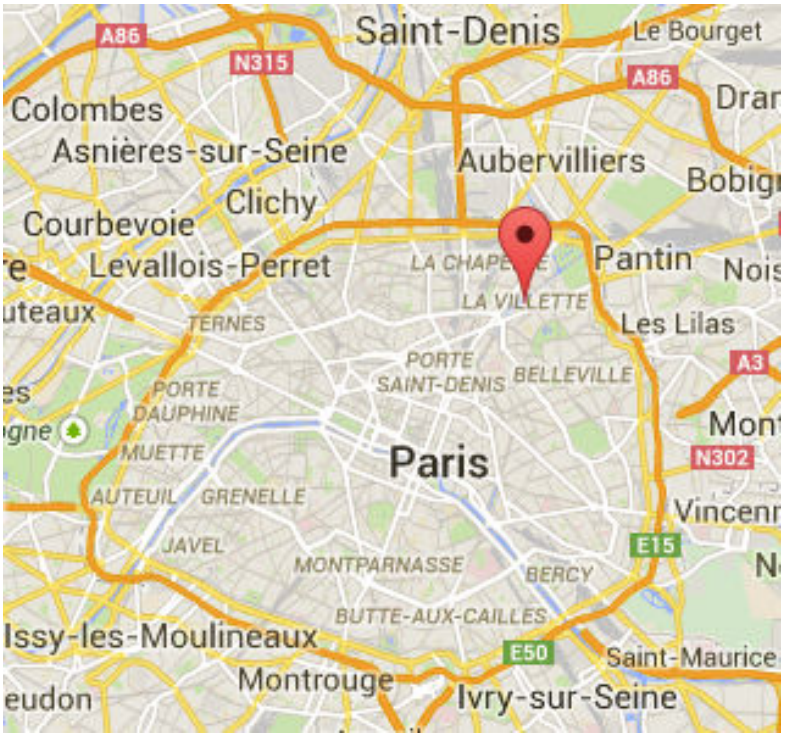
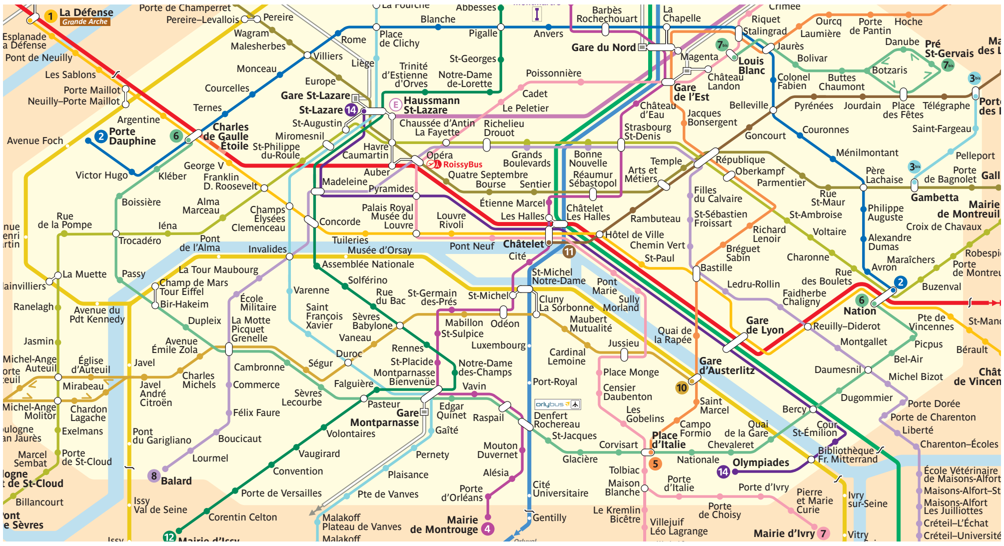
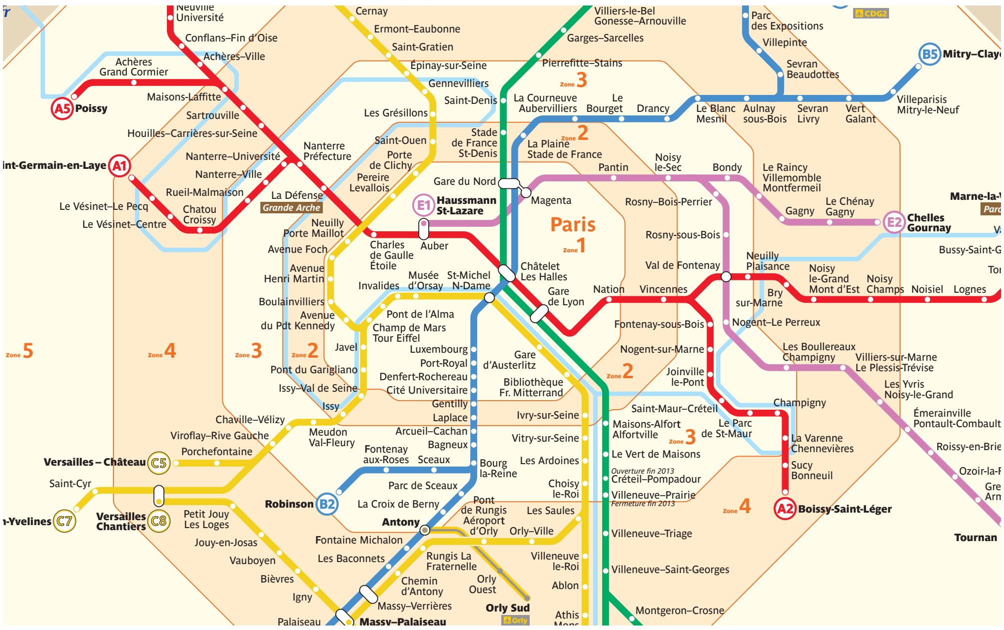


FIGURE II.6: Le réseau routier partiel de Paris et ses environs

Pour les transports en commun, cette technique d'assistance n'est pas aussi appréciée que pour les réseaux routiers [Bauer et al., 2011]. L'explication de ce manque d'efficacité réside dans le défaut de hiérarchie inhérente sur un réseau de transport en commun. Par exemple sur le réseau de transport parisien, le réseau de RER qui couvre l'Île de France est superposé sur celui du Métro de Paris(configuré par la Figure II.7). Autrement dit, Métro et RER possèdent la même classe hiérarchique à Paris ; aucun d'entre eux profite d'un privilège pour le problème de la planification qui dépend plutôt de l'horaire des lignes correspondantes. En revanche, si nous considérons un itinéraire de Paris à une autre ville d'Île de France, la hiérarchie du réseau fait apparaître au moins deux niveaux : inter-ville et intra-ville.



(A) métro & RER



(B) RER

FIGURE II.7: Une partie de réseau métro &RER de Paris

II.2.6.3 Autres techniques

Un cas extrême pour le précalcul est de calculer toutes les distances entre chaque paire de nœuds dans le graphe. La réponse à la requête possède un temps de réponse de l'ordre de $O(1)$ via une opération de *lookup*, mais la complexité de précalcul est de $O(n^2)$ pour le temps et pour l'espace. Cette pratique a été appliquée pour le problème de planification d'itinéraire sur le réseau routier. Compte tenu de la complexité de précalcul, un sous-ensemble de nœuds, (par exemple les nœuds importants pour le transit sur un réseau routier) est pris en compte pour un précalcul plus raisonnable [Bast et al., 2009] [Bast et al., 2007].

II.2.6.4 Combinaison des techniques

Les techniques décrites ci-dessus utilisent les différentes propriétés du graphe. Pour obtenir une accélération supplémentaire, il est possible de profiter des avantages des uns et des autres en combinant ces différentes techniques.

La littérature sur la combinaison des techniques d'accélération est très riche. L'alliance entre l'approche hiérarchique et la recherche objective-dérivée est étudiée dans [Jagadeesh et al., 2002] [Bauer et al., 2010]. Les travaux [Holzer et al., 2004] [Holzer et al., 2009] concluent systématiquement sur toutes les combinaisons entre les techniques d'accélération.

Dans cette thèse, nous nous sommes intéressés à la combinaison du graphe hiérarchisé et à la recherche bidirectionnelle en transport comodal. Avec le pré-traitement du réseau, le graphe est transformé en une hiérarchie. En appliquant l'algorithme de planification d'itinéraire sur ce graphe en niveau, nous obtenons des itinéraires candidats avec les spécifications des fenêtres de temps. En suite, une recherche bidirectionnelle est appliquée sur le graphe temps-étendu en prenant en compte des fenêtres de temps.

Nous nous concentrons dans la section suivante sur l'optimisation et nous présontons les principes d'optimalité.

II.3 Optimisation

II.3.1 Principe d'optimalité de la programmation dynamique

La résolution du problème avec la programmation dynamique se fonde sur *le principe d'optimalité de Bellman* [Bellman, 1957].

Ce principe fournie une stratégie de résolution ; d'une façon ascendante, nous commençons par résoudre des sous-problèmes les plus essentiels pour par la suite détruire étape par étape les solutions de l'ensemble en combinant des solutions optimales des sous-problèmes précédents.

La recherche du plus court chemin dans un graphe avec Dijkstra se base sur ce principe d'optimalité. Nous allons voir aussi une autre application de cette règle dans la section III.5.2.1 où les solutions optimales sont trouvées avec une méthode exacte. La recherche heuristique est également utilisable, elle montre souvent des avantages par rapport à la programmation dynamique pour la résolution de certains problèmes.

II.3.2 Optimisation multicritère

II.3.2.1 Problème d'optimisation multicritère

Un problème d'optimisation multicritère (aussi appelée *multiobjectif*) consiste à optimiser simultanément plusieurs fonctions coût souvent contradictoires. Sans perte de généralité, il peut s'écrire comme suit :

$$\text{minimiser} \quad F(x) = (F_1(x), F_2(x), \ldots, F_n(x)), \quad x \in \Omega \tag{II.1}$$

où

- x est une solution possible pour le problème considéré ;
- Ω est l'espace des solutions faisables.

Généralement les fonctions de coût $F_1(\cdot), F_2(\cdot), \ldots, F_n(\cdot)$ ne possèdent pas un minimum *commun* surtout quand les fonctions objectifs sont antagonistes. Pour un problème combinatoire, nous devons optimiser un ensemble de fonctions objectifs simultanément.

Pour ce faire, il existe principalement deux catégories de méthodes : les approches Pareto-optimales et les méthodes de pondération.

- **Optimalité au sens de Pareto**
  La famille de solutions d'un problème multicritère représente tous les éléments dans l'espace de recherche ne pouvant être amélioré simultanément. C'est le concept de l'optimalité au sens de Pareto. Toutes les solutions de cette famille sont des solutions optimales au sens de Pareto.
  Nous reprenons l'Équation II.1 pour une définition formelle :
  La relation de dominance entre deux solution x et y est notée comme $F(y) \prec F(x)$, si y est dominée par x. Nous avons :
  (a) $\forall i \in \{1, \ldots, n\}, F_i(x) \leq F_i(y)$;

(b) $\exists j \in \{1, \ldots, n\}, F_j(x) < F_j(y)$.

Une solution x est dite optimale au sens de Pareto *si et seulement si* x n'est dominée par aucune autre solution. Cette approche aboutie souvent à un ensemble de solutions au lieu d'une seule, ce qui donne une flexibilité pour prendre la décision multicritère.

Prenons un exemple (cf. la Figure II.8) pour illustrer cette approche. Les deux fonctions objectifs sont à minimiser. Comme indique dans cette figure, l'ensemble des solutions optimales au sens de Pareto fait aussi la frontière de Pareto. La

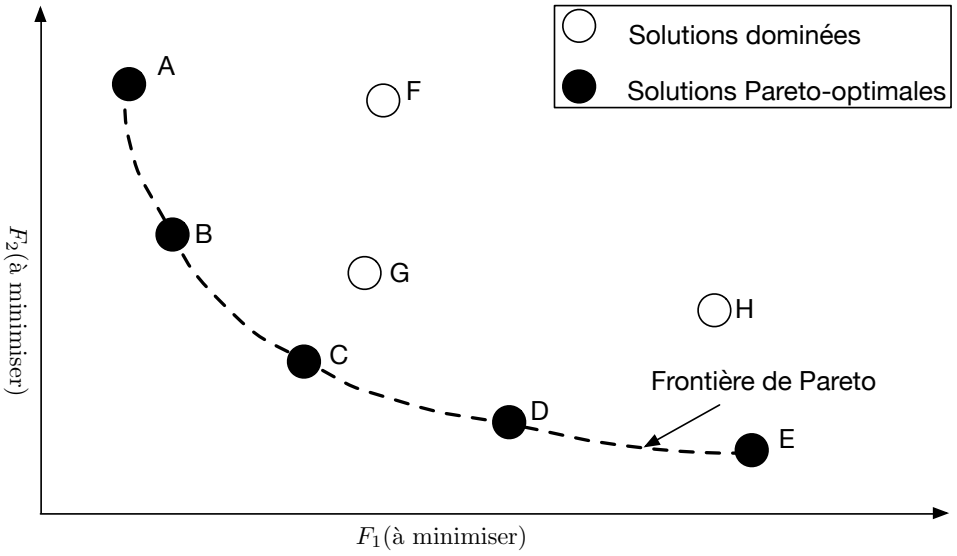


FIGURE II.8: Approche de l'optimalité au sens de Pareto avec deux objectifs à minimiser. Les points A, B, C, D et E sont les solutions optimales au sens de Pareto

Figure II.8 montre la frontière de Pareto formé par les points A, B, C, D, E. Le point F est dominé par B, G est dominé par C et H est dominé par C et D.

— **Les approches non Pareto**

Une approche non basée sur la dominance Pareto est fondée sur l'agrégation des différents critères dans un objectif simple. Cette technique cherche à ramener le problème multicritère à un ou plusieurs problèmes mono-critères. Cette méthode combine les différentes fonctions coût F_i en une seule fonction objectif F généralement linéaire, comme représenté par l'Équation II.2

$$F(x) = \sum_{i=1}^{n} \omega_i F_i(x) \tag{II.2}$$

où

— $F(\cdot)$ est la somme d'agrégation des fonctions objectifs F_i avec différents poids ω_i ;

— $\forall i \in \{1, \ldots, n\}, \omega_i \in [0, 1]$;

— $\sum_{i=1}^{n} \omega_i = 1$.
Le vecteur $W = (\omega_1, \ldots, \omega_n)$ est nommé vecteur de poids qui influe directement sur les résultats du problème.

Afin de gérer les fonctions coût contradictoires, un décideur doit d'intervenir à un certain moment. Il existe trois catégories de méthodes de résolution du problème :

— *a priori* : les préférences sont fixées avant la résolution ;
— *interactive* : au cours de la résolution, le décideur prend la décision d'intervenir ;
— *a posteriori* : après avoir obtenu l'ensemble de solutions potentielles, le décideur en choisit une.

II.3.2.2 Approches *a priori*

Ces approches consistent à construire une seule fonction objectif qui inclut l'ensemble des objectifs pour rendre le problème d'optimisation à un seul objectif. Ceci permettrait d'appliquer des algorithmes bien connus.

Quelques méthodes qui peuvent servir comme une approche *a priori* seront présentées.

— **Ordre lexicographique** : Les objectifs sont classés par ordre de préférence. Plusieurs optimisations seront mises en place en suivant le niveau d'importance du critère. Comme cette approche se concentre trop sur des solutions extrêmes sans aucun compromis, elle très rarement appliquée.
Mais, même avec cet inconvénient, elle est appliquée dans le problème du plus court chemin. Par exemple, dans le cas où le critère est de faire le moins de changement, cette approche implique la solution faisable même si la distance de parcours est trop importante.
— **Agrégation des objectifs** : Cette approche linéarise des objectifs en faisant la somme pondérée de chacun.
— **Intégrale de Choquet** : L'intégrale de Choquet est une méthode pour modéliser les préférences du décideur plus finement que par la simple agrégation. Au lieu d'associer une note à chaque solution, cette méthode permet de définir des préférences entre les solutions. Cela rend cette méthode plus utile surtout pour la modélisation des préférences hétérogènes. Cette technique a été appliquée pour le problème du plus court chemin dans [Galand et al., 2010] [Fouchal et al., 2011].

II.3.2.3 Approches *interactives*

Les approches interactives consistent à proposer une solution qui laisse une liberté au décideur de choisir des critères sur lesquels il souhaiterait améliorer la solution. Ensuite

une nouvelle solution sera fournie. Le processus est répété jusqu'à l'obtention d'une solution satisfaisante. Les approches interactives sont capable d'exploiter les solutions obtenues dans les calculs précédents en vue d'améliorer les performances. Ce type d'approche se destine souvent aux problèmes dont le temps de calcul est important.

II.3.2.4 Approches *a posteriori*

Cette approche a pour objectif d'obtenir la frontière de Pareto la plus complète possible. Elle vise à fournir l'ensemble de solutions au décideur qui en sélectionnera une, qu'il considère comme la plus adaptée. L'avantage de cette méthode est de garder toute solution qui peut être intéressante pour le décideur. En même temps, l'inconvénient est la difficulté de représenter l'ensemble de la frontière de Pareto dans le cas de plus de deux objectifs.

— **Métaheuristiques** : Une métaheuristique est un algorithme d'optimisation visant à résoudre des problèmes d'optimisation difficiles, tentant d'apprendre les caractéristiques du problème pour en trouver la meilleure solution ou son approximation. Pour son adaptation au problème d'optimisation multicritère, les algorithmes génétiques sont souvent employés.
— **Approches exactes** : Les approches exactes sont très variées et souvent proposées selon le type de problème. Lors de l'élaboration d'une approche exacte, il faut toutefois éviter l'explosion combinatoire de l'espace de recherche.

II.3.3 Algorithme génétique

Une brève présentation des algorithmes génétiques (AGs) sert à la compréhension globale et fait introduire l'application dans cette problématique particulière. Les ouvrages ou les articles exposants l'algorithme évolutionniste et génétique sont très nombreux [Holland, 1975] [Goldberg, 1989] [Davis et al., 1991] [Fonseca et al., 1993] [Bäck and Schwefel, 1993] [Bäck, 1996] [Gen and Cheng, 2000].

Les algorithmes génétiques appartiennent à la famille des algorithmes évolutionnistes. Grâce à une analogie avec la théorie de l'évolution, ils sont basés sur le processus d'évolution génétique, où une procédure de séparation et d'évolution d'une population de solutions potentielles s'applique à travers des générations. Au lieu de chercher directement les valeurs, ce sont des formules qui forment l'espace de recherche.

II.3.3.1 Une brève histoire

Les principes fondamentaux des AGs ont été initiés par John H. Holland [Holland, 1975]. Au cours de ses développements, le travail de [Goldberg, 1989] applique cette méthode pour la première fois pour la résolution des problèmes d'optimisation et il populariser les algorithmes génétiques. De nombreux travaux de recherche se réalisent et font avancer ces méthodes d'optimisation, nous citons à titre d'exemple : [Michalewicz et al., 1992] [Mahfoud and Goldberg, 1995] [Deb, 1999].

II.3.3.2 Les concepts essentiels

Étant basés sur des phénomènes biologiques, des termes de génétique sont empruntés et employés dans le contexte de l'algorithme génétique. Dans les cellules de l'organisme, les chaînes d'ADN comportent des codages des fonctionnalités de l'organisme appelés *chromosomes* dont l'élément de base est un *gène*. Ce dernier peut être positionné par son *locus* qui signifie sa position sur le chromosome. Un seul *gène* possède les différentes versions, appelées *allèles*. Les algorithmes génétiques traitent une *population* qui rassemble un nombre d'*individus*. L'ensemble des *gènes* d'un individu fait son *génotype*. Dans les AGs, une analogie avec *la théorie de l'évolution* propose que les *gènes* conservés ceux qui sont les plus adaptés aux besoins.

II.3.3.3 Architecture générale d'un algorithme génétique

Les algorithmes génétiques constituent une classe de stratégies de recherche afin d'aboutir à un équilibre entre l'exploration et l'exploitation. Ils représentent des procédures faisant appel à un choix aléatoire comme outil pour guider l'exploration dans l'espace des paramètres codés [Mesghouni, 1999]. Quelque soit la problématique considérée, l'architecture générale représentant un algorithme génétique peut être illustrée par la Figure II.9 [Goldberg, 1989] [Deb et al., 2002].

En considérant la structure générale de l'algorithme génétique, nous présentons les éléments essentiels comme suit :

- **Codage** : Une formule apte représente la solution ;
- **Population initiale** : La génération initiale des individus qui est générée aléatoirement mais peut influencer la rapidité de la convergence vers l'optimum au cours de l'exécution des AGs ;
- **Fonction d'évaluation ou fonction de fitness** : Une fonction à optimiser qui sert à l'évaluation des individus de la population. D'après cette fonction, des notes sont attribuées aux solutions qui correspondent à leurs adaptations au problème ;

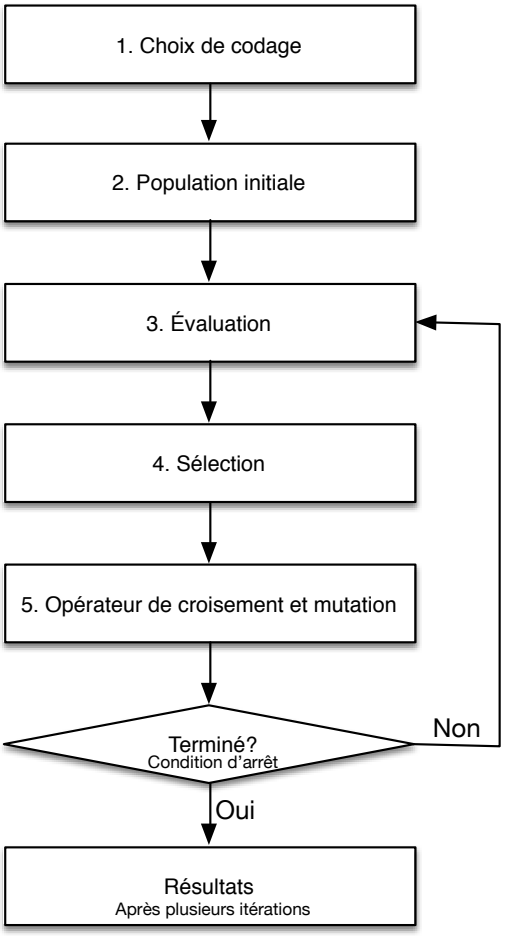


FIGURE II.9: Architecture d'un algorithme génétique

— **Sélection** : Un mécanisme pour déterminer quels individus candidats sont plus aptes à obtenir à des résultats plus performants. Cette procédure est basée sur la note attribuée. Plusieurs techniques existent ;

— **Opérateur de croisement** : Une procédure de manipulation des chromosomes parents en échangeant des parties de leurs chaînes pour aboutir à de nouveaux chromosomes dans la nouvelle génération. Le croisement peut être simple (à un point) ou multiple (à multi-points) ;

— **Opérateur de mutation et correction** : Une procédure de manipulation de gène au sein d'un chromosome qui peut être substitué à un autre d'une façon aléatoire. Afin d'éviter une convergence prématurée de l'algorithme, cet opérateur sert à garantir l'exploitation de l'espace ;

— **Paramétrage** : la mise en place des paramètres comme la taille de la population, les probabilités concernant les opérateurs précédents.

II.3.3.4 Codage des chromosomes

Le codage est un processus de modéliser de la solution avec un chromosome. Cette modélisation peut employer différents types d'alphabets : des bits, des nombres, des caractères, des listes etc. Le choix de ce dernier dépend principalement du problème à résoudre. Le codage le plus fréquent et classique est celui de l'alphabet binaire : {0,1}. Avec ce type de codage, chaque chromosome représente une séquence binaire où chaque unité peut illustrer des caractéristiques de la solution.

II.3.3.5 Population initiale

La rapidité de la convergence vers la solution optimale dépend en partie de la phase d'initialisation. En réalité, les connaissances à priori de solutions candidates de "bonne qualité" comme le point de départ permet d'accélérer la convergence de cet algorithme. En revanche, si nous ne procédons aucune information sur la position de l'optimum parmi l'ensemble des solutions faisables alors les individus de la population initiale seront générés de façon aléatoire. Les sélections seront aussi réalisées d'une façon uniforme dans chacun des domaines associés aux composantes de l'ensemble des solutions en respectant les contraintes. Les opérateurs de croisement et de mutation sont introduits pour parcourir l'ensemble des solutions le plus largement possible et donc de s'assurer de la diversité d'une population au cours des générations.

II.3.3.6 Fonction d'évaluation

Elle consiste à évaluer les chromosomes dans la population courante et à leur attribuer des valeurs d'adaptation qui permettent de choisir les individus pour se reproduire et participer à la génération suivante. Le choix des individus est très important car la performance de la nouvelle génération est fortement concernée. Pour garantir une amélioration maximale de la qualité des solutions au cours des générations, il est important de bien sélectionner cette fonction d'évaluation. La fonction d'évaluation prend en compte l'ensemble des critères à optimiser.

II.3.3.7 Sélection

Après avoir fixé les critères d'évaluation, selon lesquels chaque individu peut être évalué, une stratégie doit être choisie pour créer la génération prochaine. La sélection est un processus pour choisir deux individus parmi la population comme parents et effectuer les opérations suivantes. La seule mesure à notre disposition est la valeur de la fonction

d'évaluation. Plus le score d'un individu selon la fonction d'évaluation est important, plus il est probable d'être choisi. Plusieurs principes de sélection sont présentés dans la littérature et nous en allons introduire quatre dans la suite de ce paragraphe.

— **Sélection par roulette** : Cette technique simule le fonctionnement d'une roulette de jeux pour choisir les chromosomes. La portion sur la roulette à laquelle chaque individu correspond est proportionnelle à sa performance d'évaluation. Un tirage au sort homogène est ensuite effectué sur cette roulette.
— **Sélection par tournoi** : Cette technique consiste à choisir aléatoirement des pairs d'individus et à permettre au plus fort parmi ces pairs d'être sélectionné.
— **Élitisme** : Ce type de sélection consiste à garder (reconduire) un ou plusieurs des meilleurs individus dans la nouvelle génération. Ensuite, le reste de la population est généré selon l'algorithme de sélection normale.

II.3.3.8 Opérateur de croisement

L'opérateur de croisement permet aux deux chromosomes parents d'échanger partiellement leur patrimoine génétique, donnant naissance à deux nouveaux individus. Après cette opération, les gènes sont redistribuées. Nous tentons de relier des "bons" gènes venant des parents à l'aide de cet opérateur de croisement. Cependant, aucun nouveau gène n'est produit dans la population durant cette opération. Avec une probabilité de croisement P_c, les individus regroupés par paires effectuent l'échange de leur gènes en partie. Cette probabilité est généralement proche de 1. Il existe plusieurs modes de croisement, parmi lesquels les croisements à un point et à deux-points sont les plus importants.

Parent 1	100001	111111
Parent 2	100010	100001

(A) Chromosomes parents

Enfant 1	100001	100001
Enfant 2	100010	111111

(B) Chromosomes enfants

FIGURE II.10: Opérateur de croisement à un point

Parent 1	10000	111	1111
Parent 2	10001	010	0001

(A) Chromosomes parents

Enfant 1	10000	010	1111
Enfant 2	10001	111	0001

(B) Chromosomes enfants

FIGURE II.11: Opérateur de croisement à deux points

II.3.3.9 Opérateur de mutation

Avec une probabilité faible P_m, une modification de gènes des individus enfants est réalisée. Elle permet d'introduire des nouveaux gènes et diversifier les individus, sans

forcément faire évoluer toute la génération. Cette mesure permet aussi d'explorer l'espace de recherche aléatoirement et d'éviter la convergence prématurée vers un optimum local.

Chromosome 1	1	10010
Chromosome 1 après mutation	0	10010

FIGURE II.12: Opérateur de mutation

II.3.3.10 Paramétrage

L'application des algorithmes génétiques ne nécessite pas une connaissance de la particularité du problème concerné. Néanmoins, l'efficacité dépend fortement du réglage des différents paramètres. Ces derniers comprennent l'effectif de la population, le nombre de générations autorisé, la probabilité de mutation P_m et la probabilité de croisement P_c.

Les travaux de recherche focalisent aussi sur le réglage des paramètres et les interactions entre eux [Goldberg et al., 1991] [Harik et al., 1999] [Goldberg et al., 1992] [Thierens and Goldberg, 1993] [Chakraborty et al., 1996]. Pour les deux premiers paramètres, ils dépendent de la complexité du problème traité. Il faut éviter une population de trop faible effectif qui provoque une convergence prématurée, également éviter une grande population qui donnerai un temps de calcul excessif. De même, le nombre de génération est choisi pour allier qualité de solution et temps de calcul.

Quand il s'agit des deux autres paramètres, une règle générale souvent appliquée consiste à attribuer une probabilité proche de 1 pour l'opérateur de croisement et une petite probabilité pour l'opérateur de mutation (par exemple : $P_m = 0,05$ et $P_c = 0,95$).

II.3.3.11 Algorithme génétique pour les problèmes de transport

Les algorithmes génétiques ont été appliqués dans divers domaines avec succès, notamment dans celui qui nous intéresse dans le cadre de cette thèse à savoir le transport. Nous citons dans ce paragraphe quelques travaux de la littérature.

Dans [Ahn and Ramakrishna, 2002], un algorithme génétique avec un codage spécifique et des chromosomes à longueur variable est proposé pour traiter le problème du plus court chemin. Les travaux ont mis l'accent sur le paramétrage de l'algorithme génétique surtout la taille de la population des chromosomes afin d'obtenir la qualité des solution et la vitesse de convergence désirée. Dans [Yu and Lu, 2012], on propose un algorithme génétique pour résoudre le problème du plus court chemin multimodal. Les chromosomes

à longueur variable constituent de plusieurs parties, chacune représentant un mode de transport. En même temps, les nouveaux opérateurs sont définis pour le croisement et la mutation pour les opérations "inter-mode". Dans [Herbawi and Weber, 2012] [Herbawi, 2012], on propose un algorithme génétique combiné d'une heuristique d'insertion pour le problème de covoiturage dynamique dans lequel les voyageurs sont affectés aux chauffeurs afin d'organiser le ramassage et le dépôt des voyageurs. Dans un contexte d'optimisation multicritère, la fonction coût prend en compte la minimisation de la distance parcourue et le temps de conduite (ou temps des chauffeurs) et le temps de parcours des voyageurs, ainsi que la maximisation du nombre d'affectations. L'algorithme génétique est appliqué dans un premier temps, par la suite, l'heuristique d'insertion est appliquée sur le résultat obtenu lors de la première étape pour traiter les demandes d'affectation dynamiques reçues en temps réel.

II.3.4 La recherche tabou

La métaheuristique recherche tabou est présentée pour la première fois en 1986 dans [Glover, 1986] puis développée toujours par F. Glover dans [Glover, 1989] [Glover, 1990]

Le principe de cette méthode d'optimisation est assez simple. Au sens large, il s'agit d'une approche itérative de recherche locale. En fait, cette métaheuristique profite de l'idée d'explorer le voisinage d'une position donnée et ensuite de converger vers la position qui minimise (ou maximise) la fonction objectif. Il est possible d'accepter temporairement des solutions qui dégradent le résultat, c'est le cas, quand tous les voisins ont une valeur plus élevée dans le cas d'un problème de minimisation. C'est cette technique qui permet à l'algorithme de sortir d'un optimum local. L'assurance de ne plus revenir vers cet optimum local consiste à doter cette méthode d'une mémoire : c'est la liste *tabou*, les dernières positions déjà explorées sont ajoutées et conservées dans cette liste. La taille de cette dernière est un paramètre de l'approche à régler et joue parfois un rôle très important dans certains types de problèmes [Tsubakitani and Evans, 1998] [Salhi, 2002]. Comme cette liste doit prendre en compte toutes les positions visitées, elle risque de consommer beaucoup de mémoire. Une technique pour éviter cet inconvénient est de ne mémoriser que les mouvementes précédents correspondants aux valeurs de fonction.

L'algorithme tabou pour un problème de minimisation est décrit par le pseudo code Algorithme *TABOU*.

Algorithm 2 Algorithme de la recherche tabou (**TABOU**)

1: $s \leftarrow s_0$ ▷ Initialiser le point de départ
2: $s_m \leftarrow s$ ▷ La meilleure solution courante
3: $N(s) \leftarrow$ voisinage de s
4: $T \leftarrow \emptyset$ ▷ Initialiser la liste tabou
5: **Tant que** (!Conditions d'arrêt)
6: **Pour tout** $s_c \in N(s_m)/T$
7: $s_v \leftarrow$ le meilleur s_c
8: **Fin pour**
9: **Si** $fitness(s_v) < fitness(s_m)$
10: $s_m \leftarrow s_v$
11: $T \leftarrow T \cup s_m$ ▷ Régler la liste tabou
12: **Fin si**
13: **Fin Tant que**

En fonction de la façon de régler la liste et de définir le voisinage, plusieurs versions existent.

Pour la gestion de la liste, nous avons une classification selon la durée de mémorisation [Glover, 1989] :

— Liste à court terme : les solutions récemment considérées sont maintenues dans la liste. C'est-à-dire, les solutions potentielles de la liste ne peuvent pas être ré-explorées si un certain point d'expiration n'est atteint.
— Liste à terme intermédiaire : c'est une intensification de la liste à court terme pour mettre plus de poids sur les zones qui sont identifiées plus prometteuses. Cette pratique favorise l'exploration approfondie des espaces de recherche considérés particulièrement intéressants et diriger la recherche vers ces régions.
— Liste à long terme : contrairement à la dernière méthode, c'est une stratégie de diversification qui réoriente périodiquement la recherche vers une zone rarement exploitée pour éviter la convergence vers une solution sous-optimale.

Une définition de voisinage de la recherche tabou est celle du voisinage variable : changer systématiquement le voisinage au fil de la recherche. Cette méthode permet d'explorer les voisinages éloignés de la solution courante et aller vers une nouvelle solution aléatoirement. Avec cette définition du voisinage, les caractéristiques favorables d'une solution sont souvent gardées et mis à profit pour obtenir les solutions voisines prometteuses [Hansen and Mladenović, 2001] [Paraskevopoulos et al., 2008].

La métaheuristique avec la recherche tabou a montré son efficacité et elle est capable de fournir des solutions de très bonnes qualités pour certains problèmes d'optimisations.

Nous citons dans ce qui suit, des exemples d'utilisation dans le domaine de transport. Les travaux de recherche présentés dans [Shen and Kwan, 2002] concernent le problème d'arranger et le remplacement des chauffeurs de transport en commun avec fenêtres de temps. Une approche basée sur la recherche tabou est mise en place pour résoudre le problème qui ne peut qu'être résolu qu'avec compromis en utilisant les approches existantes. Les résultats obtenus sont comparables à ceux qui sont trouvés en utilisant les méthodes basées sur la programmation linéaire. Dans [Nuortio et al., 2006], le problème de tournées de véhicules pour le ramassage des déchets solides municipaux, les auteurs ont utilisés la recherche tabou et ont développé et mis en place une stratégie de voisinage de recherche varié et guidé pour s'adapter au problème réel. Dans [Pacheco et al., 2009], le problème à résoudre est constitué de deux niveaux ; concevoir les routes d'un réseau de bus puis affecter les bus aux routes pour optimiser le niveau de service. La fonction objectif prend en compte le temps d'attente et le temps de parcours pour les voyageurs au sein du réseau. La recherche tabou appliquée pour ce problème d'optimisation définit le voisinage de recherche comme routes obtenues avec plusieurs opérations telles que l'insertion ou l'élimination d'un arrêt dans une route.

II.3.5 Algorithme de colonies de fourmis

L'algorithme de colonies de fourmis est une métaheuristique qui est inspiré du comportement intelligent des fourmis pour la recherche du plus court chemin entre leur nid et une source de nourriture. Cet algorithme simule des fourmis artificiels pour trouver les solutions des problèmes d'optimisation combinatoire. Proposé par Colorni et al. dans les années 1990 [Colorni et al., 1991], l'algorithme est initialement appliqué pour la recherche du plus court chemin dans un graphe. Lorsqu'une fourmi individuel bouge entre sa colonie et une source de nourriture, elle laisse une piste de phéromones qui sont des substances chimiques et attractives pour les autres fourmis. Ces dernières ont tendance à suivre cette piste et la renforcent en déposant des phéromones. Les fourmis suivent souvent la piste la plus concentrée en phéromones qui sont cependant volatiles. La piste la plus courte pour atteindre la même source de nourriture est favorisée et parcourue par plus de fourmis. Ainsi, la piste la plus courte devient de plus en plus renforcée et attractive aux fourmis. D'autre part, les phéromones sur la longue piste vont disparaître par vaporisation. Toutefois, des fourmis ignorent occasionnellement la plus courte piste et prennent d'autres chemins pour arriver à la source de nourriture ce qui permet éventuellement d'explorer d'autres chemins et peut être de trouver un autre chemin plus court que le précédent. Plusieurs travaux de recherche sont dédiés

aux comportements des fourmis pour les métaheuristiques [Dorigo and Gambardella, 1997] [Dorigo et al., 1999].

Avec le constat que l'échange d'informations entre fourmis est réalisé via l'environnement sous nom du principe "stigmergie", cet algorithme fait partie de la famille des algorithmes stigmergiques, comme un grand nombre d'algorithmes qui partage le même principe. Les méthodes avec ce type d'algorithmes nous amènent parfois "l'optimisation stigmergique" [Crina and Ajith, 2006].

L'implémentation d'un algorithme de colonies de fourmis se fait en trois étapes : ConstruireSolutions, DeamonActions, et màjPhéromones(Algorithme *FOURMIS*). Un graphe est préalablement construit pour le problème cible avec des nœuds et des arcs. Dans la première étape, la colonie de fourmis visitent les états adjacents du problème considéré en avançant vers des nœuds voisins du graphe construit. En utilisant la piste de phéromones et des informations heuristiques, les fourmis appliquent des stratégies stochastiques pour prendre des décisions locales et construisent des solutions au problème. Ensuite, les fourmis évaluent les solutions pour décider de la quantité de phéromones à laisser dans la deuxième étape, ainsi, les pistes sont mises à jour : si une fourmi décide de laisse des instances chimiques, la piste sera renforcée ; dans le cas contraire, l'évaporation des phéromones entraînera la diminution de l'attractivité de la piste. La troisième étape met en place des actions centralisées et ne peut pas être effectuée par une fourmi individuel, par exemple, la collecte des informations globales pour décider du dépôt des phéromones afin de diversifier le processus de la recherche d'une vue globale. Cette dernière étape est optionnelle.

Algorithm 3 Algorithme de colonies de fourmis (**FOURMIS**)

```
1: Tant que ( !Conditions d'arrêt)
2:     ConstruireSolutions                          ▷ Construire des solutions
3:     màjPhéromones          ▷ màj des phéromones avec le coefficient d'évaporation
4:     DeamonActions              ▷ optionnel, mise en place des actions centralisées
5: Fin Tant que
```

Les algorithmes de colonies de fourmis ont été appliqués pour les problèmes d'optimisation combinatoire, surtout pour ceux de tournée de véhicules où chaque fourmi artificielle représente un véhicule et pouvant sélectionner des points à visiter pour construire des routes. De plus, un avantage en comparaison avec les autres métaheuristiques est l'adaptation flexible de la colonie de fourmis aux changements dynamiques du graphe étudié. Ceci montre un intérêt spécifique pour le routage réseau [Sim and Sun, 2003]. nous citons ici quelques exemples dans le domaine de transport.

Les travaux de recherche dans [Yuan and Wang, 2009] traitent le problème du plus court chemin dans les scénarios avec perturbations pour le management de la logistique. Une variation de Dijkstra est appliquée pour le modèle avec comme objectif la minimisation du temps de parcours. Basé sur ce modèle simple, un autre algorithme est développé avec la prise en compte de perturbations comme les embouteillages et les pannes. En outre, le critère temps, la complexité du chemin est prise en compte dans la fonction objectif pour ce nouveau problème. L'algorithme de colonies de fourmis est utilisé pour ce problème d'optimisation multiobjectif "avec efficacité". Dans [Rizzoli et al., 2007], les travaux de recherche s'adressent à l'application de la métaheuristique de colonies de fourmis à plusieurs problèmes réels de tournée de véhicules tels que la livraison pour une chaînes de magasins avec des fenêtres de temps, le ramassage et le dépôt des marchandises pour la distribution dynamique.

II.3.6 Optimisation dans le domaine de transport

Nous citons à présent les travaux de recherche traitant les problèmes d'optimisation dans le domaine du transport. Ils sont soit relatifs à la recherche d'itinéraires dans un contexte monomodal ou multimodal, soit relatifs à la régulation du trafic, soit ils concernent le Transport à la Demande (TAD) ou encore les problèmes de tournées de véhicules pour le ramassage et le dépôt de personnes ou de marchandises etc.

Pour les problèmes de la recherche d'itinéraires, on applique des algorithmes spécifiques avec une adaptation de celui de Dijkstra dans [Kammoun, 2007] [Feki, 2010] ; on emprunte les méthodes d'optimisation basée sur les colonies de fourmis dans [Zidi, 2006], dans ce même travail, une procédure interactive est adoptée avec une utilisateur pour la recherche d'itinéraire et d'aide aux déplacements ; dans [Hizem, 2008], un modèle dynamique des réseaux routiers tenant en compte de l'incertitude est proposé avec l'application des algorithmes du plus court chemin, avec une étude de l'optimisation monocritère et multicritère Pour les problèmes de la régulation du trafic, les travaux dans [Fayech, 2003] proposent une approche basée sur les algorithmes génétiques pour la régulation du trafic. Pour les problèmes de TAD, les algorithmes d'optimisation sont proposés pour organiser le routage de véhicules avec la prise en compte des demandes dynamiques. Dans [Horn, 2002], des heuristiques sont proposées pour l'algorithme de cheminement afin d'optimiser les insertions des demandes, après chacune insertion une optimisation de tous les itinéraires est également proposée. Concernant le problème de tournée de véhicules pour les marchandises, on cite les travaux effectués dans [Wang

et al., 2006] qui propose un algorithme en deux phases : dans la première étape, l'ensemble des clients sont subdivisés en plusieurs zones distinctes basées sur la notion de classification hiérarchique ; puis on cherche les solutions avec une version adaptée des algorithmes génériques. Dans ce même contexte, mais pour le transport de personnes, nous pouvons citer [Diana and Dessouky, 2004] qui traite les problèmes d'optimisation de transport à la demande de personnes avec la prise en compte des fenêtres horaires de service. L'heuristique proposée vise à minimiser la fonction généralisée en prenant en compte le temps supplémentaire de trajet pour chaque client en comparaison avec le trajet direct, la distance parcourue par les véhicules et le temps où les véhicules sont non occupés. Avec cette approche proposée, les demandes dont les heures de départ sont les plus proches et les points d'origines sont les plus éloigné de celui du dépôt sont insérées en premier lieu ; ensuite, les autres demandes sont insérées en suivant une procédure spécifique minimisant la hausse du coût généralisé de chaque itinéraire.

Dans la section suivante, nous introduisons les systèmes multi-agents en présentant ses caractéristiques ainsi que leurs domaine d'application.

II.4 Système multi-agents

II.4.1 Introduction

Les systèmes multi-agents ont vu le jour dans les années 70 aux États-Unis et ont connu un développement important. Dans un système multi-agents, plusieurs entités sont mis ensemble pour la résolution d'un même et seul problème. Ces entités sont capables de communiquer entre eux.

Avec leurs caractères, les SMA favorisent la mise en œuvre des processus distribués sur des entités différentes et autonomes pour obtenir l'optimalité. Ils sont dotés d'une des Intelligences Artificielles Distribuées (IAD), les SMA sont employés dans plusieurs domaines comme la communication, la production, et surtout pour simuler des scénarios.

II.4.2 Concept de SMA

Le système multi-agents est composé d'un ensemble d'agents ; des entités autonomes et interactives. Un concept lié aux SMA est la notion d'organisation qui décrit la manière suivant laquelle les différentes entités interagissent entre elles. Par ailleurs, les agissements et communications entre les agents qui sont décrits dans l'organisation régissent les relations et la dynamique des interactions entre les agents. Plus concrètement, un

SMA consiste en un environnement, avec ensemble d'objets passifs, et un ensemble d'agents actifs et avec des relations qui les relient entre eux, possédant des capacités opératoires et qui sont régis par un ensemble de lois universelles.

II.4.2.1 Définitions élémentaires

Le terme *agent* bénéfice de plusieurs identifications et spécificités qui se rapprochent incontestablement.

D'après la définition dans [Ferber, 1999] [Ferber, 1995], un agent est une entité virtuelle (logicielle) ou physique autonome :

- capable d'agir dans un environnement
- doté de capacité de communication la reliant directement aux autres agents
- mue par un ensemble de tendances (sous forme d'objectifs individuels ou d'une fonction de satisfaction voire même de survie qu'elle veut optimiser)
- en possession de ressources qui lui sont propres
- doté d'aptitudes de perception des parties limitées de son environnement
- possède une représentation partielle, voire même nulle de l'environnement dans lequel il évolue
- doté d'un certain nombre de compétences et apte à offrir un ensemble de services
- capable éventuellement de se reproduire
- suit un comportement qui lui est spécifique dans le but d'atteindre ses objectifs fixés et en prenant en considération ses perceptions et représentations et en fonction aussi des communications et messages dont il est réceptif. Ceci est faisable notamment grâce aux compétences et ressources qu'il possède.

D'après [Ferber, 1999] [Ferber, 1995], le paradigme agent ici est spécifié par rapport à l'environnement où il évolue.

À partir de la définition citée ci-dessus, le cycle de vie d'un agent est composé de trois grandes étapes et permet de retracer l'activité d'un agent dès sa création :

(a) La *perception* de l'environnement : l'agent utilise ses connaissances et comportements pour le faire ;

(b) La *prise de décision* : consiste à décider de l'action suivante à entreprendre ;

(c) L'*exécution* : consiste concrètement à effectuer l'action choisie lors de la précédente étape du cycle de vie de l'agent.

II.4.2.2 Types des agents

D'après la typologie des agents qui décrit la différence comportementale entre eux, il existe deux types principaux :

(a) Les agents réactifs : Ce type d'agents réagit rapidement pour répondre aux stimuli venant de l'extérieur. Pour le faire, ils restent toujours en état de veille, en attendant les changements potentiels ;

(b) Les agent cognitifs : Contrairement aux agents réactifs, ces agents réfléchissent avant de répondre aux stimuli, ils utilisent leurs connaissances de l'environnement et de l'ensemble des autres agents.

II.4.2.3 Propriétés des agents

Un certain nombre de propriétés sont indispensables pour tout agent. Ici, nous présentons une liste de cinq propriétés typiques [Wooldridge and Jennings, 1995] [Jennings et al., 1998] [Russell et al., 1995].

(a) L'*autonomie* : l'autonomie d'un agent implique que l'existence de celui ci ne dépend pas de celle des autres. Selon [Russell et al., 1995], cette propriété est considérée comme la capacité d'agir et réagir sans l'assistance humaine ou la présence des autre agents. Donc, différent d'un objet, un agent est une entité logicielle qui peut répondre elle-même à un appel de méthode ou aux compétences qu'elle intègre d'après sa propre décision. De ce fait, il peut initier une action ou réorienter l'exécution d'un programme sans intervention externe.

(b) La *réactivité* : cette propriété définit la capacité de réagir aux modifications de l'environnement où il se trouve et fait son évolution pour adapter sont comportement à un milieu changeant.

(c) La *proactivité* : cette propriété propose que l'agent possède la capacité d'agir de sa propre initiative pour parvenir aux objectives fixés.

(d) La *continuité* : cette notion présente un agent comme une entité logicielle implanté sur des processus légers nommé *threads* ayant une exécution parallèle. Cette propriété est définie comme étant la persistance d'une identité et d'un état sur une longue période.

(e) La *sociabilité* : les agents arrivent à communiquent entre eux afin d'interagir en utilisant des langages spécifiques et dédiés. Cette propriété de communication est le moteur principal à un comportement social évolué.

II.4.3 Caractéristiques des systèmes multi-agents

Grâce aux propriétés spécifiques des agents, les SMA manifestent plusieurs caractéristiques qui les permettent de coopérer dans une société afin de résoudre le problème en question.

II.4.3.1 Communication

Comme les agents ont une capacité d'interagir et d'échanger de l'information, il faut que ces entités communiquent afin d'accomplir leurs tâches. Il existe principalement deux types de langages de communication. Pour le premier type dit déclaratif, il se base sur des phrases déclaratives (comme des hypothèses, des définitions) ; le deuxième type la communication est basée sur un contenu exécutable est dit procédural. En réalité, la plupart des langages sont du premier type, le plus connu est KQML.

II.4.3.2 Coordination

Le processus de coordination entre les agents est développé pour s'assurer que les agents individuels formant la communauté puissent agir d'une façon cohérente et harmonieuse [Sycara, 1998]. Le processus de coordination permet d'empêcher les comportements désordonnés des agents. Pour coordonner les agents, les tâches sont attribuées, le processus de coordination est réalisé via l'affectation aux agents des responsabilités et des ressources nécessaires pour résoudre les problèmes. Les tâches sont allouées lors de la conception du système multi-agents avec la planification de la procédure de résolution du problème. Du côté agent, la coordination est une phase pour construire les actions séquentielles avec la considération des objectifs, des capacités et des contraintes environnementales. Cette planification en avance est souvent considérée comme une pratique obligatoire pour éviter les chevauchements et les conflits entre les entités. La coordination est donc un processus adopté entre les agents coopérants dans le but de réussir leur coopération et objectif.

II.4.3.3 Négociation

Il s'agit d'une méthode de coordination et de résolution du problème basée sur le principe de la communication entre les agents d'un même groupe. Cette méthode permet à une communauté d'agents d'établir un accord pour entreprendre une certaine action. Au cours de l'exécution du processus, il y a des échanges d'informations, des relaxations des buts initiaux, des concessions ou bien des menaces, cependant, ce processus a pour but principal de trouver un consensus. Parmi les types de négociations, on trouve la

coopération qui est appliquée dans les cas où les agents partagent un but commun envisagé par le système ; en revanche, celui de la compétitivité concerne les situations où les agents ont différents intérêts et essayent de faire un choix de compromis avec des alternatives bien fixées.

II.4.3.4 Formation de coalition

La formation de coalition des agents montre plusieurs avantages, parmi lesquels nous citons les deux plus importants :

— Le fait que la formation de coalition est dépendante du contexte du problème permet aux agents d'adapter leurs objectifs et comportements ;

— Le concept d'engagement temporaire permet aux participants de la coalition de réviser dynamiquement leurs intérêts, et donc les objectifs.

II.4.4 Applications des SMA dans le domaine de transport

Les SMA ont connu un grand développement dans le domaine de transport où plusieurs travaux de recherche les ont abondamment utilisés. Les premières applications développées avec le paradigme agent, concerne la surveillance de véhicules en 1991 [Durfee and Lesser, 1991]. Si nous nous concentrons particulièrement sur le cas où les SMA sont pris comme support de base, nous pouvons citer les travaux suivants : [Mandiau et al., 2002] sur la simulation de trafic automobile. [Fayech, 2003] sur la gestion des perturbations et de la régulation dans les réseaux de transport multimodal. [Zidi, 2006] dans la recherche et composition d'itinéraires optimisés pour l'aide au déplacement des voyageurs. [Feki, 2010] sur un système d'aide à la décision pour les voyageurs en termes de plan de déplacements avec la prise en compte des perturbations du réseau. [Sghaier, 2011] sur l'optimisation d'un système de covoiturage dynamique. [Durfee and Lesser, 1991] sur la surveillance des véhicules automatisés, [Fischer et al., 1995] sur le transport de marchandises et [Chaib-draa, 1996] sur la gestion de trafic urbain.

Nous détaillons ici une application de SMA dans le cadre de la régulation des correspondances. Dans son travail Laichour présente un modèle basé sur trois types d'agent : Agent Acquisition, Agent Correspondance et Agent Superviseur [Laichour et al., 2001] qui ont pour différentes missions :

— Agent Acquisition : effectue la gestion des données correspondantes aux passages des bus aux arrêts de régulation ;

— Agent Correspondance : détecte et diagnostique les perturbations au niveau des correspondances et propose des solutions ;

— Agent Superviseur : sert comme interface entre le régulateur et le système d'aide à la régulation des correspondances.

II.4.5 Implémentation des SMA pour le problème posé

Nous aspirons à modéliser notre système avec les SMA et utilisons l'optimisation qui sera présentée dans la prochaine section.

Compte tenu de la problématique présentée dans le Chapitre I, nous allons proposer un système d'information multi-agent ouvert et adapté aux besoins d'un système d'information de transport comodal dans lequel différents sous-systèmes sont intégrés.

Au sein d'un SMA, les différents agents ayant des rôles variés forment une société de coopération et assurent le bon fonctionnement du système. Il existe des agents responsables de communication système-utilisateur, des agents de coordination et de coopération, des agents d'optimisation etc. La coalition entre certains agents permet de réaliser les tâches d'optimisation en vue d'obtenir des solutions satisfaisantes face aux différentes demandes d'utilisasteurs et aux perturbations du réseau de transport. L'implémentation de SMA sera abordée avec plus de détails dans Chapitre IV.

Ensuite, nous présentons l'optimisation en rappelant notamment la relation entre l'optimisation et SMA.

Dans la section suivante, nous proposons de profiter de l'alliance entre l'optimisation et l'approche basée sur le paradigme agent tout en se positionnant dans le contexte de notre propre problématique.

II.5 Alliance entre optimisation et SMA à l'avantage de transport comodal

Plusieurs travaux de recherche montrent l'efficacité de l'alliance entre les SMA et l'optimisation pour la résolution des problèmes variés. Les algorithmes et méthodes d'optimisations sont intégrés au sein des agents en sorte que les méthodes d'optimisations soient adaptées aux objectifs et contraintes des agents. Cette alliance fait émerger, en effet, deux couches d'optimisations, à savoir : une optimisation globale dite distribuée, et une optimisation locale intégrée dans les comportements des entités d'agent. Dans [Zidi, 2006], on a traité le problème de l'optimisation de la recherche des informations des itinéraires au sein d'un réseau de transport multimodal distribué en appliquant la méthode du paradigme mobile agent. Cet agent est responsable des traitements des requêtes en

se déplaçant entre les différents nœuds du réseau de transport. Le travail dans [Meng et al., 2007] propose une solution pour le problème de génération des épreuves pour les étudiants en associant un système d'agents et un algorithme générique. Dans [Feki, 2010] qui profite également de cette association des SMA et l'optimisation, on propose un système distribué d'aide à la décision pour les voyageurs avec la prise en compte des perturbations du réseau de transport. Le travail effectué dans [Sghaier, 2011] étudie l'optimisation d'un système de covoiturage dynamique avec la modélisation et la partition des zones desservies. Dans [Satunin and Babkin, 2014], un système de transport à la demande reposant sur l'alliance d'un SMA et d'une méthode de vote combinatoire est proposé. Nous pouvons marquer que les travaux de recherche cités comme [Zidi, 2006] [Feki, 2010] [Sghaier, 2011] [Satunin and Babkin, 2014], comme le nôtre, traitent le problème d'optimisation d'un système d'information distribué.

II.5.1 Optimisation pour le transport comodal

Il s'agit de plusieurs niveaux d'optimisation pour le transport comodal. La planification d'itinéraire avec Dijkstra peut être considérée comme une résolution avec la programmation dynamique en quelque sorte. En effet, la réduction de l'espace de recherche avec des heuristiques (par exemple la recherche bidirectionnelle) conduit à l'amélioration de performance de l'algorithme.

Si une demande cherche des itinéraires non-dominés avec plusieurs critères (par exemple le temps de parcours et la dispense totale du trajet), cela va entraîner la recherche des solutions Pareto-optimales.

L'attribution des ressources de transport aux utilisateurs correspond à un problème d'affectation classé sous la famille de problèmes *NP-difficiles* dont la complexité est exponentielle. Il faut donc emprunter la bonne stratégie de résolution en vue d'éviter l'explosion combinatoire.

II.5.2 SMA pour le transport comodal

Différents modes de transport sont intégrés dans un système de transport comodal, y compris les véhicules partagés. Avec la considération du fait que les ressources et données de transport sont distribuées, le problème se trouve dans un environnement dynamique en évolution continuelle. Cela nécessite une mise en place d'un système avec l'optimisation des opérations en vue d'obtenir une amélioration en termes de temps de réponse, de ressources nécessaires etc. Compte tenu des capacités des agents comme l'autonomie, la communication, la collaboration et la négociation, le concept de SMA

s'adapte à l'environnement du problème dans lequel ils évoluent et permet de fournir des solutions efficaces et robustes dans un cadre dynamique assujettissant.

II.5.3 Alliance optimisation-SMA et positionnement

Dans le but de concevoir un système qui puisse accueillir autant d'utilisateurs que possible et s'occuper au même moment des données distribuées et combinatoires, nous employons le concept SMA. L'intégration de l'idée d'optimisation aux SMA permet de mettre en œuvre deux couches d'optimisation locale et distribuée pour offrir des services optimaux en termes de préférences des utilisateurs. La combinaison de SMA et optimisation a pour effet d'intégrer le sens d'optimisation au sein des agents par la bonne pratique d'arranger leurs comportements et de diriger leurs actions dans le sens de la recherche des solutions optimales.

Les agents qui intègrent l'optimisation locale dans leurs actions et comportements se trouvent eux-mêmes dans un contexte d'optimisation globale et distribuée. Ce contexte est concrétisé par coopération, coordination et coalition entre les agents qui dédient l'optimisation locale et leurs interactions à l'avantage de l'intelligence collective.

Il convient de noter ici un type d'agent spécifique appelé Route Agent qui sera défini en détail dans le Chapitre IV. Les tâches principales de cet agent sont de représenter une section route et de chercher des affectations possibles et optimisées des voyageurs aux véhicules. Des comportements spécifiques des agents sont voués à assurer l'exécution d'algorithmes d'affectations comme optimisations locales, par exemple avec les algorithmes génétiques. La coalition entre des Route Agents permet de relier des segments de routes dans le but de formuler des itinéraires complets pour chaque voyageur. Ce type d'agent est doté de comportements appropriés pour coopérer avec les autres, nous allons expliquer avec plus de détails(c.f. Chapitre IV).

Dans le cadre de cette thèse, l'environnement de transport comodal accueillant des différents systèmes d'information de transport pour des différents modes peut être considéré comme distribué. De plus, un procédé d'optimisation distribuée est apte pour la mise en place des traitements en parallèle. Une architecture distribuée, basée sur un SMA, est un choix juste pour mettre en œuvre notre système. Avec cette architecture qualifiée d'étendue et de flexible, différentes approches d'optimisation sont prêtes à être intégrées.

II.6 Conclusion

Dans ce chapitre, nous avons introduit les fondements théoriques de notre travail qui s'appliqueront dans les chapitres suivants. Le chapitre a débuté par la présentation de la théorie des graphes et les algorithmes pour la recherche d'itinéraire. L'état de l'art sur l'optimisation et les Systèmes Multi-Agents a ensuite été exposé. Une proposition de combinaison de ces deux techniques a été évoquée et servira dans notre travail.

Chapitre III

Planification d'itinéraire et problème d'affectation au sein d'un système de transport comodal

Sommaire

Where there's a will, there's a way.
Proverbe anglais

III.1 Introduction

Dans le chapitre précédent, nous avons présenté les fondements théoriques sur lesquels se base notre travail. Dans ce chapitre, nous nous intéressons à la planification d'itinéraire dans un environnement de transport comodal, ainsi qu'au problème d'affectation des véhicules aux voyageurs. L'optimisation d'itinéraire nécessite une mise en compétition de l'ensemble des modes disponibles sur la base d'un ensemble de critères et de contraintes.

Nous avons organisé ce chapitre de la manière suivante : tout d'abord, nous proposons le modèle de transit afin de gérer les données correspondantes aux transferts entre les différents modes. Ensuite, nous détaillons la modélisation d'un graphe et la planification d'itinéraire. Les solutions trouvées peuvent être multiples, nous abordons alors, le problème de l'optimisation d'une chaîne de déplacements en respectant les contraintes temporelles ainsi que les algorithmes génétiques qui sont utilisés pour la résolution des problèmes d'affectation.

III.2 Modèle de *transit*

Afin d'avoir des résultats précis, les correspondances entre les différents modes sont pris en compte lorsque la planification d'itinéraire se trouve dans un environnement de transport intégrant plusieurs modes de transport. Le modèle développé prend en considération des données liées au changement de mode.

Nous employons le terme de *transit* pour décrire le transfert inter-opérateurs d'un itinéraire. Ces types de transferts inter-opérateurs ne peuvent se faire souvent que dans des endroits dédiés où coexistent plusieurs modes de transport. Par exemple d'un parc-relais où l'on dépose sa voiture vers une station de transport en commun pour continuer

son trajet, ou dans la sens inverse. Une section de route pédestre est donc concernée pour arriver au *Point Info* de la section suivante, comme le montre la Figure III.1.

Définition. *Point Info* : Il se réfère au lieu où s'affiche les informations nécessaires pour effectuer un trajet de voyage du côté des voyageurs. À titre d'exemple, les panneaux d'informations dans les gares de train sont installés pour indiquer les informations sur le numéro de train, le temps de départ, la plateforme correspondante, ou encore le retard etc.

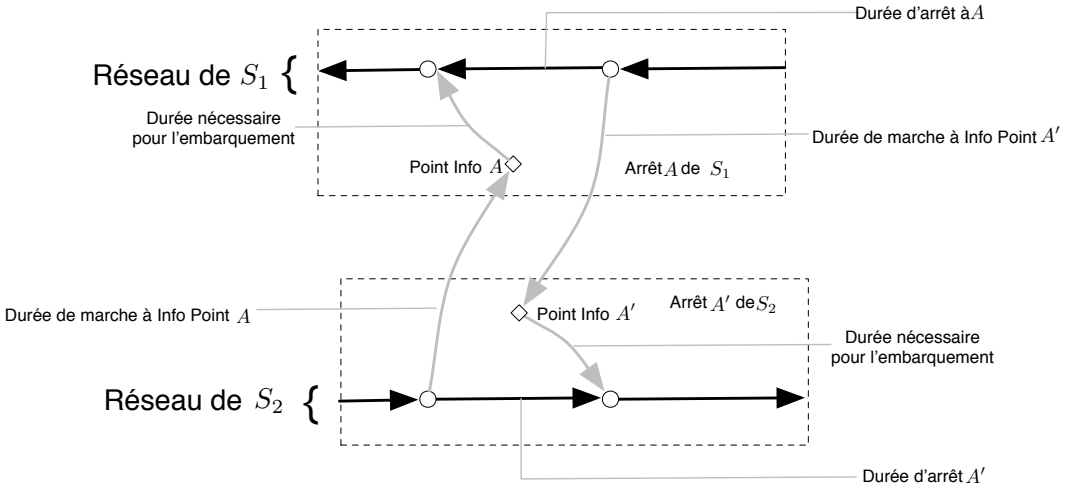


FIGURE III.1: La représentation microscopique d'un transfert

Les informations nécessaires se réfèrent aux plateformes et aux horaires pour les trains, au tableau de marche horaire pour les bus etc. Plusieurs types de transport exigent la présence du voyageur quelques minutes avant l'horaire de départ du véhicule pour assurer l'embarquement. À titre d'exemple, il faut se présenter au moins 2 minutes avant l'horaire de départ dans le cas d'un voyage en TGV en France, dans le cas contraire, l'accès n'est pas garanti. Le fait que les gares sont rarement déplacées après leurs installations rend facile la collecte des données pour créer une base de donnée de transfert à pied. La durée nécessaire est basée sur plusieurs éléments :

- un temps de sortie du mode t_s ;
- un temps de correspondance t_c basé sur la distance à parcourir et la vitesse moyenne de la marche à pied (la distance pédestre est la vitesse moyenne de la marche à pied et le produit de la durée nécessaire) ;
- un temps d'accès au mode t_a.

La durée nécessaire pour ce transit est alors égale à leur somme : $t_{tr} = t_s + t_c + t_a$. Cependant, il peux exister aussi le temps passé à attendre un autre moyen de transport en cas de correspondance dans les scénarios réels.

La Figure III.1 montre un exemple de transfert inter-opérateurs. Il existe un transfert possible entre A et A' qui appartiennent à deux réseaux de modes différents. Si le coût de transfert signifie le temps nécessaire, il convient de noter que le coût de transfert $c(A, A')$ n'est pas nécessairement le même que $c(A', A)$ pour le sens inverse.

Ce modèle permet de créer et de maintenir la base de donnée du temps de transfert.

La Figure III.2 illustre un exemple d'un transfert inter-opérateurs au sein du système, où S_1 et S_2 sont deux sous-systèmes et les nœuds A et A' sont les points de transferts qui sont reliés par les arcs pédestres. Le temps de transfert t_1 et t_2 peuvent être obtenus à partir des données publics. Les arcs pédestres de transfert vont servir comme connexion entre deux sections de routes adjacentes correspondants à deux opérateurs différents.

III.2.1 Représentation du transit

Nous notons par *classe* C un sous-système dans un système distribué et par m le mode d'un sous-système (un sous-système est représenté par un mode). Pour un mode m_i, le réseau de transport du sous-système correspondant *classe* C_i est modélisé par un graphe G_i. Ce dernier est appelé le mode graphe de m_i ou le graphe de C_i.

$G_i = (V_i, A_i)$ est un graphe directionnel avec des poids des arcs positifs où V_i désigne l'ensemble de nœuds de C_i et A_i désigne l'ensemble des arcs de C_i :

- $V_i = \{v|$sur un noeud v, les ressources de mode m_i sont disponbibles$\}$;
- $A_i = \{(u, v)|$il existe un lien direct (sans arrêt intermédiaire) de mode m_i de u à v avec $u \in V_i, v \in V_i, u \neq v\}$.

M est un ensemble de modes, $M = \{m_i | i \in \{1, 2, \ldots, N\}\}$ où N est le nombre de sous-systèmes dans le système distribué. Le changement de mode pendant le voyage signifie qu'il existe un transit entre les modes. Comme le montre dans la Figure III.2, A et A' sont deux points de transit.

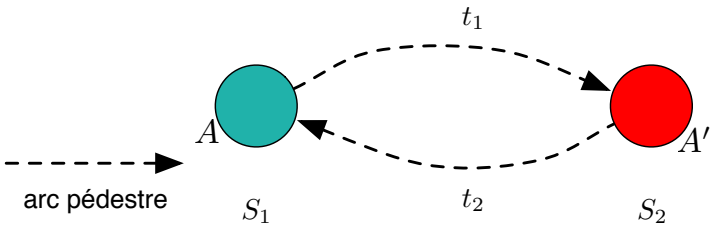


FIGURE III.2: La représentation macroscopique de transfert inter-opérateur

Le graphe intégral du système entier est $G_M = (V, A)$ où

- $V = \cup_{i=1}^{n} V_i$;

— $A = (\cup_{i=1}^{n} A_i) \cup A_{Transit}$ avec $A_{Transit}$ l'ensemble d'arcs de transit.

Définition. *Connexion de transit* : Une connexion de transit γ est un 5-tuple (m_f, m_t, v_f, v_t, c) défini sur $M \times M \times V \times V \times \mathbb{R}^+$. Ce concept décrit l'arc d'un transit de nœud v_f dans le graphe de mode m_f vers un nœud v_t dans le graphe de mode m_t avec un coût c.

Définition. *Tableau de Transit* : Le Tableau de Transit réunit toutes les connexions de transit au sein d'un graphe G.

Nous considérons deux nœuds : v_i^k de G_i et v_j^l de G_j.

Quand $i = j$, il s'agit d'un transfert au sein d'un même sous-système.

Quand $i \neq j$, nous le définissons avec l'Équation III.1 :

$$\gamma_{(i,j)}^{(k,l)} = \begin{cases} (m_i, m_j, v_i^k, v_j^l, c), & \text{si une connexion de transit existe} \\ Null, & \text{sinon} \end{cases} \tag{III.1}$$

Nous utilisons $\gamma_{(i,j)}^{(k,l)}$ pour décrire $\gamma(v_i^k, v_j^l)$ le transit de v_i^k à v_j^l.

III.2.2 Exemple d'illustration

La Figure III.3 illustre le graphe d'un système composé de trois sous-systèmes. Les graphes G_1, G_2, G_3 sont reliés par des connexions de transit.

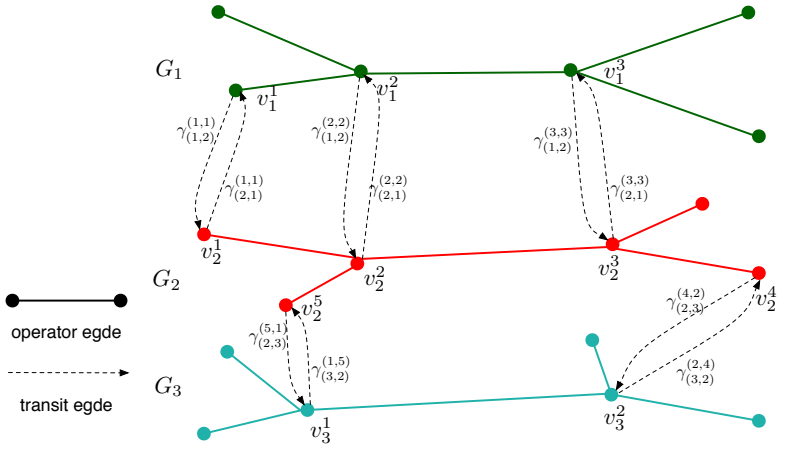


FIGURE III.3: Graphe avec arcs de transit à trois composants

Le tableau de transit du système présenté dans la Figure III.3 est construit d'après l'équation III.1. Comme le montre le Tableau III.1, les nœuds des colonnes sont les nœuds de départ et ceux des lignes sont les nœuds d'arrivée. Les transferts au sein des

sous-systèmes ne sont pas pris en compte dans ce tableau. La non existence de connexion de transit entre les nœuds est représentée par N dans les cellules du tableau.

TABLEAU III.1: Le tableau de transit pour les trois composants G_1, G_2 et G_3

De \ À		G_1			G_2					G_3	
		v_1^1	v_1^2	v_1^3	v_2^1	v_2^2	v_2^3	v_2^4	v_2^5	v_3^1	v_3^2
G_1	v_1^1				$\gamma_{(1,2)}^{(1,1)}$	N	N	N	N	N	N
	v_1^2				N	$\gamma_{(1,2)}^{(2,2)}$	N	N	N	N	N
	v_1^3				N	N	$\gamma_{(1,2)}^{(3,3)}$	N	N	N	N
G_2	v_2^1	$\gamma_{(2,1)}^{(1,1)}$	N	N						N	N
	v_2^2	N	$\gamma_{(2,1)}^{(2,2)}$	N						N	N
	v_2^3	N	N	$\gamma_{(2,1)}^{(3,3)}$						N	N
	v_2^4	N	N	N						N	$\gamma_{(2,3)}^{(4,2)}$
	v_2^5	N	N	N						$\gamma_{(2,3)}^{(5,1)}$	N
G_3	v_3^1	N	N	N	N	N	N	N	$\gamma_{(3,2)}^{(1,5)}$		
	v_3^2	N	N	N	N	N	N	$\gamma_{(3,2)}^{(2,4)}$	N		

III.3 Réseau de transport et graphe hiérarchisé

La structure hiérarchisée est un moyen efficace pour modéliser les niveaux des réseaux avec des transferts [Bielli et al., 2006]. Selon le travail [Van Nes, 2002], le concept de la hiérarchie est réalisé lors de la phase de la conception des réseaux de transport. L'approche hiérarchisée avec graphe en plusieurs niveaux est introduite pour traiter et accélérer la recherche du plus court chemin en utilisant les horaires des trains [Schulz et al., 2002]. Les graphes se superposent et les arcs supplémentaires sont ajoutés. Ce travail a été développé dans [Holzer et al., 2009] pour avoir une meilleure performance et on réduit le nombre d'arcs supplémentaires. Dans le travail de [Kammoun, 2007], l'auteur a mentionné mais sans l'avoir développée, l'idée de l'organisation hiérarchique des SIAD (Systèmes d'Information d'Aide au Déplacement) de différents niveaux en vue de concevoir des systèmes d'information coopératif de mobilité.

En effet, comme la taille du réseau (cardinalité de l'ensemble de nœuds et celle de l'ensemble d'arcs) de transport est souvent importante, il faut mettre en place une représentation condensée et efficace du réseau afin de limiter la consommation de la mémoire et du temps de calcul lors de l'utilisation des algorithmes de la recherche des itinéraires.

Dans le cadre de nos travaux de recherche, nous proposons une autre méthode hiérarchisée appliquée au domaine qui nous intéresse ; à savoir le transport multimodal (en comparaison avec les travaux cités [Schulz et al., 2002] [Holzer et al., 2009] qui sont consacrés pour les trains). En effet, cette approche proposée permet de hiérarchiser le graphe au

même temps de réduire l'espace de recherche éventuellement lors de l'exécution de l'algorithme. Elle fait partie des techniques d'accélération de la planification d'itinéraire et s'adapte particulièrement au problème traité.

III.3.1 Différents modes de transport

Nous souhaitons représenter de façon la plus complète possible l'ensemble des modes de transport collectifs pour la mobilité des voyageurs : transport en commun (métro, train, bus, tramway, etc.), véhicules partagés (autopartage, covoiturage) voir la marche à pied. Pour la modélisation de réseaux d'opérations, les modes sont aussi regroupés en fonction des contraintes qu'introduit leur usage sur la suite d'un trajet.

III.3.1.1 Réseau pour les transports publics unimodaux

Chaque réseau de transport en commun unimodal de mode m est représenté par un graphe directionnel G_m. Les services de transports en commun sont basés sur des horaires fixes et des itinéraires prédéfinis. Par exemple, une ligne de transport public correspond à une suite fixe d'arrêts desservis plusieurs fois une journée.

III.3.1.2 Réseau routier

Étant donné que la voiture personnelle n'est pas pris en compte dans ce travail, le réseau routier est dédié au covoiturage et à l'autopartage ou encore aux piétons. En effet, les stations de l'autopartage pour les flottes de véhicules sont bien installées. Nous pouvons donc les considérer comme des arrêts pour les transports publics.

III.3.2 Regroupement des graphes hétérogènes avec liens de transit

III.3.2.1 Définitions

Ce paragraphe pose les définitions importantes qui seront utilisées par la suite dans ce chapitre.

Définition. *Graphe unimodal* : Un graphe $G_m = (V_m, A_m)$ est dit unimodal s'il représente un réseau de transport unimodal, où V_m est l'ensemble de nœuds, A_m est l'ensemble d'arcs et m désigne le mode de transport utilisé dans le réseau.

À titre d'exemple, le graphe qui représente uniquement un réseau de métro ou celui de vélo-partage est un graphe unimodal.

Définition. *Graphes hétérogènes* : Les graphes unimodaux qui modélisent les réseaux de transport de différents modes sont des graphes hétérogènes.

Dans le monde réel, plusieurs graphes hétérogènes pourraient coexister même pour représenter le réseau de transport d'une ville. Les graphes correspondants au réseau de bus et celui de métro sont sans doute deux graphes hétérogènes. Généralement, les graphes hétérogènes sont *disjoints*. Avec la possibilité de transfert entre ces différents modes de transport sur quelques nœuds, les graphes peuvent y être reliés à l'aide des liens de transit.

Définition. *Graphe multimodal* : Un graphe est dit multimodal s'il représente un réseau de transport multimodal.

Nous notons ce type de graphe par $G_M = \{V_M, A_M, M\}$ ou simplement $G_M = \{V_M, A_M\}$ avec

- $M = \{m_i | i \in \{1, 2, \ldots, N\}\}$ est l'ensemble de modes d'un graphe multimodal ;
- $V_M = \bigcup_{i=1}^{N} V_i$ est l'ensemble de nœuds ;
- $A_M = \bigcup_{i=1}^{N} A_i$ est l'ensemble d'arcs.

Dans la définition du graphe multimodal, les nœuds où les transferts peuvent avoir lieu sont fusionnés et donc considérés comme un seul nœud dans ce graphe. Cela permet la jonction entre les graphes hétérogènes.

La définition d'un chemin (itinéraire) au sein d'un graphe simple est établie (cf. section II.2.1). Nous décrivons un chemin dans un graphe multimodal.

Dans un graphe $G = (V, A)$, un chemin ou un itinéraire $P(n_1, n_k)$ est une séquence d'arcs entre une paire de nœuds n_1 et n_k, $P(n_1, n_k) = ((n_1, n_2)_{m_{i_1}} (n_2, n_3)_{m_{i_2}} \ldots (n_{k-1}, n_k)_{m_{i_{k-1}}})$ où m_{i_j} est le mode de transport utilisé sur le morceau de route (n_j, n_{j+1}).

Définition. *Un chemin multimodal* : Un chemin est dit multimodal si plus qu'un seul mode sont concernés.

Lorsque les transferts entre les différents modes sont pris en compte, il faut intégrer les liens de *transit* dans le graphe multimodal. Nous ajoutons donc les arcs entre les nœuds où les transferts entre les modes existent.

Définition. *Coût d'un chemin multimodal* : Le coût d'un chemin multimodal est défini comme la somme des coûts des morceaux de route empruntés suivant l'ordre parcouru d'un point de départ au point d'arrivée.

Définition. *Graphe multimodal complet* : À partir d'un graphe multimodal, le graphe multimodal complet est défini avec l'ajout des arcs de *transit*.

III.3.2.2 Graphe de transfert : une procédure d'abstraction

Pour simplifier la représentation du graphe multimodal, des travaux de recherche ont proposé des modèles comme "l'intersection graph" [Wang and Kaempke, 2004] et le "transfer graph" [Galvez-Fernandez et al., 2009] en vue d'obtenir un modèle qui facilite la représentation du graphe multimodal et d'optimiser le temps de calcul pour le problème de cheminement.

Par la suite, nous présentons le graphe de transfert qui permet de réduire le graphe multimodal en termes de nombre de nœuds et d'arcs.

Un graphe de transfert est décrit par un ensemble de composants et un ensemble d'arcs de transit avec lesquels les composants sont connectés. Nous le notons par $G_{Tr} = \{V_{Tr}, A_{Tr}, C\}$ où

- $C = \{C_i | i \in \{1, 2, \ldots, N_C\}\}$ est l'ensemble de composants ; pour un composant $C_i = \{V_i, A_i\}$ qui correspond au graphe unimodal G_i, V_i est l'ensemble de nœuds de transfert de G_i, A_i est l'ensemble d'arcs reliant les nœuds de transfert.
- $V_{Tr} = \bigcup_{i=1}^{N_C} V_{C_i}$ est l'ensemble de nœuds des composants ;
- $A_{Tr} = (\bigcup_{i=1}^{N_C} A_{C_i}) \bigcup A_{Transit}$ est l'ensemble d'arcs des composants et $A_{Transit}$ est l'ensemble d'arcs de transit reliant les composants.

La construction du graphe de transfert pour un réseau de transport multimodal peut être considérée comme une procédure d'abstraction [Galvez-Fernandez et al., 2009]. L'ensemble de nœuds du graphe est une collection des nœuds de transfert de chaque composant.

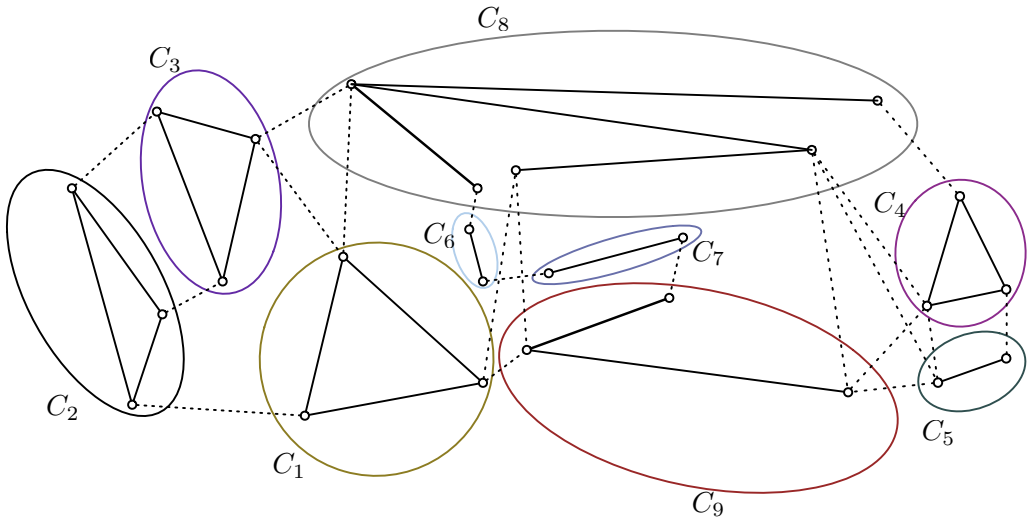


FIGURE III.4: Un graphe de transfert à 9 composants avec des liens de transit

Comme le montre la Figure III.4, nous constatons que les composants sont séparés dans le graphe de transfert par les liens de transit. Cette représentation du réseau multimodal

est plus performante et robuste au sens où une perturbation au sein d'un seul composant ne va pas entraîner des bouleversements pour la représentation globale du réseau, en effet, il met à jour uniquement le composant du réseau concerné.

III.3.3 Mise en hiérarchie du graphe

III.3.3.1 Niveaux de transfert

Les graphes unimodaux peuvent être classés en plusieurs niveaux selon les zones desservies. Nous pouvons distinguer trois niveaux : local, régional et national (il est possible de l'étendre à l'échelle internationale).

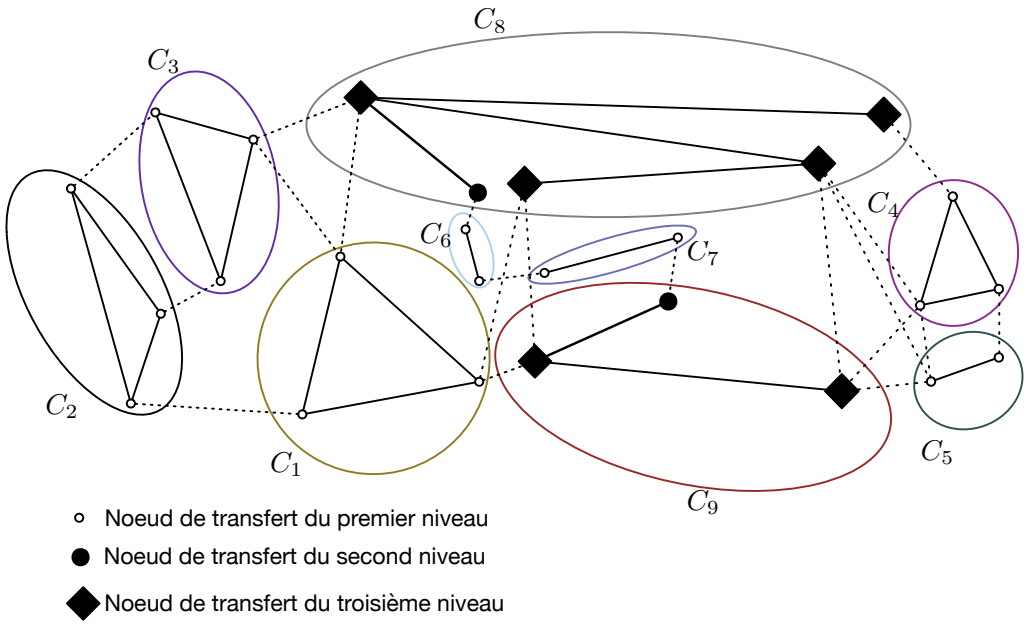


FIGURE III.5: Un graphe de transfert avec des liens de transfert et des nœuds de transfert en niveau

La Figure III.5 montre un graphe de transfert avec des liens de transfert en précisant les niveaux des nœuds. Les composants C_1, C_2, C_3, C_4, C_5, C_6 et C_7 se constituent des nœuds de niveaux plus bas, tandis que les composants C_8 et C_9 se constituent des nœuds de niveaux plus élevés. Cette représentation nous donne une impression que C_8 et C_9 jouent un rôle de liant avec les autres composants qui sont disjoints.

En outre, au sein d'un composant, il existe des nœuds qui jouent à la fois des rôles de nœud de transfert du second et troisième niveau. Pour montrer efficacement les niveaux du graphe, les nœuds de ce genre se divisent en autant de nœuds que le nombre de rôles qu'ils jouent. Par conséquent, les nœuds obtenus sont reliés avec les *arcs* n'ayant pas de poids. Une mesure supplémentaire donc permet de distinguer les nœuds au sein d'un

composant selon leur niveau. La Figure III.6 montre cette mesure en divisant certain nœuds en deux et en les reliant avec les *arcs*. Nous considérons alors comme des arcs de transit.

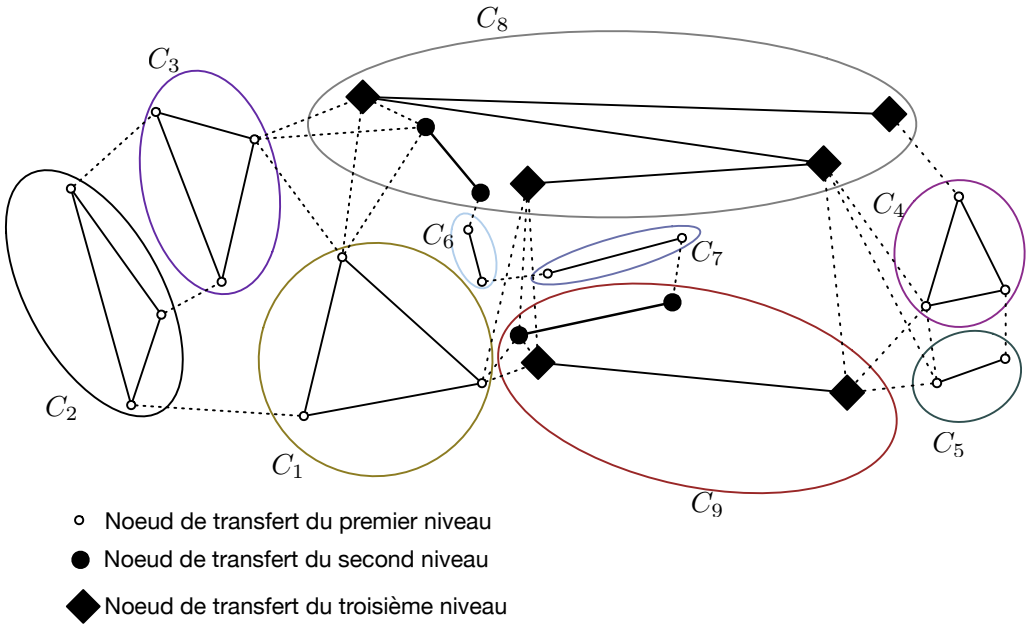


FIGURE III.6: Un graphe de transfert avec des liens de transfert et des nœuds de transfert en niveau

Le graphe hiérarchisé de G, noté comme H_G, est un graphe qui :

- met les composants en hiérarchie ;
- permet de calculer le plus court chemin $O - D$ en gardant l'optimalité.

La construction du graphe hiérarchisé se base sur deux éléments :

- le graphe de transfert avec des liens de transfert G_{Tr} ;
- la classification des nœuds selon leurs niveaux.

Nous distinguons deux types d'arcs pour construire un graphe hiérarchisé :

- arcs du niveau : les arcs des composants de même niveau et les arcs de transfert entre ces composants ;
- arcs inter-niveau : les arcs de transit reliant les composants de différents niveaux.

III.3.3.2 Construction des graphes hiérarchisés

À partir du graphe de transfert, le graphe hiérarchisé est construit en respectant les étapes suivantes :

1. regrouper les nœuds selon leurs niveaux ;
2. mettre des arcs inter-niveau.

On construit le graphe hiérarchisé $H_{G_{Tr}}$ du graphe de transfert $G_{Tr} = \{V_{Tr}, A_{Tr}, C\}$ où $C = \{C_i | i \in \{1, 2, \ldots, N_C\}\}$. On désigne l le nombre de niveaux. Les nœuds de transfert de chaque composant C_i sont classés également en l catégories. Un graphe de transfert en trois niveaux est visualisé sur la Figure III.7.

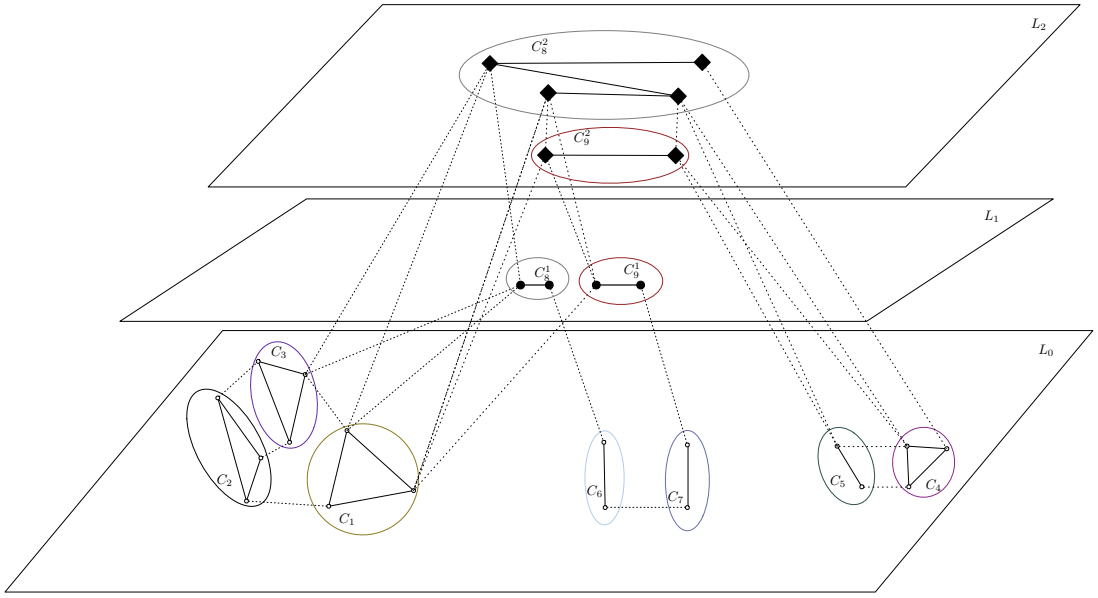


FIGURE III.7: Un graphe de transfert en trois niveaux

Définition. *Module* : Les composants de même niveau interconnectés font un module $\mathscr{M}$, dont l'ensemble de nœuds est l'union des ensembles de nœuds de ses composants, et l'ensemble d'arcs est l'union des ensembles d'arcs de ses composants et les arcs de transit.

Chaque niveau se constitue d'un ensemble de modules. On note un module de niveau k par $\mathscr{M}_k$. On rappelle que les modules de même niveau sont disjoints et les modules s'interconnectent via les arcs inter-niveau.

Compte tenu de la présence d'un ensemble de modes de transports collectifs et individuels au sein d'un module et la complémentarité entre eux, un trajet $u-v$ peut être effectué de plusieurs façons en fonction de différentes combinaisons de modes, voici une illustration avec la Figure III.8.

Définition. *Graphe de modules* : On le désigne par $G_{\mathscr{M}}$. Les modules sont considérés comme les nœuds du graphe de modules. Si deux modules sont connectés, on ajoute un arc entre ces deux nœuds et on associe à chaque arc un poids unitaire.

La Figure III.9 est un graphe de transfert en module.

Pour déterminer le sous-graphe d'une paire de nœuds $O-D$ qui est noté par $H_{G_{Tr}}(O, D)$, nous utilisons une chaîne de modules.

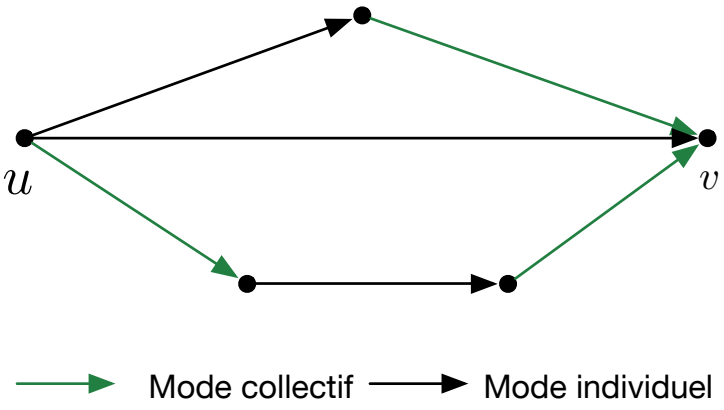


FIGURE III.8: Différentes combinaisons de modes pour la mobilité au sein d'un module

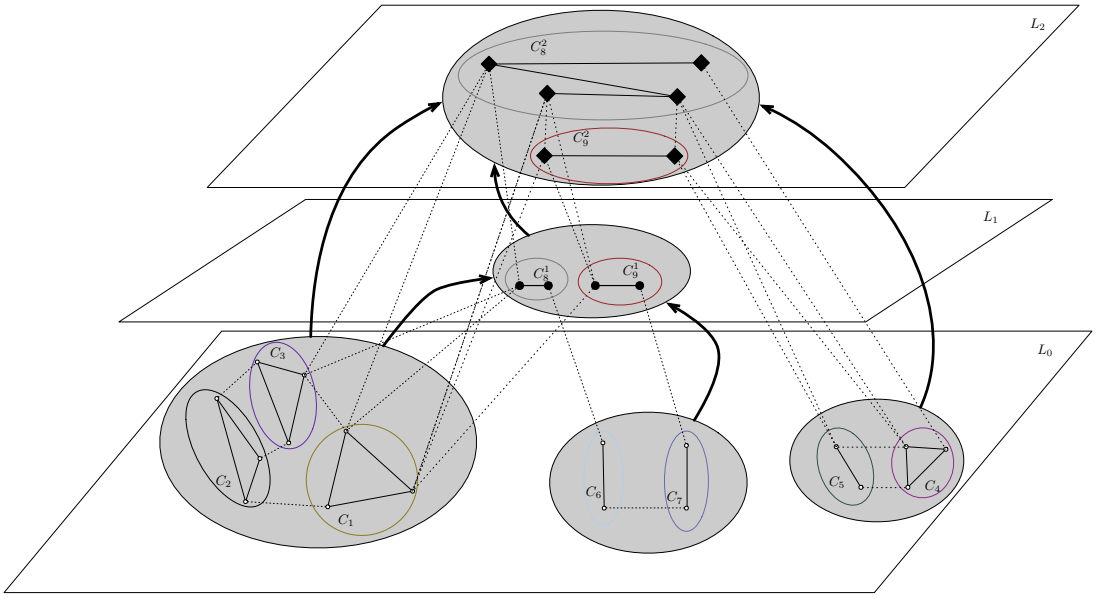


FIGURE III.9: Un graphe de transfert en module pour le graphe sur la Figure III.7

Définition. *Chaîne de modules* : Pour une paire de nœuds O et D, nous considérons la chaîne de modules $\mathscr{M}(O) - \mathscr{M}(D)$. Notons K comme le plus petit k avec $\mathscr{M}_k^O = \mathscr{M}_k^D$. Une chaîne $\mathscr{M}(O) - \mathscr{M}(D)$ est définie comme :

$$\mathscr{M}(O) \to \mathscr{M}_{k_1}^O \to \cdots \to \mathscr{M}_K^O(\mathscr{M}_K^D) \to \cdots \to \mathscr{M}_{k_1'}^D \to \mathscr{M}(D)$$

où $k_1 < \cdots < K$ et $k_1' < \cdots < K$.

Une chaîne de modules $\mathscr{M}(O) - \mathscr{M}(D)$ pour la paire de nœuds $O - D$ est visualisée sur la Figure III.10

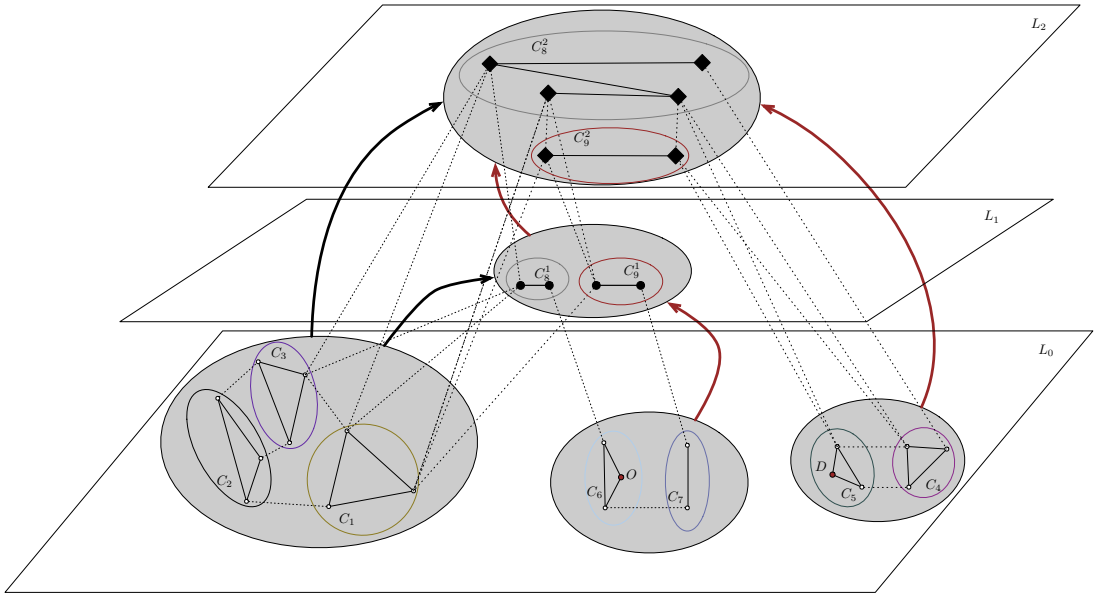


FIGURE III.10: Un graphe de transfert en module

III.4 Planification d'itinéraire sur le graphe hiérarchisé

III.4.1 Identification du domaine de recherche

Pour la planification d'itinéraire sur un réseau de transport, la procédure proposée se déroule en deux phases :

- la première phase ayant pour objectif de construire un graphe hiérarchisé du réseau de transport ;
- la seconde étape vise à identifier le domaine de recherche sur lequel l'algorithme de la planification d'itinéraire s'exécute.

La première étape est dédiée à la construction d'un graphe hiérarchisé d'un réseau de transport. La réalisation de cette phase fait appel à l'Algorithme *ACGH*.

Algorithm 4 Algorithme pour la construction du graphe hiérarchisé (**ACGH**)

1: Étape 1 : Construire le graphe de transfert G_{Tr} d'un réseau de transport
2: Étape 2 : Regrouper les nœuds de transfert selon leurs niveaux
3: Étape 3 : Construire le graphe hiérarchisé $H_{G_{Tr}}$

L'Algorithme *ACGH* illustre la procédure à suivre pour construire le graphe hiérarchisé avec lequel la seconde étape suivra pour identifier le domaine de recherche. La recherche de la plus courte chaîne de modules est décrit par l'Algorithme *PCCM*.

La plus courte chaîne de modules pour une paire de nœuds $O-D$ au sein d'un graphe de modules est la plus courte chaîne allant du module de O au module de D. Étant donné

Algorithm 5 Algorithme pour trouver la plus courte chaîne de modules entre O et D (**PCCM**)

1: Étape 1 : Construire le graphe de modules $G_{\mathscr{M}}$ du graphe de transfert hiérarchisé $H_{G_{Tr}}$
2: Étape 2 : Identifier les modules $\mathscr{M}(O)$ et $\mathscr{M}(D)$ qui contiennent respectivement O et D
3: Étape 3 : Trouver la plus courte chaîne de modules entre $\mathscr{M}(O)$ et $\mathscr{M}(D)$ ▷ le problème du plus court chemin dans le graphe $G_{\mathscr{M}}$

que les arcs du graphe de modules ont tous un poids égal à 1, l'algorithme de Dijkstra s'applique pour trouver la plus courte chaîne $\mathscr{M}(O) - \mathscr{M}(D)$. En effet, ce dernier qui est un sous-graphe de G_{Tr} noté comme $H_{G_{Tr}}(O, D)$, est le domaine de recherche pour le problème du plus court chemin entre $O - D$.

III.4.2 Problème du plus court chemin

III.4.2.1 Le plus court chemin au sein d'un module $\mathscr{M}$

Au sein d'un module se constituant des composants qui sont distribués, la recherche du plus court chemin consiste à rechercher le chemin ayant le coût minimum à travers les composants. [Wang and Kaempke, 2004] a proposé l'Algorithme *DSRA* pour résoudre ce genre de problème mais sans la prise en compte du coût de transfert. Nous proposons une approche (Algorithme *ADPC*) qui permet justement la prise en compte des transferts entre les réseaux unimodaux.

Le graphe d'un module $\mathscr{M}$ de composant $C_{\mathscr{M}}$ est défini comme $G(\mathscr{M}) = (V_{\mathscr{M}}, A_{\mathscr{M}})$ avec $V_{\mathscr{M}} = \bigcup_{C_i \in C_{\mathscr{M}}} V_{C_i}$, $A_{\mathscr{M}} = (\bigcup_{C_i \in C_{\mathscr{M}}} A_{C_i}) \bigcup A_{Tr}$ où A_{Tr} est l'ensemble d'arcs de transfert intra-module. Les nœuds se partagent en deux types : nœuds de transfert intra-module et nœuds de transfert inter-module. Pour le plus court chemin au sein d'un module, les nœuds de transfert inter-module ne sont pas concernés.

La Figure III.11 montre un graphe de module de trois composants $C = \{C_1, C_2, C_3\}$ où $V_1 = \{b_2, d_2, e\}$ avec e un nœud de transfert inter-module, $V_2 = \{a_1, b_1, c_1\}$, $V_3 = \{a_2, c_2, d_1\}$.

Pour calculer le plus court chemin au sein du module $\mathscr{M}$ entre une paire de nœuds $u - v$, nous étendons le graphe $G(\mathscr{M})$ pour avoir *le graphe étendu* $G(\mathscr{M}, u, v)$:

— en ajoutant u et v à $V_{\mathscr{M}}$;
— en ajoutant à $A_{\mathscr{M}}$ les arcs relatifs aux plus courts chemins au sein du composant qui relient u (ou v) aux nœuds de transfert intra-module du même composant que u (ou v).

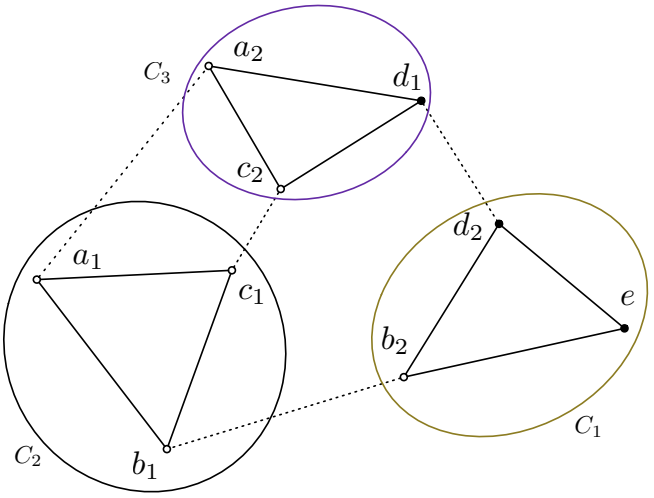


FIGURE III.11: Le graphe d'un module

La Figure III.12 visualise le graphe étendu du module $G(\mathscr{M}, u, v)$ pour le plus court chemin de $u - v$ au sein de $\mathscr{M}$.

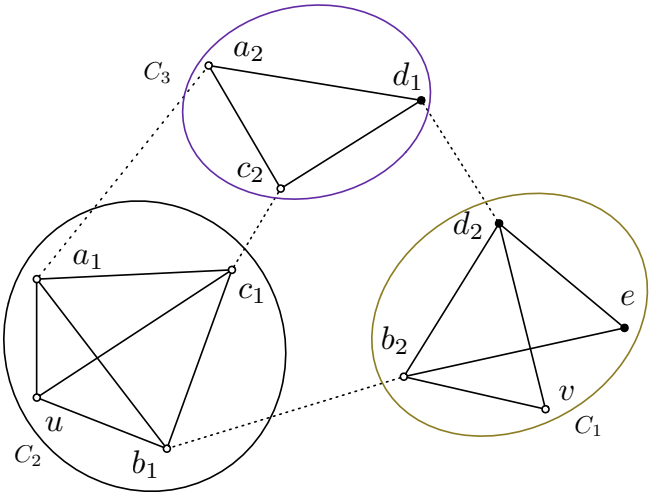


FIGURE III.12: Le graphe étendu d'un module $G(\mathscr{M}, u, v)$

Algorithm 6 Algorithme distribué du plus court chemin $u - v$ au sein d'un module $\mathscr{M}$ (**ADPC**)

1: Étape 1 : Construire le graphe étendu $G(\mathscr{M}, u, v)$ avec le module $\mathscr{M}$ ▷ Ajouter les nœuds u et v, puis ajouter les arcs reliant $u(v)$ avec les nœuds de transfert de son composant
2: Étape 2 : Calculer le plus court chemin $(u, n_1, n_2, \ldots, v)$ dans $G(\mathscr{M}, u, v)$ en utilisant l'algorithme du plus court chemin (Dijkstra)
3: Étape 3 : Pour chaque paire de nœuds consécutifs du chemin obtenu dans l'étape 2, retrouver son graphe uni-modal correspondant pour avoir les détails sur ce tronçon de route

Dans cet algorithme, la première étape vise à générer le graphe $G(\mathscr{M}, u, v)$ sur lequel l'algorithme du plus court chemin s'exécute. La deuxième étape consiste à calculer le plus court chemin $u - v$ avec l'algorithme Dijkstra. Dans la troisième étape, les informations sur les sections de chemin dans les sous-systèmes sont ajoutées après identification du graphe unimodal correspondant.

III.4.2.2 Le plus court chemin au sein d'une chaîne de modules

Le graphe d'une chaîne de modules est l'union des graphes de chaque module de la chaîne qui est complétée par l'ensemble d'arcs de transfert inter-module. La Figure III.13 montre le graphe de la chaîne de modules $\mathscr{M}(O) - \mathscr{M}(K) - \mathscr{M}(D)$. Les modules se constituant des composants sont interconnectés par les arcs de transfert inter-module.

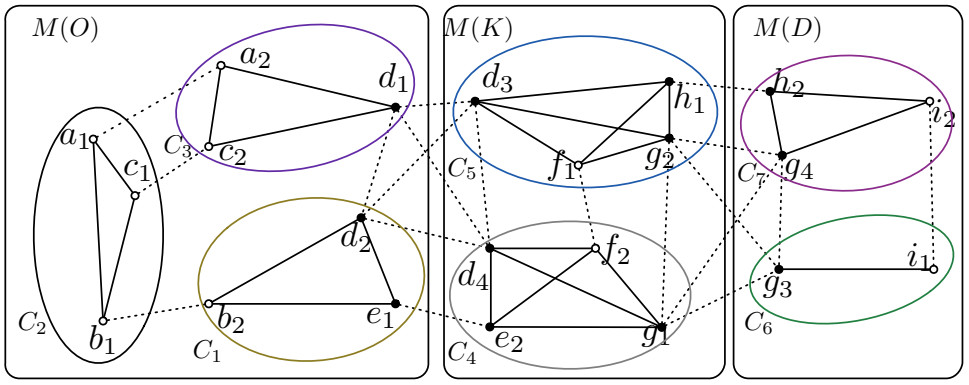


FIGURE III.13: Le graphe du chaîne de modules

Pour calculer le plus court chemin au sein de la chaîne de modules entre une paire de nœuds $u - v$, nous étendons le graphe de la chaîne de modules pour obtenir le graphe étendu $H_G(u, v)$:

— en ajoutant u et v dans l'ensemble de nœuds ;

— en ajoutant les arcs relatifs aux plus courts chemins au sein du composant qui relient $u(v)$ aux nœuds du même composant et les arcs relatifs aux plus courts chemins au sein du module qui relient $u(v)$ aux nœuds de transfert inter-module du même module à l'ensemble de arcs.

La Figure III.14 visualise le graphe étendu pour la paire de nœuds $u - v$.

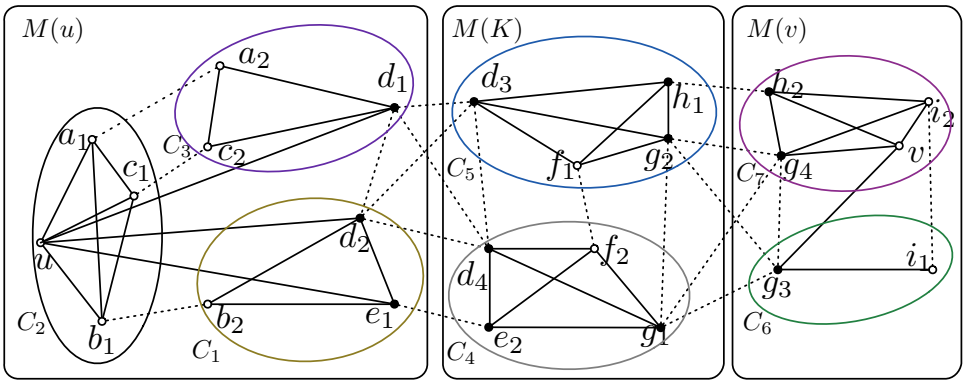


FIGURE III.14: Le graphe étendu du chaîne de modules

L'algorithme proposé pour calculer le plus court chemin $u - v$ au sein d'une chaîne de modules est l'Algorithme *ADPCC*.

Algorithm 7 Algorithme distribué du plus court chemin $u - v$ au sein d'une chaîne de modules (**ADPCC**)

1: Étape 1 : Construire le graphe étendu $H_G(u, v)$ avec la chaîne de modules
2: Étape 2 : Calculer le plus court chemin $u - v$ en passant par les nœuds inter-module dans $H_G(u, v)$ ▷ Algorithme 6 est utilisé pour le plus court chemin au sein d'un module
3: Étape 3 : Pour chaque paire de nœuds consécutifs du chemin obtenu dans l'étape 2, retrouver son graphe de module correspondant

Dans l'étape 2 de l'Algorithme *ADPCC*, on calcule le plus court chemin au sein du graphe dont l'ensemble de nœuds se réduit à celui des nœuds de transfert inter-modules complété par u et v. Les arcs intra-module sont étiquetés avec l'Algorithme *ADPC* lors de l'application de l'algorithme Dijkstra pour le plus court chemin.

En effet, le plus court chemin obtenu selon le critère choisi est une séquence de nœuds. Les contraintes horaires pour le départ ou l'arrivée ne sont pas encore prises en compte. Pourtant, cette séquence de nœuds permet de trouver les combinaisons de routes avec les services de transport assurés par les différents exploitants de transport sur les réseaux correspondants.

III.4.2.3 Les chemins attractifs

Pour aller d'un point d'origine au point de destination au sein d'un réseau multimodal, l'ensemble des itinéraires est très grand. Nous présumons donc que les usagers choisissent un chemin parmi les chemins les plus attractifs. Pour calculer les itinéraires attractifs, l'algorithme de k plus courts chemins est appliqué.

L'algorithme proposé dans le travail [Yen, 1971] peut être utilisé pour calculer les k plus courts chemins sans boucle avec un seul critère. Avec l'approche d'implémentation de ce dernier proposée par [Martins and Pascoal, 2003], nous trouvons les chemins les plus attractifs avec le sous-graphe trouvé.

III.5 Plus court chemin dans un graphe temps-étendu

III.5.1 Séquence multimodale

Définition. *Séquence multimodale* : Une série de déplacements multimodaux mis en ordre et interconnectés.

Une séquence multimodale constituée de $m+1$ activités est définie par une chaîne de $m+2$ nœuds. La résolution du problème de séquence multimodale consiste à trouver le choix de mode et l'heure de départ sur chaque nœud selon divers critères. Par exemple, le critère du plus faible temps de parcours exige la minimisation du temps passé pour les déplacements successifs. Pour les critères multiples, la résolution du problème vise à trouver des itinéraires optimaux multicritères.

Notons l'ensemble de nœuds dans la séquence multimodale par $\{n_i\}(i \leq m+1, n_0 = O, n_{m+1} = D)$ et l'ensemble de départs correspondants à n_i aux différents instants par $\{n_i^k\}$. Un itinéraire de cette séquence multimodale peut s'écrire alors sous forme $\{(O,n_1)(n_1,n_2)\dots(n_m,D)\}$. Cet itinéraire se constitue d'un ensemble de morceaux de route de réseaux de transports unimodaux. Si on prend en compte le transfert entre deux morceaux de route consécutifs, dont la connexion de transit peut être identifiée d'après une opération de consultation (ou *look-up*) dans le tableau de transfert, l'itinéraire P s'écrit alors $\{(O,n_1)\gamma_1(n_1',n_2)\gamma_2\dots\gamma_m(n_m',D)\}$ avec l'insertion des connexions de transit entre les morceaux de route.

Formellement, le problème de la séquence multimodale peut être exprimé ainsi :

Conditions :

— Un vecteur de nœuds $\langle n_i \rangle$;

— Un ensemble de départs sur chaque nœud aux différents instants $\{n_i^k\}$;

— Intervalle de temps pour le départ à O : $[T_1, T_2]$ et pour l'arrivée à D : $[T_3, T_4]$;

Objectif : trouver les séquences multimodales faisables telles que :

— les sections de la séquence multimodale sont réalisables ;

— les contraintes temporelles sont respectées ;

— le chemin représenté par chaque séquence multimodale est optimal.

Critères : Uni-critère ou multicritère.

La faisabilité d'une séquence multimodale se réfère à la *connectivité temporelle des déplacements*. C'est-à-dire, si on arrive à n_i à l'instant t et le temps de transfert de n_i à n'_i est $\Delta(n_i, n'_i)$, le départ de n'_i vers n_{i+1} est forcément après l'instant $t + \Delta(n_i, n'_i)$.

La section suivante vise à résoudre le problème décrit ci-dessus en faisant usage du graphe temps-étendu.

III.5.2 Stratégie de résolution

Un trajet entre le point d'origine et la destination passant par les nœuds intermédiaires ordonnés, peut être considéré comme une séquence multimodale. Trouver les meilleurs arrangements de déplacements de O à D sur l'ensemble de tronçons de routes pour l'itinéraire $P = \{(O, n_1)\gamma_1(n'_1, n_2)\gamma_2 \ldots \gamma_m(n'_m, D)\}$ (cf. Figure III.15), revient à résoudre le problème de la séquence multimodale.

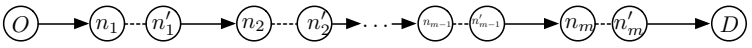


FIGURE III.15: Une séquence multimodale avec transits

Pour résoudre ce problème posé, nous présentons une méthode avec le graphe temps-étendu en prenant en compte le temps de correspondance. Conformément à ce que nous avons présenté dans la section II.2.3.3, un nouveau nœud est ajouté au graphe temps-étendu pour chaque départ sur l'ensemble des nœuds de la séquence. Sans perte de généralité, la propriété *non-FIFO* du graphe est assurée. De plus, l'attente dans les nœuds pour prendre les véhicules suivants est autorisée.

III.5.2.1 *"Diviser pour régner"*

Comme pour les autres problèmes du plus court chemin, l'espace de recherche pour ce problème de séquence multimodale est composé de tous les itinéraires qui commencent avec les nœuds source, à savoir l'ensemble de départs conformes aux contraintes temporelles. Étant donné les nœuds de la séquence et les contraintes temporelles, la multiplicité

des départs sur les nœuds intermédiaires entraîne une explosion combinatoire de combinaisons de nœuds, parmi lesquelles nous allons chercher les solutions optimales selon les critères et les contraintes.

En effet, pour résoudre ce problème de séquence multimodale et réduire l'espace de recherche, il existe deux stratégies pour le diviser en certain nombre de sous-problèmes. L'idée de la division revient à découper le problème ciblé en sous-problème plus simple de même type, puis résoudre ces sous-problèmes et combiner les solutions des sous-problèmes pour se ramener à une solution du problème initial.

Pour la première stratégie, chaque sous-problème vise à obtenir la meilleur connexion entre deux nœuds consécutifs dans la séquence. Ensuite, les solutions optimales trouvées de chaque sous-problème sont fusionnées pour former la solution du problème posé.

Ce schéma de division est une méthode naïve car les sous-problèmes résultants de la division ne sont pas indépendants et la *somme* d'optimalité de sous-problèmes ne garantie pas l'optimalité de la solution au problème posé.

Pour la deuxième stratégie, nous prenons le problème du point d'origine O à un nœud n_i de la séquence comme un sous-problème. Nous rappelons que ces sous-problèmes sont indépendants. Cependant, un sous-problème peut être résolu à partir des solutions d'un autre sous-problème qui lui précède dans l'ordre de la séquence de nœuds.

Pour ce schéma de division, la séquence de $m+2$ nœuds entraîne $m+1$ sous-problèmes dont le premier p_1 est de trouver l'ensemble de sous-itinéraires *adéquats* de l'origine O au premier nœud de transfert n_1. Les sous-problèmes sont résolus d'une façon récursive, c'est-à-dire, la résolution du sous-problème p_{i+1} peut être basée sur l'ensemble de solutions du sous-problème $p_i (i < m+2)$. En effet, l'ensemble de solutions du $p_i (i < m+2)$ constitue les combinaisons *adéquates* de O à n_i. Et ainsi de suite, les solutions du problème de la séquence multimodale seront parmi l'ensemble de solutions du dernier sous-problème. Afin de limiter l'espace de recherche (qui dépend du nombre de solutions pour chaque sous-problème), l'ensemble de solutions du sous-problème réduit à celui qui n'est composé que des itinéraires "suspendus" *adéquats*. Une solution *adéquate* du sous-problème se réfère à un sous-itinéraire qui est *utile* pour résoudre le prochain sous-problème. Dans un réseau dynamique, la définition d'une solution *adéquate* peut varier selon différents critères d'optimisation. Citons le temps de parcours et le coût du trajet comme exemple. Si on veut trouver le chemin le plus rapide, le sous-itinéraire ayant le moins de temps de parcours en comparaison avec ceux ayant le même temps d'arrivée, est une solution *adéquate*. En revanche, si on veut calculer le chemin le moins cher, une solution *adéquate* est soit, celle qui est meilleure que les sous-itinéraires qui assurent les arrivées plus tôt qu'elle même, soit, celle ayant l'arrivée le plus tôt possible. Si on

cherche les solutions optimales au sens de Pareto avec ces deux critères, une solution *adéquate* est soit, celle qui n'est pas dominée par celles ayant les arrivées non plus tard qu'elle-même, soit, celle qui est Pareto optimale.

III.5.3 Recherche bidirectionnelle dans un graphe temps-étendu

III.5.3.1 Ajout d'un nœud remplaçant

Avec un graphe temps-étendu, afin d'éviter le problème de type *plusieurs-à-plusieurs* qui est beaucoup plus compliqué que celui de type *un-à-plusieurs*, un nœud remplaçant M_f est ajouté dans le graphe (cf. Algorithme *ANRF*). En effet, pour le sous-problème p_{i+1}, M_f est relié avec n_i par un arc dont le poids est le coût du chemin $O - n_i$. Ainsi, le sous-problème traité p_{i+1} devient un problème de type *un-à-plusieurs*. Par conséquent, le dernier sous-problème pour une recherche avec l'étiquetage vers l'avant est de trouver le plus court chemin $M_f - D$ de type *un-à-plusieurs*.

Algorithm 8 Ajout d'un nœud remplaçant pour l'étiquetage vers l'avant (**ANRF**)

1: $G = (V, A) \leftarrow$ graphe temps-étendu ▷ graphe pour le sous-problème courant
2: $M_f \leftarrow$ Nœud remplaçant
3: $E \leftarrow$ Ensemble de sous-itinéraires du sous-problème précédent
4: $V \leftarrow V \cup \{M_f\}$ ▷ Ajouter le nœud
5: **Pour tout** $p \in E$
6: $n \leftarrow$ dernier nœud de p
7: $\mathbf{v}(p) \leftarrow$vecteur de coût de p
8: $A \leftarrow A \cup (M_f, n, \mathbf{v}(p))$ ▷ Ajouter les arcs reliant n et M_f
9: **Fin Pour**
10: **Sortie :** G

Pour la recherche avec l'étiquetage vers l'avant, la méthode d'ajout d'un nœud remplaçant M_f est illustrée par la Figure III.16.

L'application de la recherche avec l'étiquetage vers l'avant sur la totalité du problème est visualisée sur la Figure III.17.

Pour la recherche avec l'étiquetage vers l'arrière, la même stratégie d'ajout d'un nœud remplaçant M_b est appliquée. M_b est ajouté dans le graphe (cf. Algorithme *ANRB*). La méthode d'ajout d'un nœud remplaçant est illustrée par la Figure III.18. En effet, pour le sous-problème $n_{j-1} - D$, ce nœud ajouté est relié avec n_j du sous-problème précédent par un arc dont le poids est le coût du chemin $n_j - D$. Le nouveau sous-problème à traiter $n_{j-1} - D$ est un problème de type *plusieurs-à-un* avec l'étiquetage vers l'arrière. Ainsi de suite, le dernier sous-problème est de trouver le plus court chemin $O - M_b$.

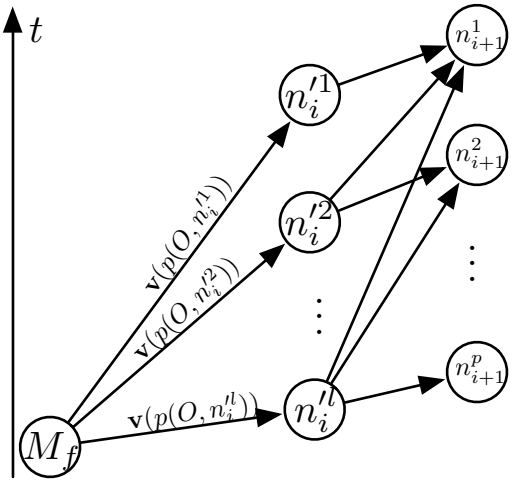


FIGURE III.16: La recherche avec l'étiquetage vers avant dans un graphe temps-étendu

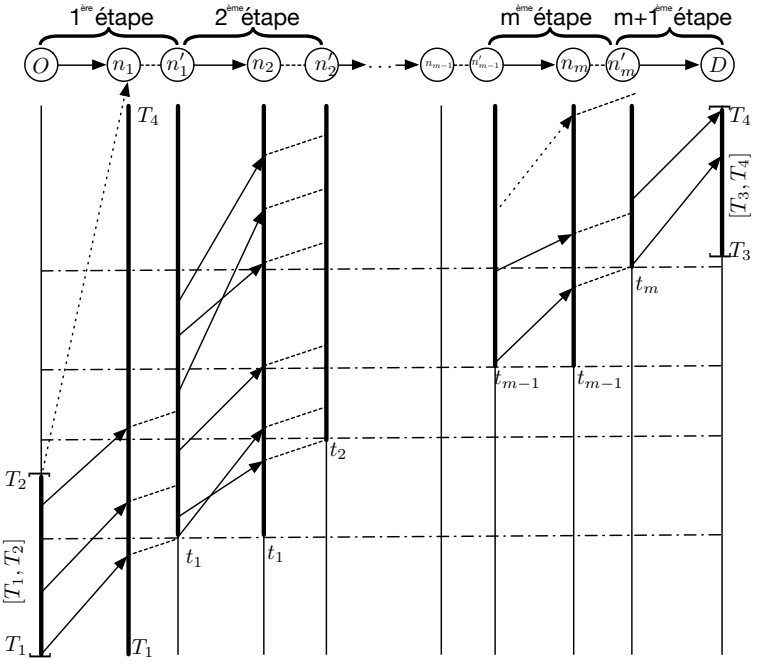


FIGURE III.17: La recherche avec l'étiquetage vers l'avant avec le raffinement de fenêtre de temps

Algorithm 9 Ajout d'un nœud remplaçant pour l'étiquetage vers l'arrière (**ANRB**)

1: $G = (V, A) \leftarrow$ graphe temps-étendu ▷ graphe pour le sous-problème courant
2: $M_b \leftarrow$ Nœud remplaçant
3: $E \leftarrow$ Ensemble de sous-itinéraires du sous-problème précédent
4: $V \leftarrow V \cup \{M_b\}$ ▷ Ajouter le nœud
5: **Pour tout** $p \in E$
6: $n \leftarrow$ premier nœud de p
7: $\mathbf{v}(p) \leftarrow$vecteur de coût de p
8: $A \leftarrow A \cup (M_b, n, \mathbf{v}(p))$ ▷ Ajouter les arcs reliant n et M_b
9: **Fin Pour**
10: **Sortie :** G

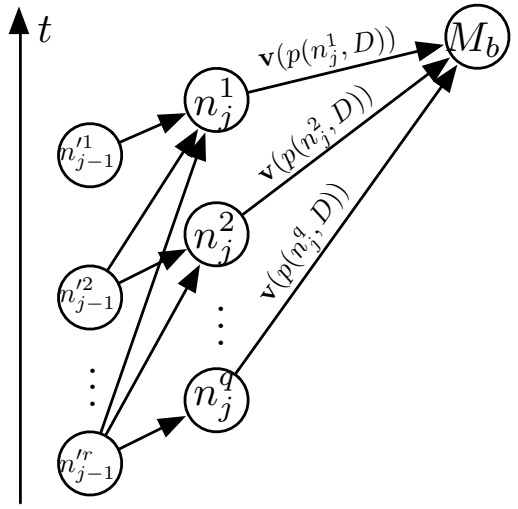


FIGURE III.18: La recherche avec l'étiquetage vers l'arrière dans un graphe temps-étendu

L'application de la recherche avec l'étiquetage vers l'arrière sur la totalité du problème est visualisée sur la Figure III.19.

Au cours des étapes, certains sous-itinéraires suspendus n'aboutissent pas à une solution faisable à cause de la violation des contraintes temporelles. Nous pouvons réduire le nombre de nœuds explorés en limitant le nombre des sous-itinéraires suspendus.

III.5.3.2 Recherche bidirectionnelle dans un graphe temps-étendu

Avec l'ajout de nœuds remplaçants, nous pensons à la mise en place de l'heuristique de recherche bidirectionnelle en vue de réduire l'espace de recherche. Lors de l'exécution de l'algorithme, la recherche avec l'étiquetage vers l'avant et avec l'étiquetage vers l'arrière, les sous-problèmes sont devenus de type *un-à-plusieurs* (avec l'étiquetage vers l'avant) et de type *plusieurs-à-un* (avec l'étiquetage vers l'arrière). La recherche s'arrête quand

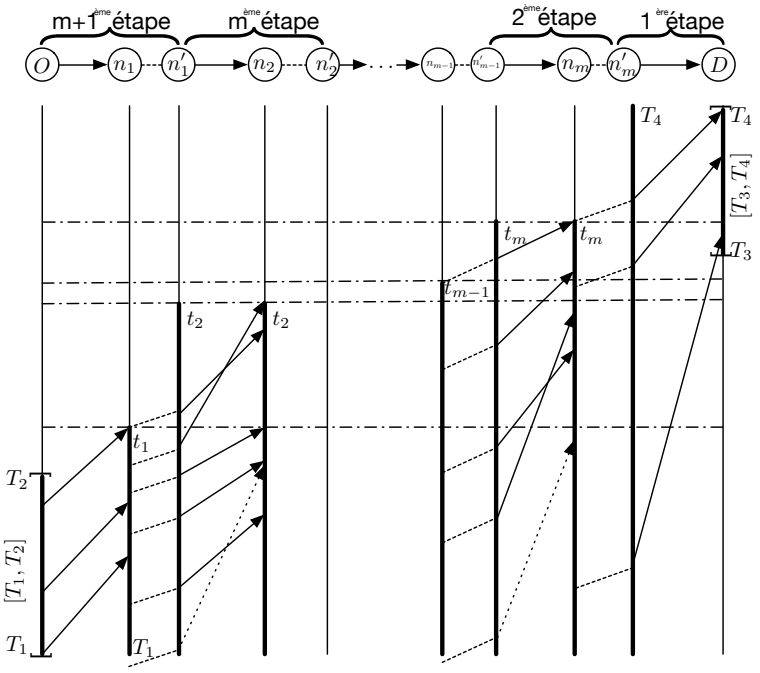


FIGURE III.19: La recherche avec l'étiquetage vers l'arrière avec le raffinement de fenêtre de temps

les deux frontières de l'espace de recherche se croisent. Par conséquent, le dernier sous-problème est de type *un-à-un*.

La recherche bidirectionnelle aide à raffiner les fenêtres de temps pour chaque étape vers l'avant et vers l'arrière. Les contraintes entraînées sur les nœuds intermédiaires sont utilisées pour la résolution du sous-problème suivant. L'approche du raffinement des fenêtres horaires est illustrée sur la Figure III.20.

Selon la stratégie de division avec la recherche bidirectionnelle, le processus de résolution est décrit pour le problème de la séquence multimodale dans la section III.5.1. La recherche avec l'étiquetage vers l'avant et celle vers l'arrière s'alternent. Le premier sous-problème à résoudre avec l'étiquetage vers l'avant est celui du point de départ O au point n_1 avec $[T_1, T_2]$ comme fenêtre de temps de départ et $[T_1, T_4]$ comme fenêtre de temps d'arrivée. L'ensemble de solutions d'un sous-problème $O-n_i$ avec l'étiquetage vers l'avant est noté par S_f qui sera utilisée pour étiqueter les arcs reliant le nœud remplaçant M_f et n_i pour la résolution du prochain sous-problème avec l'étiquetage vers l'avant. L'heure d'arrivée au plus tôt à n'_i est notée par t_i qui sera l'heure de départ au plus tôt pour la prochaine recherche avec l'étiquetage vers l'arrière. Le deuxième sous-problème est celui de n_m à D à résoudre avec l'étiquetage vers l'arrière dont $[t_1, T_4]$ est la fenêtre de temps de départ et $[T_3, T_4]$ est la fenêtre de temps d'arrivée. L'ensemble de solutions d'un sous-problème $n_j - D$ avec l'étiquetage vers l'arrière est noté par S_b qui sera utilisé pour étiqueter les arcs reliant le nœud remplaçant n_j et M_b pour la résolution du

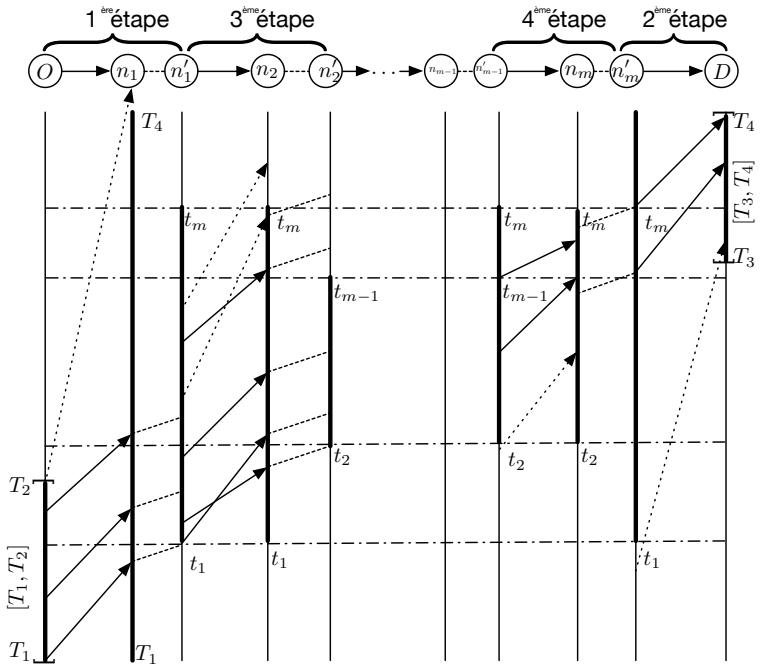


FIGURE III.20: La recherche bidirectionnelle avec le raffinement de fenêtre de temps

prochain sous-problème avec l'étiquetage vers l'arrière. L'heure de départ au plus tard à n_j est notée par t_j qui sera l'heure d'arrivée au plus tard pour la prochaine recherche avec l'étiquetage vers l'avant. À la fin de ce processus, les deux frontières S_f et S_b se rejoignent pour résoudre le problème $M_f - M_b$ de type *un-à-un*.

Cet algorithme basé sur Dijkstra bidirectionnel est décrit par l'Algorithme *AIPB*.

III.5.3.3 Raffinements préliminaires des fenêtres horaires

Avant de mettre en place la recherche bidirectionnelle, une technique servant à raffiner les fenêtres temporelles appliquée préalablement permet de réduire l'espace de recherche.

À partir des contraintes temporelles exprimées par l'usager pour le départ et l'arrivée, les plages horaires sur chaque nœud intermédiaire peuvent être identifiées afin de limiter le nombre de sous-itinéraires *suspendus*. Ces plages temporelles sont trouvées suite à la résolution de deux problèmes, à savoir : le problème d'arrivée au plus tôt et le problème de départ au plus tard.

Dans l'objectif de résoudre ces deux problèmes, nous procédons de la manière suivante :

— *problème d'arrivée au plus tôt* : recherche de l'heure d'arrivée au plus tôt t_a dans $[T_3, T_4]$ pour tout départ de O dans $[T_1, T_2]$(étiquetage vers l'avant), en effet, les heures d'arrivée au plus tôt en tous les nœuds intermédiaires pour tout départ de O dans $[T_1, T_2]$ sont également obtenues ;

Algorithm 10 Algorithme pour calculer l'ensemble des itinéraires optimaux au sens de Pareto avec la recherche bidirectionnelle (**AIPB**)

$P \leftarrow \{(O, n_1)\gamma_1(n'_1, n_2)\gamma_2 \dots \gamma_m(n'_m, D)\}$ un itinéraire avec les nœuds intermédiaires
Fenêtre de temps de départ$\leftarrow [T_1, T_2]$
Fenêtre de temps d'arrivée$\leftarrow [T_3, T_4]$
$i \leftarrow 1, j \leftarrow 1$
$S_f \leftarrow \emptyset$ ▷ Ensemble de solutions sous-itinéraire avec l'étiquetage vers l'avant
$S_b \leftarrow \emptyset$ ▷ Ensemble de solutions sous-itinéraire avavec l'étiquetage vers l'arrière
$t_0 \leftarrow T_1, t_{m+1} \leftarrow T_4$
Faire Tant Que $(i < m + 2)\{$
  Faire Tant Que $(\lceil \frac{i}{2} \rceil < \lceil \frac{m+3}{2} \rceil)$ $\{$ ▷ Recherche avec l'étiquetage vers l'avant
    $k \leftarrow \lceil \frac{i}{2} \rceil$
    Sous-problème $sp_i : \{(O, n_1) \dots (n'_{k-1}, n_k)\}$
    Replacer $\{(O, n_1) \dots (n'_{k-2}, n_{k-1})\}$ par nœud M_f et construire le graphe temps-étendu basé sur S_f
    Fenêtre de temps d'arrivée $\leftarrow [t_k, t_{m+1-k}]$, Fenêtre de temps de départ $\leftarrow [t_k, t_{m+1-k}]$ ▷ $i = 1$, Fenêtre de temps de départ $\leftarrow [T_1, T_2]$
    Obtenir l'ensemble de solutions non-dominées S_f à sp_i
    MAJ l'ensemble de solutions avec le coût de transit S_f
    $t_k \leftarrow$ Le temps d'arrivée au plus tôt
    $i \leftarrow i + 1$ $\}$
  Fin Tant Que
  Faire Tant Que $(\lceil \frac{i}{2} \rceil < \lfloor \frac{m+3}{2} \rfloor)$ ▷ Recherche avec l'étiquetage vers l'arrière
    $\{j \leftarrow \lceil \frac{i}{2} \rceil$
    MAJ l'ensemble de solutions prenant en compte le coût de transfert S_b
    Sous-problème $sp_i : \{(n'_{m+1-j}, n_{m+2-j}) \dots (n'_m, D)\}$
    Remplacer $\{(n'_{m+2-j}, n_{m+3-j}) \dots (n'_m, D)\}$ par le nœud remplaçant M_b et construire le graphe temps-étendu basé sur S_f
    Fenêtre de temps pour le départ $\leftarrow [t_j, t_{m+2-j}]$, Fenêtre de temps pour l'arrivée $\leftarrow [t_j, t_{m+2-j}]$ ▷ $i = 1$, Fenêtre de temps pour l'arrivée $\leftarrow [T_3, T_4]$
    L'ensemble de solutions faisables et non-dominées de S_b à sp_i
    $t_{m+1-j} \leftarrow$ Temps du dernier départ
    $i \leftarrow i + 1\}$
  Fin Tant Que
$\}$
Fin Tant Que
Rejoindre S_f et S_b pour un problème de type *un-à-un*
L'ensemble d'itinéraires non-dominés S
Return : S

— *problème de départ au plus tard* : recherche de l'heure de départ au plus tard t_d dans $[T_1, T_2]$ de O pour arriver en D dans $[T_3, T_4]$(étiquetage vers l'arrière), en effet, les heures de départ au plus tard en tous les nœuds intermédiaires pour tout arrivée en D dans $[T_3, T_4]$ sont également obtenues.

La recherche bidirectionnelle avec raffinement préalable des fenêtres horaires est illustrée dans la Figure III.21.

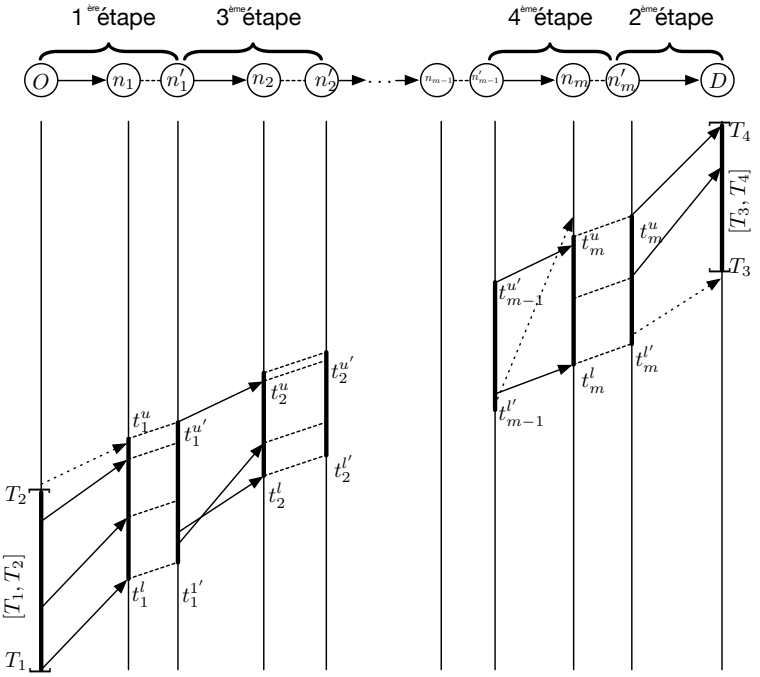


FIGURE III.21: La recherche bidirectionnelle avec raffinement préalable des fenêtres de temps

En outre, cette mise en place du raffinement des fenêtres de temps permet de

— réduire l'espace de recherche avant d'effectuer la recherche bidirectionnelle ;

— identifier la flotte de véhicules sur chaque morceau de route compatibles avec les conditions de temps pour le départ et l'arrivée.

III.6 Problème d'affectation

Dans cette section, nous allons formuler le problème d'affectation. À titre d'exemple, un itinéraire ayant deux transferts $P = \{(O, n_1)(n'_1, n_2)(n'_2, D)\}$ peut être schématisé par la Figure III.22. Ce trajet allant de O à D possède deux transferts sur n_1 et n_2 avec (n_1, n'_1) et (n_2, n'_2) les deux liens de transit. Durant le temps de transfert entre modes, un délai minimum est nécessaire pour le transit et un temps d'attente peut être présent. Il convient de noter que ce temps de transfert entre modes est considéré comme une partie du temps de parcours du trajet.

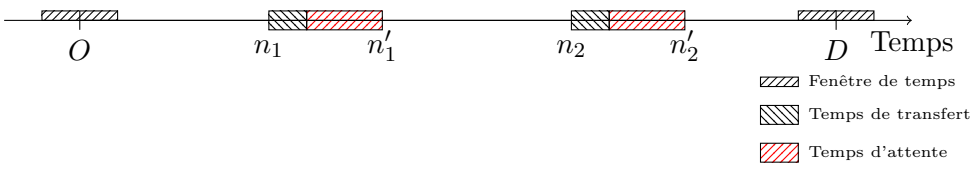


FIGURE III.22: Illustration d'un trajet avec deux transferts

À l'aide du raffinement des fenêtres horaires pour chaque nœud de la séquence multi-modale (cf. section III.5.3.3), une flotte de véhicules conformes aux conditions de temps peut être identifiée sur chaque section de route. Nous prenons une section (a, b) comme exemple. Comme le montre la Figure III.23, les fenêtres horaires correspondantes aux nœuds a et b sont visualisées. $[t_1^a, t_2^a]$ et $[t_1^b, t_2^b]$ sont les deux plages horaires pour a et b.

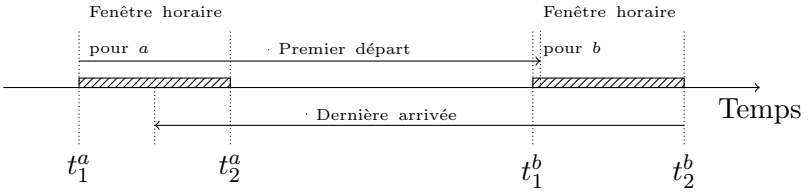


FIGURE III.23: Une section de route (a, b) avec ses fenêtres de temps

III.6.1 Formulation du problème d'affectation

Nous nous concentrons par la suite sur une seule section de route où se présentent des demandes (des voyageurs) et des offres (des véhicules). Dans cette étape, il s'agit d'un problème d'affectation ou de "matching" entre les demandes et les offres. Reprenons la section de route (a, b) allant de a à b. Considérons qu'il s'agit d'un ensemble de n requêtes $I = \{I_1, I_2, \ldots, I_n\}$ et un ensemble de m véhicules $V = \{V_1, V_2, \ldots, V_m\}$. Les véhicules se distinguent en trois catégories en fonction de leurs capacités d'accueillir des voyageurs : véhicule de transport en commun à grande capacité (i.e. train), véhicule de covoiturage à capacité limitée et véhicule libre-service pour un seul voyageur à chaque fois.

On désigne par s le nombre de requêtes réussies qui sont attribuées sur cette section de véhicules.

Nous prenons en compte trois critères : le temps, le coût et le nombre de requêtes réussies. Les poids définissant les importances relatives des différents critères sont ω_1, ω_2 et ω_3. Au niveau de la fonction objectif, se présentent trois composants, à savoir : le temps total, le coût total et le nombre de requêtes bien affectées.

Nous avons alors un problème de programmation linéaire en nombres entiers (Integer Programming), le récapitulatif des notations est donné avec le Tableau III.2.

Tableau III.2: Notations mathématiques de la formulation du problème

Notation	**Définition**
(a, b)	Section de route allant du nœud a au nœud b
I	Ensemble des requêtes sur la section (a, b)
N	Nombre de requêtes sur la section (a, b)
V	Ensemble des véhicules sur la section (a, b)
C^j	Capacité de véhicule V_j
$\omega_1, \omega_2, \omega_3$	Poids des différents composants
x_i^j	Variable de décision d'affectation entre I_i et V_j
t_j	Temps nécessaire pour le tronçon (a, b) avec le véhicule V_j
c_j	Coût pour le tronçon (a, b) avec le véhicule V_j
s	Nombre de requêtes affectées avec réussite
T_d^j	Instant où le véhicule V_j est sur le nœud d
l^j	Nombre de passagers sur véhicule V_j au début de l'affectation
L^j	Nombre de passagers sur véhicule V_j à l'issue de l'affectation
$[\alpha_d, \beta_d]$	Fenêtre horaire pour le nœud d

L^j signifie le nombre de passagers après l'affectation sur le véhicule V_j dont la capacité est C^j, l^j signifie le nombre de passagers avant le processus d'affectation.

Le problème peut être formulé ainsi :

La fonction multiobjectif à minimiser est fournie par l'équation III.2.

$$\min \quad \omega_1 \sum_{V_j \in V} \sum_{I_i \in I} x_i^j t_j + \omega_2 \sum_{V_j \in V} \sum_{I_i \in I} x_i^j c_j + \omega_3 (N - s) \tag{III.2}$$

La fonction objectif III.2 peut s'écrire également comme :

$$\min \quad \omega_1 \textstyle\sum_{V_j \in V} \sum_{I_i \in I} x_i^j t_j + \omega_2 \sum_{V_j \in V} \sum_{I_i \in I} x_i^j c_j + \omega_3 (N - \sum_{V_j \in V} \sum_{I_i \in I} x_i^j)$$

sous les contraintes :

$$\sum_{V_j \in V} x_i^j \leq 1, \forall I_i \in I \tag{III.3}$$

$$x_i^j (T_a^j + t_j - T_b^j) \leq 0, \forall I_i \in I, V_j \in V \tag{III.4}$$

$$\alpha_d \leq T_d^j \leq \beta_d, \forall V_j \in V, d \in \{a, b\} \tag{III.5}$$

$$l^j \leq L^j \leq C^j, \forall V_j \in V \tag{III.6}$$

L'équation III.6 peut être écrite également comme

$$0 \leq \sum_{I_i \in I} x_i^j \leq C^j - l^j, \forall V_j \in V$$

$$x_i^j \in \{0,1\}, \forall I_i \in I, V_j \in V \tag{III.7}$$

La contrainte (III.3) assure qu'une requête ne puisse être affectée qu'une seule fois. La contrainte (III.4) assure que la condition du temps minimum de voyage sur cette section de route soit respectée. La contrainte (III.5) assure que les contraintes temporelles soient respectées. La contrainte (III.6) assure que la capacité de véhicule soit respectée.

Dans la section suivante, nous proposons une approche basée sur les algorithmes génétiques qui permet notamment d'éviter les optima locaux pour résoudre le problème d'affectation.

III.7 Approche évolutionniste pour résoudre le problème d'affectation

Dans cette section, nous essayons de mener la résolution du problème d'affectation qui est formulé dans la section précédente, à l'aide de la métaheuristique ([Wang et al., 2014b]).

En effet, le problème d'affectation est un problème d'allocation de ressources. Autrement dit, s'il existe R_e ressources disponibles et M demandes avec une fonction coût (qui dépend de la ressource utilisée) pour chaque demande, il faut choisir un sous-ensemble de ressources, R, afin de maximiser (ou minimiser) la somme des fonctions coût pour chaque demande. C'est un problème combinatoire. En effet, il faut sélectionner R ($0 < R < R_e$) de manière que la somme des fonctions coût soit maximisée (ou minimisée) avec un espace de recherche extrêmement grand. À titre d'exemple, sélectionner 10 parmi 100 compte plus de 1.7×10^{14} possibilités qui ne peuvent être toutes examinées afin de trouver les solutions optimales.

Nous pouvons dire que ce problème d'affectation est un problème d'optimisation combinatoire multi-objectif. Les métaheuristiques sont souvent utilisées et sont réputées efficaces pour résoudre ce type de problème [Ishibuchi and Murata, 1998] [Ulungu and Teghem, 1994]. Une approche évolutionniste va être employée pour ce problème d'optimisation. Concrètement, nous proposons un algorithme génétique pour résoudre ce problème d'affectation.

Dans cette section, tout d'abord, nous allons introduire la méthode de codage pour représenter la solution d'affectation. Ensuite, la notion de schéma utilisée favorisant la convergence rapide de solution est présentée. Par la suite, les opérateurs de croisement, mutation et correction seront présentés. Dans l'étape qui suit, seront détaillés les critères choisis, ainsi que la fonction objectif pour ce problème d'optimisation. En fin, des exemples sont fournis afin d'illustrer cette méthode.

III.7.1 Représentation des solutions (codage)

Avant d'utiliser les algorithmes génétiques, une question essentielle se pose : comment représenter les solutions, à savoir les résultats d'affectation pour notre problème ? Ce problème d'encodage est le premier pas avec les algorithmes génétiques. Cette section est dédiée à la représentation génétique adaptée à notre problématique.

III.7.1.1 Description générale

Étant donné que c'est un problème d'affectation ou de "matching", nous empruntons le codage binaire qui s'accorde bien avec la description de l'affectation, présentée dans la section III.6 pour la formuler le problème.

Afin de représenter l'affectation des offres aux demandes sur une section de route, nous employons la forme matrice dont les deux dimensions sont les suivantes : les lignes correspondent aux demandes (passagers) sur la route, et les colonnes sont associées aux offres (véhicules) qui assurent le service de transport sur la même section de route. Les éléments de cette matrice d'affectation représentent les résultats de l'affectation du passager de la ligne au véhicule de la colonne définie par une variable binaire, à savoir "0" ou "1".

Sur la section de route (a, b) pendant la fenêtre de temps $[\alpha, \beta]$, la matrice d'affectation $\mathbf{A}$ est définie par l'Équation (III.8) :

$$A_{(a,b)}^{[\alpha,\beta]}(i,j) = \begin{cases} 1, & \text{si } I_i \text{ est affectée à } V_j, \\ *, & \text{si } I_i \text{ peut être affectée à } V_j, \\ 0, & \text{si } I_i \text{ n'est pas affectée à } V_j \end{cases} \tag{III.8}$$

où $I = \{I_i\}$ est l'ensemble des requêtes (ou passagers), $V = \{V_j\}$ est l'ensemble des véhicules, l'élément $A_{i,j}$ est le résultat d'affectation du passager I_i au véhicule V_j. Les éléments de la matrice peuvent ainsi avoir une des valeurs {0,*,1} : "1" signifie que le

passager I_i est affecté au véhicule V_j et "0" signifie que le passager I_i n'est pas affecté au véhicule V_j.

III.7.1.2 Les contraintes du problème

Comme pour la formulation du problème avec programmation linéaire en nombres entiers (cf. section III.6), plusieurs contraintes existent également avec l'approche évolutionniste. Il faut que ces contraintes soient respectées au cours de l'évolution des générations.

Nous prenons un exemple de matrice d'affectation **A** sur la route (a, b) pendant la fenêtre de temps $[\alpha, \beta]$ (III.9).

$$A_{n,m} = \begin{array}{c} \\ I_1 \\ I_2 \\ \vdots \\ I_n \end{array} \begin{array}{c} \begin{array}{cccc} V_1 & V_2 & \cdots & V_m \end{array} \\ \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,m} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,m} \end{pmatrix} \end{array} \tag{III.9}$$

où

- $I = \{I_1, I_2, \ldots, I_n\}$ est l'ensemble des requêtes correspondantes ;
- $V = \{V_1, V_2, \ldots, V_m\}$ est l'ensemble des véhicules associés.

Si les offres existent sur le tronçon de route (a, b), il existe les paramètres spécifiques pour chaque véhicule desservant le tronçon en question :

- $N^j_{(a,b)}$: c'est la quantité des ressources disponibles de transport sur (a, b). Ici nous distinguons trois cas : (1) le nombre de places disponibles dans le véhicule V_j quand il s'agit de covoiturage ; (2) le nombre de véhicules disponibles du type V_j sur le parking au nœuds a quand il s'agit d'autopartage ou de vélo partage ; (3) le nombre de places disponibles qui est considérée comme important dans le véhicule V_j pour le transport en commun.
- $[d_j, a_j]$: ceci indique, pour tout véhicule V_j desservant le tronçon de route (a, b), le temps de départ d_j à l'extrémité a et le temps d'arrivée a_j à l'extrémité b.

Pour la population initiale, chaque chromosome est généré aléatoirement en respectant les contraintes :

- La contrainte où chaque requête ne peut être attribuée qu'une seule fois est représentée comme suit :

$$\sum_{j=1}^{m} a_{k,j} \leq 1, \forall k \in [1, n] \tag{III.10}$$

— La contrainte où la quantité des véhicules attribués aux passagers ne peut pas être supérieur à celui d'offres est formulée comme suit :

$$\sum_{i=1}^{n} a_{i,k} \leq N^{k}_{(a,b)}, \forall k \in [1, m], \tag{III.11}$$

où $N^{k}_{(a,b)}$ est la quantité de l'offre disponible pour le véhicule V_k.

Au cours des générations, les processus de croisement et mutation fournissent des nouvelles populations dont les chromosomes doivent respecter les contraintes (III.12) et (III.13).

$$\sum_{j=1}^{m} a_{k,j} = 1, \forall k \in [1, n] \tag{III.12}$$

$$\sum_{i=1}^{n} a_{i,k} \leq N^{k}_{(a,b)}, \forall k \in [1, m] \tag{III.13}$$

III.7.1.3 Génération des chromosomes

L'objectif de la théorie des schémas [Holand, 1975], est que les algorithmes génétiques soient plus efficaces et plus rapides, en construisant la solution en favorisant la reproduction des individus respectant les modèles générés par les bons schémas, au lieu de l'ensemble des chromosomes. Dans le cas des problèmes d'affectation, les préférences des requêtes et les contraintes temporelles sont utilisées pour la mise en œuvre de cette technique.

Les requêtes d'itinéraires sont saisies avec les préférences. Nous citons à titre d'exemple : quelqu'un ne possède pas le permis de conduire va sans doute exclure le service d'autopartage, ou encore quelqu'un avec une valise de grand volume souhaite trouver une offre de covoiturage qui accepte des gros bagages.

Chaque usager exprime une requête en indiquant un intervalle horaire dans lequel il souhaite être servi. Cette condition permet de déterminer les véhicules qui sont compatibles sur chaque tronçon de route tout au long du trajet.

L'idée est de créer, à partir des conditions existantes, un schéma d'affectation qui sert à contrôler l'algorithme génétique. Ce schéma va donc représenter des contraintes à respecter par les chromosomes générés (cf. Algorithme *AGSA*)

Algorithm 11 Algorithme de la génération du schéma d'affectation (**AGSA**)

1: $I \leftarrow \{I_1, I_2, \ldots, I_n\}$ ▷ L'ensemble de requêtes concernées
2: $V \leftarrow \{V_1, V_2, \ldots, V_m\}$ ▷ L'ensemble de véhicules concernés
3: $\mathbf{S} \leftarrow * \cdot I_{n\times m}$ ▷ Initialisation
4: **Pour** $1 \leq i \leq n$
5: **Pour** $1 \leq j \leq m$
6: **Si** I_i exclut V_j **Alors** $S_{ij} = 0$; ▷ Exclusion comme préférence ou temps non compatible
7: **Fin si**
8: **Si** I_i ne prend que V_j **Alors** $S_{ij} = 1$; ▷ Choix obligatoire comme préférence
9: **Fin si**
10: **Fin pour**
11: **Fin pour**
12: **Sortie :** Le schéma d'affectation **S**

Pour le schéma, nous définissons les éléments suivants :

- La valeur "0" signifie que l'affectation du passager au véhicule est *interdite* ;
- La valeur "1" signifie que l'affectation du passager au véhicule est *obligatoire*, dans ce cas, toutes les valeurs de la même ligne sont forcément égales à "0" ;
- Le symbole "*" signifie que l'affectation est *possible*, autrement dit, plusieurs "1" ne sont pas possibles en même temps dans la même ligne.

Exemple : Supposons que sur un tronçons de route (u, v), pour la fenêtre de temps $[8h, 11h]$, il existe 4 types d'offres et 4 requêtes.

Du côté des offres, nous avons :

- Train : $V1_{u_1,v_1}^{[8h30,9h30]}$;
- Covoiturage : $V2_{u_2,v_2}^{[8h40,10h30]}(2,3)$ n'accepte pas de gros bagage à bord ; $(2,3)$ signifie qu'il y a au total 3 places dont 2 sont disponibles pour $V2$;
- Covoiturage : $V3_{u_2,v_2}^{[9h20,11h00]}(1,3)$ accueille des participants qui peuvent conduire ; $(1,3)$ signifie qu'il y a au totale 3 places dont 1 est disponible pour $V3$;
- Autopartage : $V4_{u_3,v_3}^{[t_0,t_0+2h]}(3)$. (3) signifie que 3 véhicules d'autopartage sont disponibles sur le parking près de u.

Du côté des demandes, nous avons :

- I_1 : Passager 1 ne prend que le transport en commun ;
- I_2 : Passager 2 n'a pas de permis de conduire, cependant, il accepte tous les types de véhicules ;
- I_3 : Passager 3 voyage avec un gros bagage et possède un permis de conduire ;
- I_4 : Passager 4 n'a imposé aucune condition par rapport au type de véhicule.

L'application de la procédure d'affectation nous donne les résultats suivants :

$$S^{[8h,11h]}_{(u,v)} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1^{[8h30,9h30]}_{(u_1,v_1)} & V2^{[8h40,10h30]}_{(u_2,v_2)} & V3^{[9h20,11h00]}_{(u_2,v_2)} & V4^{[t_0,t_0+2h]}_{(u_3,v_3)} \end{array} \\ \begin{pmatrix} \mathbf{*} & 0 & 0 & 0 \\ * & * & 0 & 0 \\ * & 0 & * & * \\ * & * & * & * \end{pmatrix} \end{array} \quad \text{(III.14)}$$

Nous constatons qu'il existe une seule possibilité d'affectation pour I_1, l'affectation entre I_1 et V_1 devient donc être *obligatoire*. Après la modification de $*$ à 1, le schéma d'affectation devient :

$$S^{[8h,11h]}_{(u,v)} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1^{[8h30,9h30]}_{(u_1,v_1)} & V2^{[8h40,10h30]}_{(u_2,v_2)} & V3^{[9h20,11h00]}_{(u_2,v_2)} & V4^{[t_0,t_0+2h]}_{(u_3,v_3)} \end{array} \\ \begin{pmatrix} \mathbf{1} & 0 & 0 & 0 \\ * & * & 0 & 0 \\ * & 0 & * & * \\ * & * & * & * \end{pmatrix} \end{array} \quad \text{(III.15)}$$

Ce schéma couvre l'ensemble des possibilités intéressantes pour affecter des offres aux demandes et des interdictions selon leurs préférences et contraintes temporelles. Toutefois, les affectations respectant ce modèle ne garantissent pas forcément une solution qui possède la meilleure fonction d'adaptation. Les affectations qui ne représentent pas une bonne fonction d'adaptation ont peu de chance d'entrer dans les générations suivantes dans la phase de sélection de l'algorithme génétique. Cette approche est décrite par la Figure III.24. En effet, elle consiste à prendre les spécifications au niveau des préférences et du temps comme contrainte pour générer la population initiale, et ensuite, à mettre en place les étapes classiques de l'optimisation génétique, en même temps, le modèle d'affectation induit par le schéma est pris en compte au cours des générations. Selon le schéma d'affectation, la faisabilité des individus est vérifiée. Si l'individu ne respecte pas le schéma, un opérateur de correction sera activé pour régler ce problème. Cette méthode nous permet de réduire l'espace de recherche afin d'accélérer la convergence d'optimisation et d'assurer une bonne qualité des solutions finales lors d'utiliser l'algorithme génétique.

La démarche de cet algorithme génétique contrôlé est résumée dans l'algorithme *AFFECTATION*.

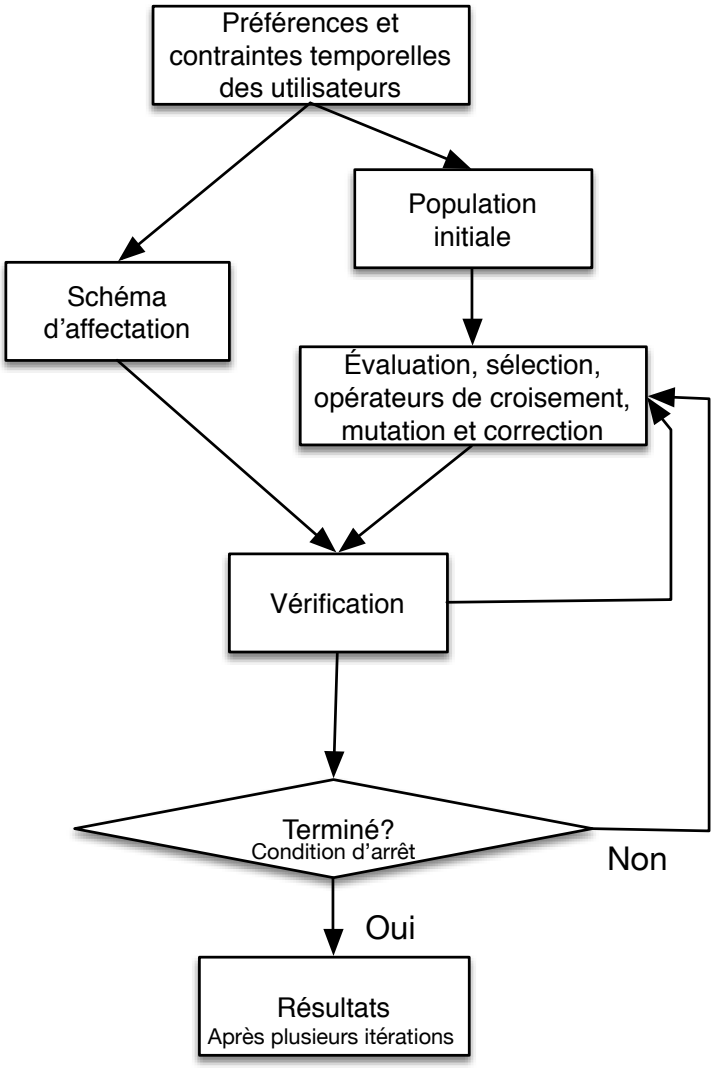


FIGURE III.24: Algorithme génétique contrôlé pour l'affectation

Algorithm 12 Algorithme génétique contrôlé pour l'affectation des demandes aux offres (**AFFECTATION**)

1: Initialisation de la population selon les contraintes
2: Générer le schéma d'affectation selon Algorithme 11
3: Évaluer les individus de la population pour appliquer la sélection par roulette sur les couples de chromosomes choisis selon l'évaluation précédente
4: Générer une nouvelle population en appliquant l'opérateur de croisement avec une probabilité P_c et l'opérateur de mutation avec une probabilité P_m
5: Vérifier si le schéma d'affectation est respecté et appliquer si nécessaire l'opérateur de correction
6: Répéter les étapes précédentes (de la ligne 2 à 5) jusqu'à atteindre le nombre d'itérations fixé.
7: **Sortie :** chromosomes conformes à la faisabilité et qui ont le meilleur score de la fonction d'évaluation

III.7.2 Opérateur de croisement

L'opérateur de croisement est une procédure de recombinaison qui produit un ou plusieurs enfants à partir de deux parents dans la génération courante.

Après avoir choisi les deux chromosomes parents, nous appliquons le croisement à un point présenté ci-dessous avec l'Algorithme *CROISEMENT*.

Algorithm 13 Algorithme de l'opérateur de croisement (**CROISEMENT**)

1: Choisir aléatoirement deux chromosomes dans la génération courante M_1 et M_2
2: Choisir aléatoirement une position r par rapport à la numérotation de la ligne ▷ Choisir une ligne comme repère
3: L'enfant E_1 garde les mêmes gènes entre ligne 1 et ligne r que M_1, et il complète le reste en empruntant les gènes de M_2 à partir de la ligne $r+1$
4: L'enfant E_2 garde les mêmes gènes entre ligne 1 et ligne r que M_2, et il complète le reste en empruntant les gènes de M_1 à partir de la ligne $r+1$
5: **Sortie :** Deux chromosomes enfants E_1 et E_2

Ci-dessous un exemple d'illustration pour montrer le fonctionnement de cet opérateur.

III.7.2.1 Exemple d'illustration de l'opérateur de croisement

D'après le schéma d'affectation, deux chromosomes parents $P1^{[8h,11h]}_{(u,v)}$ et $P2^{[8h,11h]}_{(u,v)}$ sont représentés comme suit :

$$P1^{[8h,11h]}_{(u,v)} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1^{[8h30,9h30]}_{(u_1,v_1)} & V2^{[8h40,10h30]}_{(u_2,v_2)}(2,3) & V3^{[9h20,11h00]}_{(u_2,v_2)}(0,3) & V4^{[t_0,t_0+2h]}_{(u_3,v_3)}(2) \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix} \end{array}$$

$$P2^{[8h,11h]}_{(u,v)} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1^{[8h30,9h30]}_{(u_1,v_1)} & V2^{[8h40,10h30]}_{(u_2,v_2)}(0,3) & V3^{[9h20,11h00]}_{(u_2,v_2)}(1,3) & V4^{[t_0,t_0+2h]}_{(u_3,v_3)}(2) \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \end{pmatrix} \end{array}$$

Le croisement est appliqué sur la position choisie aléatoirement, deux chromosomes enfants sont ainsi obtenus.

$$E1^{[8h,11h]}_{(u,v)} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1^{[8h30,9h30]}_{(u_1,v_1)} & V2^{[8h40,10h30]}_{(u_2,v_2)}(1,3) & V3^{[9h20,11h00]}_{(u_2,v_2)}(1,3) & V4^{[t_0,t_0+2h]}_{(u_3,v_3)}(2) \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \end{pmatrix} \end{array}$$

$$E2^{[8h,11h]}_{(u,v)} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1^{[8h30,9h30]}_{(u_1,v_1)} & V2^{[8h40,10h30]}_{(u_2,v_2)}(1,3) & V3^{[9h20,11h00]}_{(u_2,v_2)}(0,3) & V4^{[t_0,t_0+2h]}_{(u_3,v_3)}(2) \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix} \end{array}$$

III.7.3 Opérateur de mutation

Afin de garantir l'exploitation de l'espace d'état et d'éviter une convergence prématurée, on utilise l'opérateur de mutation qui permet de générer des nouveaux gènes avec une probabilité très faible après le croisement (cf. Algorithme *MUTATION*).

Algorithm 14 Algorithme de l'opérateur de mutation (**MUTATION**)

1: Sélection d'un chromosome M au sein de la génération courante d'après la probabilité de mutation P_m
2: $(R, T) \leftarrow$ dimensions de la matrice M
3: Choisir aléatoirement une position r où $r \leq R$ ▷ Choisir une ligne comme repère
4: Identifier t_0 où $M(r, t_0) = 1$
5: Choisir aléatoirement une position t où $t \leq T$ et $t \neq t_0$ ▷ Choisir une colonne comme repère
6: $M_{(r,t)} \leftarrow 1$ et $M(r, t_0) \leftarrow 0$
7: **Sortie :** Un chromosome après mutation M

III.7.3.1 Exemple d'illustration

Le chromosome initial est :

$$M^{[8h,11h]}_{(u,v)} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1^{[8h30,9h30]}_{(u_1,v_1)} & V2^{[8h40,10h30]}_{(u_2,v_2)}(1,3) & V3^{[9h20,11h00]}_{(u_2,v_2)}(0,3) & V4^{[t_0,t_0+2h]}_{(u_3,v_3)}(2) \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Après la mutation, le chromosome M devient :

$$M^{[8h,11h]}_{(u,v)} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1^{[8h30,9h30]}_{(u_1,v_1)} & V2^{[8h40,10h30]}_{(u_2,v_2)}(1,3) & V3^{[9h20,11h00]}_{(u_2,v_2)}(1,3) & V4^{[t_0,t_0+2h]}_{(u_3,v_3)}(1) \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

III.7.4 Algorithme de correction

Les chromosomes obtenus à l'issue du croisement et de la mutation peuvent être des solutions infaisables qui ne respectent pas toutes les contraintes du problème. L'algorithme de correction permet de remédier à ce genre de problème. Avant la mise en place de cette mesure, une étape de vérification est réalisée sur les conditions qui se partagent en deux types : du côté des demandes et du côté des offres. Pour la première catégorie, les conditions s'expriment sous la forme de schéma. Pour la deuxième catégorie, ce sont des conditions sur la quantité des véhicules et des places disponibles.

L'algorithme de correction des chromosomes sur les contraintes est décrit par l'Algorithme *CORRECTION*.

Algorithm 15 Algorithme de correction des chromosomes (**CORRECTION**)

1: $S \leftarrow$ le schéma d'affectation
2: $M \leftarrow$ le chromosome à "corriger"
3: $(R, T) \leftarrow$ dimensions de S et M
4: $j \leftarrow 0$
5: Pour $j < T$
6: Vérifier et corriger la colonne j de M
7: $j \leftarrow j + 1$
8: Fin pour
9: Vérifier et corriger M auprès de S
10: **Sortie :** Un chromosome faisable après correction M

III.7.4.1 Exemple d'illustration

Après avoir vérifié le chromosome M qui est comme suit,

$$M_{(u,v)}^{[8h,11h]} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1_{(u_1,v_1)}^{[8h30,9h30]} & V2_{(u_2,v_2)}^{[8h40,10h30]}(1,3) & V3_{(u_2,v_2)}^{[9h20,11h00]}(-1,3) & V4_{(u_3,v_3)}^{[t_0,t_0+2h]}(2) \end{array} \\ \begin{pmatrix} 1 & 0 & \mathbf{0} & 0 \\ 0 & 1 & \mathbf{0} & 0 \\ 0 & 0 & \mathbf{1} & 0 \\ 0 & 0 & \mathbf{1} & 0 \end{pmatrix} \end{array}$$

On constate que la condition de la quantité est violée sur la colonne de $V3$.

On le corrige avec le schéma d'affectation S,

$$S_{(u,v)}^{[8h,11h]} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1_{(u_1,v_1)}^{[8h30,9h30]} & V2_{(u_2,v_2)}^{[8h40,10h30]} & V3_{(u_2,v_2)}^{[9h20,11h00]} & V4_{(u_3,v_3)}^{[t_0,t_0+2h]} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ * & * & 0 & 0 \\ * & 0 & * & * \\ * & * & * & * \end{pmatrix} \end{array}$$

Un premier élément *1* de la colonne de *V3* devient 0, en même temps, un deuxième élément 0 sur la même ligne du premier élément devient 1. On obtient ainsi un chromosome M

$$M_{(u,v)}^{[8h,11h]} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} V1_{(u_1,v_1)}^{[8h30,9h30]} & V2_{(u_2,v_2)}^{[8h40,10h30]}(1,3) & V3_{(u_2,v_2)}^{[9h20,11h00]}(0,3) & V4_{(u_3,v_3)}^{[t_0,t_0+2h]}(2) \end{array} \\ \begin{pmatrix} 1 & 0 & \mathbf{0} & 0 \\ 0 & 1 & \mathbf{0} & 0 \\ 0 & 0 & \mathbf{1} & 0 \\ 1 & 0 & \mathbf{0} & 0 \end{pmatrix} \end{array}$$

III.7.5 Opérateur de sélection

La méthode que nous avons utilisée pour la sélection est celle de l'élitisme complétée par la sélection par roulette. Par ailleurs, les deux meilleurs individus parmi la population sont choisis et entrent dans la nouvelle génération en tant qu'individus élites.

Dans l'optique de trouver les meilleurs résultats, il faut établir la fonction d'adaptation permettant de mesurer les individus et de les faire évoluer. Les individus ayant les meilleurs scores d'adaptation sont plus sollicités à être sélectionnés pour l'opérateur de croisement.

III.7.5.1 Fonction d'évaluation

En considérant qu'il s'agit d'un problème d'optimisation multiobjectif, nous prenons en compte tous les critères concernés pour évaluer les chromosomes. Nous présentons ici à titre d'exemple une sélection à deux critères qui se présente pour la fonction d'évaluation :

— le coût financier ;
— le temps.

Le premier critère est en effet les frais de déplacement sur le trajet s'exprimant en Euros(€). En effet, le coût varie selon le type de véhicule pris sur les trajets de mêmes extrémités. C'est un critère à minimiser.

Au niveau du coût de trajet, le calcul se divise en trois cas. Pour le transport en commun, le tarif d'un trajet est prédéfini et consultable auprès de l'exploitant de transport correspondant. Pour le covoiturage, le frais est ainsi fixé par les fournisseurs de service (les conducteurs qui offrent des places libres dans leurs véhicules sous forme de covoiturage). Quant à l'autopartage, le coût est obtenu selon les formules qui se basent généralement sur la distance parcourue, la durée de la location ou encore les frais d'utilisation.

Le deuxième critère est le temps de parcours. En effet, le temps de parcours est considéré comme un indicateur essentiel pour le choix d'itinéraire, quelque soit le mode de déplacement.

Le temps de parcours est calculé par addition de plusieurs composants, à savoir :

— le temps de transit t_{tr} (cf. section III.2) ;
— le temps de transport ou de voyage t_v ;
— le temps d'attente t_a.

Le temps de parcours t_p peut alors s'écrire comme : $t_p = t_{tr} + t_v + t_a$.

Pour les transports en commun, le parcours est prédéfini et donc généralement une durée fixe. Le calcul se base sur l'horaire théorique de la ligne. De plus, on ne prend pas en compte le temps de stationnement pour le calcul de temps de parcours.

Pour les véhicules partagés, on distingue deux cas ; l'autopartage et le covoiturage. Pour l'autopartage, à part le temps de transit, le temps de parcours se constitue d'une partie pour le temps de voyage t_v et une autre relative au temps de prise en charge et dépôt (stationnement) t_s. On a alors : $t_p = t_{tr} + t_v + t_s$.

Pour le covoiturage, s'il s'agit d'un participant, le temps de parcours est mesuré comme celui avec les transports en commun ; en revanche, s'il s'agit d'un conducteur, le temps de parcours peut être calculé avec la même formule que l'autopartage.

En raison de leur conception et la simplification de données, ces méthodes d'estimation subissent des limites du point de vu de fiabilité. Toutes fois, nous n'allons pas élaborer de nouvelles approches pour estimer le temps de parcours, mais nous cherchons plutôt à valider les méthodes proposées pour la planification d'itinéraire.

Cette approche consiste à associer un poids à chaque fonction objectif individuelle et à les linéariser afin d'obtenir la fonction d'agrégation.

$$F(\mathbf{M}) = \sum_{i=1}^{2} \omega_i F_i(\mathbf{M}) \tag{III.16}$$

où

- $F(\cdot)$ est la somme pondérée des fonctions objectifs F_i qui a le poids ω_i ;
- F_i est le coût généralisé du critère i ;
- $\forall i \in \{1, \ldots, 2\}, \omega_i \in [0, 1]$;
- $\sum_{i=1}^{2} \omega_i = 1$.

Les chromosomes, à chaque itération, sont évalués et se voient attribuer des scores d'adaptation par la fonction III.16. Selon les scores représentant leurs niveaux d'adaptation, les meilleurs individus sont ainsi choisis par la sélection par l'élitisme complétée par la sélection par roulette pour se reproduire et entrer dans la génération prochaine.

III.8 Conclusion

Dans ce chapitre, nous avons présenté l'approche de la planification d'itinéraire dans un réseau de transport comodal avec la technique du graphe hiérarchisé dans un premier temps. Puis, nous avons développé la méthode basée sur le graphe temps-étendu pour déterminer les itinéraires multimodaux. Le problème de la séquence multimodale est formulé et résolu à l'aide de la recherche bidirectionnelle. De plus, le raffinement des fenêtres temporelles permet de réduire l'espace de recherche. Enfin, le problème d'affectation est établi pour mettre en relation les offres et les demandes sur une section de route et résolu avec une approche basée sur l'algorithme génétique.

Dans le chapitre suivant, nous présentons l'approche multi-agents que nous avons adoptée pour intégerer les méthodes présentées dans ce chapitre. L'architecture multi-agents, ainsi que les composants agents seront détaillés pour le traitement des requêtes, l'optimisation de la recherche et la composition d'itinéraires. Nous exposons en particulier la coalition entre les agents pour obtenir les itinéraires comodaux.

Chapitre IV

Architecture multi-agents et mise en œuvre des coalitions d'agents pour la composition des itinéraires

Sommaire

IV.1 Introduction

L'objectif que nous avons fixé dans le premier chapitre est de concevoir et développer un système d'information de transport comodal au service de la mobilité avancée. Comme nous l'avions expliqué dans la section I.6.1, nous proposons une approche composée de trois étapes, où chacune comporte un niveau d'optimisation. Le troisième niveau d'optimisation résulte des étapes précédentes dans lesquelles les itinéraires sont formulés en sections de routes et les processus d'affectation sont réalisés au sein des entités logicielles, à savoir les agents, et permet la composition de services de transport afin d'obtenir les itinéraires complets et optimisés. Nous employons désormais la notion de coalition des agents pour illustrer la coopération entre les agents.

La suite du chapitre est répartie de manière suivante : en premier lieu, nous présentons la modélisation multi-agents du système proposé, ensuite, l'accent est mis sur la formation d'une coalition des agents représentant la combinaison des services de transport et nous finissons par donner les détails sur l'approche de la coalition.

IV.2 Modélisation multi-agents du système proposé

Dans l'optique de résoudre le problème, il est nécessaire de mettre en place un procédé efficace, notre travail s'intéresse au processus d'optimisation des requêtes. Ce besoin réside dans le fait que les requêtes usagers sont probablement nombreuses et simultanées et que les sources d'information sont distantes et distribuées.

IV.2.1 Architecture multi-agents proposée

L'architecture multi-agents du système dynamique proposé est visualisée sur la Figure IV.1. Elle utilise plusieurs types d'agents au sein du système sur la base de la modélisation que nous avons préalablement établie. Les différents agents mis en place assurent le bon fonctionnement du système tel que l'identification du domaine de recherche, l'affectation optimisée des véhicules aux usagers et la coalition des agents pour la composition des itinéraires.

Ces agents sont présentés avec des comportements spécifiques qui leurs sont propres. En soulignant leurs fonctionnements, nous n'oublions pas leurs propres actions et leurs collaborations dans le processus global.

Nous distinguons plusieurs types d'agents au sein de ce système :

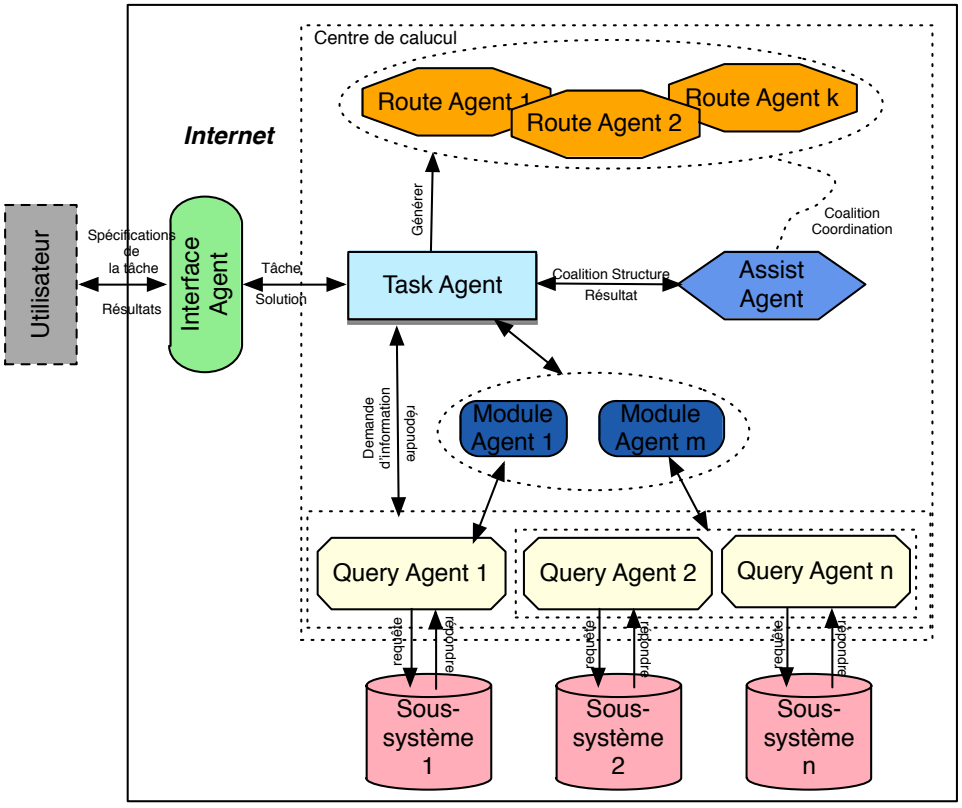


FIGURE IV.1: L'architecture SMA basée sur les agents communicants

— *Interface Agent* : ce type d'agent est associé aux usagers du système. Il est responsable de récupérer la requête et de renvoyer les résultats.
— *Query Agent* : ce type d'agent est conçu pour communiquer avec les sous-systèmes pour la collecte des informations nécessaires.
— *Module Agent* : ce type d'agent représente les modules du graphe. Chaque Module Agent est responsable d'un ensemble de Query Agents.
— *Task Agent* : ce type d'agent a plusieurs rôles. Il a pour mission d'identifier le domaine de recherche, de calculer les plus courts chemins, d'arranger les itinéraires avec une séquence de nœuds et de générer les *Route Agents*.
— *Route Agent* : ce type d'agent est utilisé pour représenter les tronçons de route sur lesquels les offres sont affectées aux demandes.
— *Assist Agent* : ce type d'agent a pour mission d'assister la formulation de la coalition entre les *Route Agents* en vue d'obtenir les solutions.

La Figure IV.2 illustre les comportements des agents. L'Interface Agent permet d'établir la communication entre les usagers et le système proposé. Les Task Agents traitent les requêtes reçues en deux étapes : identification du domaine de recherche avec les Module Agents et la recherche des trajets attractifs avec la génération des Route Agents qui représentent des morceaux de routes. Par la suite, un processus d'affectation entre les

ressources de transport et les requêtes se réalise pour les Route Agents. Les différentes combinaisons de morceaux de routes sont obtenues via l'élaboration de la coalition entre les Route Agents. À l'aide des Assist Agents, nous obtenons les réponses aux requêtes avec les combinaisons optimales des morceaux de routes.

FIGURE IV.2: Comportements des agents au sein du système

Dans le graphe de modules, chaque module constitue un ensemble de graphes monomodaux de même niveau dont chacun communique avec un Query Agent. Un Module Agent associé à un module de graphe est responsable des Query Agents de ce même module. Le domaine de recherche est identifié avec le calcul des chaînes de modules sur le graphe de modules. Les Module Agents pour des requêtes ainsi que, les préférences de mode des usagers pour les Query Agents sont identifiés. Des Route Agents générés par Task Agents représentent des tronçons de route des itinéraires attractifs pour des requêtes. Ces Route Agents, reponsables d'affecter des vehicules aux demandes, se relient selon les critères pour formuler les itinéraires comodaux optimisés à l'aide des Assist Agents pour des requêtes.

La coalition des Route Agents se forme selon les chemins attractifs trouvés au sein de la chaîne de modules. Elle peut être formée en utilisant le chaînage avant ou avec le chaînage arrière ou encore avec la recherche bidirectionnelle.

IV.2.2 Agents et comportements

Nous définissons l'ensemble des agents de la structure du SMA. Grâce à leurs caractéristiques de réactivité, les agents effectuent des actions simples ou complexes qui leur permettent d'interagir avec d'autres agents du même système.

Les différents agents se comportent différemment, chacun est guidé par son propre objectif et cherche à accomplir un certain nombre de tâches qui lui sont attribuées. Dans ce qui suit, les comportements des agents du système sont décrits et illustrés avec des diagrammes d'activité.

IV.2.2.1 Interface Agent

L'Interface Agent représente une interface entre le système et son utilisateur, et permet la communication entre les deux. En effet, l'Interface Agent permet à l'utilisateur d'accéder au système pour composer une requête en désignant les éléments nécessaires de celle-ci (comme le départ, l'arrivée etc.) et recevoir les résultats. Il a deux rôles principaux : d'un côté, un agent est créé avec l'accès au système d'un utilisateur pour assister celui-ci à formuler les spécifications de la tâche telles que les extrémités du trajet, les préférences, les options et les fenêtres de temps dans lesquelles il désire être desservi. Celles-ci seront ensuite transmises au Task Agent qui est responsable de traiter cette requête ; d'un autre côté, après avoir reçu le résultat de la requête de la part du Task Agent, l'Interface Agent diffuse la solution obtenue d'une manière appropriée selon le type de terminal utilisé par l'utilisateur.

Son comportement est illustré par le diagramme d'activité de la Figure IV.3.

Lorsque l'utilisateur accède au système et saisit sa requête, l'Interface Agent est prêt à vérifier les éléments de celle-ci. Par la suite, il envoie la requête au Task Agent dédié qui est créé préalablement et se situe dans un état d'attente par défaut, dans le cas contraire, un nouveau Task Agent est crée automatiquement. Les solutions que l'agent reçoit sont affichées à l'utilisateur dès que la réponse lui est donnée par le Task Agent. L'Interface Agent attend la confirmation de l'utilisateur qui est envoyée par la suite au Task Agent pour la mise à jour de l'état des ressources de transport.

IV.2.2.2 Query Agent

Ce type d'agent est créé afin de collecter les données nécessaires pour assurer le bon fonctionnement du système, ces informations sont hébergées sur les sous-systèmes. Un agent

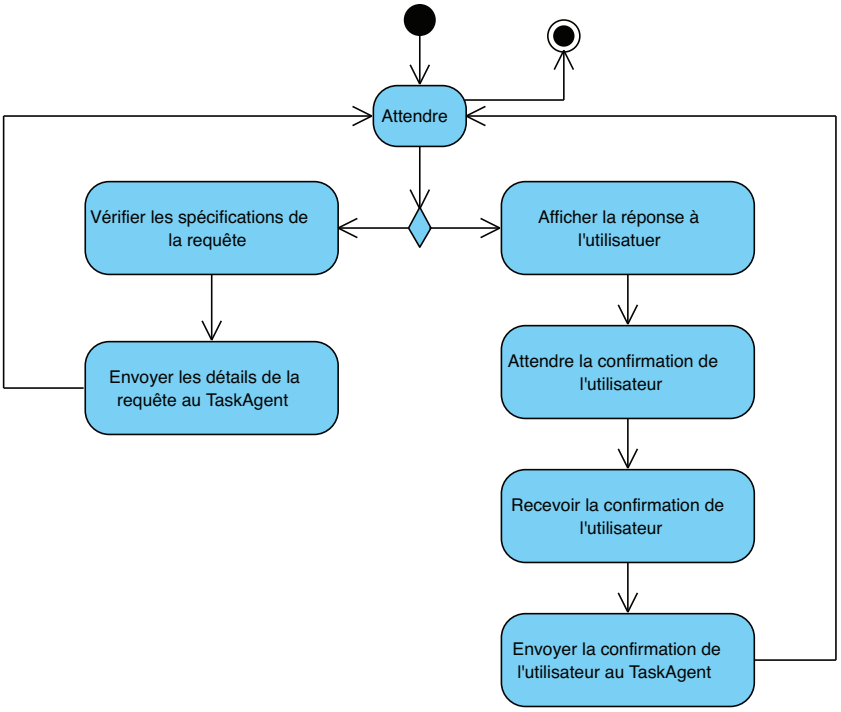


FIGURE IV.3: Diagramme d'activité d'un Interface Agent

de type Query Agent est associé à chaque sous-système pour effectuer la communication avec celui-ci.

Le fonctionnement du Query Agent se base sur la collecte des données via l'envoie des requêtes aux sous-systèmes intégrés correspondants et la réception des réponses. Son comportement est illustré par le diagramme d'activité de la Figure IV.4.

En effet, quand le Task Agent identifie les sous-systèmes concernés par la requête, il envoie les demandes aux Query Agents. Ces derniers reformulent et transforment ces demandes pour les rendre conformes aux exigences de chaque sous-système. Après avoir reçu les résultats venant du sous-système, le Query Agent les transmettent instantanément au Task Agent pour les traitements suivants.

Cette collecte des informations est indispensable tout au long de la résolution du problème. Les informations collectées auprès des sous-systèmes par les Query Agents permettent de calculer les itinéraires dans les sous-graphes et d'identifier les offres dans la phase d'affectation afin d'obtenir un traitement optimal.

Un Query Agent en état d'attente est prêt à recevoir la demande de la part du Task Agent. Lors de la réception de la demande, le Query Agent communique avec le sous-système correspondant en précisant un certain nombre d'éléments tels que le point de

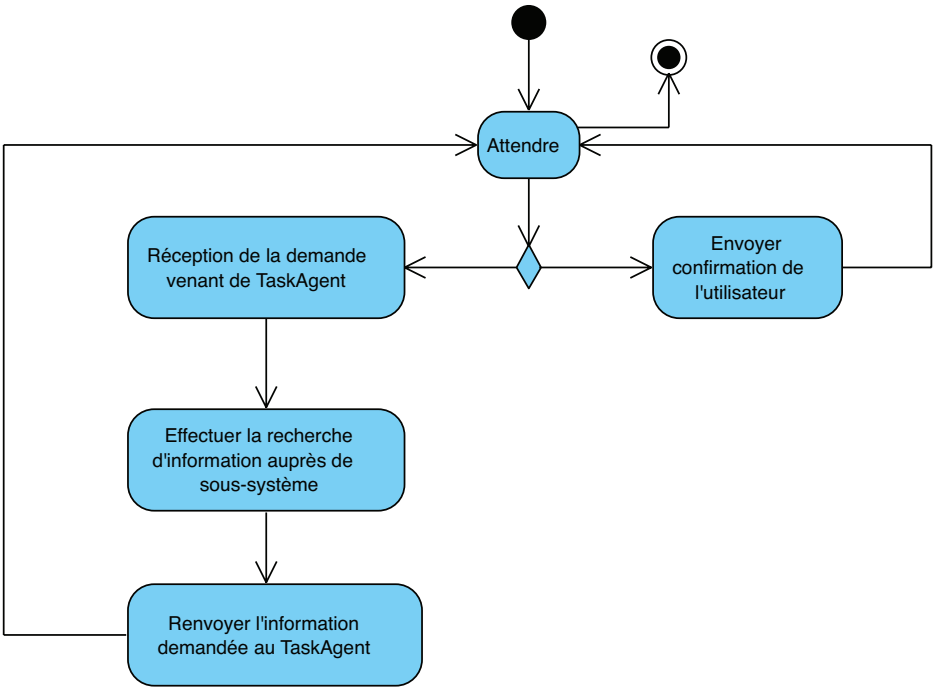


FIGURE IV.4: Diagramme d'activité d'un Query Agent

départ, le point d'arrivée et les intervalles de temps. La recherche d'informations s'effectue auprès des sous-systèmes et peut retourner un ensemble de solutions pour construire des graphes, comme par exemple, celui de type temps étendu. En outre, le Query Agent joue un autre rôle qui lui permet de communiquer avec le sous-système correspondant pour l'informer de la confirmation du côté utilisateur.

IV.2.2.3 Module Agent

À chaque module du graphe hiérarchisé, nous associons un Module Agent qui est donc responsable d'un nombre de composants. Après que le Task Agent a identifié l'espace de recherche, les Modules Agents correspondants sont consultés pour coordonner les Query Agents dont ils sont responsables.

Le comportement du Module Agent est décrit par le diagramme d'activité présenté dans la Figure IV.5.

IV.2.2.4 Task Agent

Ce type d'agent a pour mission d'exécuter plusieurs tâches différentes en coopérant avec les autres agents.

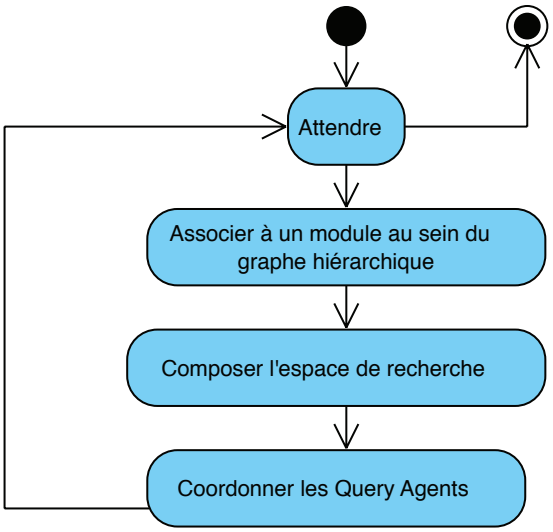


FIGURE IV.5: Diagramme d'activité d'un Module Agent

Après avoir reçu la requête d'itinéraire venant des Interface Agents, il cherche à identifier le domaine de recherche pour la requête dont le point de départ et le point d'arrivée sont repérés. Ensuite, le processus d'identification du domaine de recherche est lancé selon les algorithmes introduits dans la section III.4.1. Sur celui-ci, l'algorithme de recherche d'itinéraire s'exécute (cf. section III.4.2.1) en vue de trouver les chemins attractifs.

Le problème de la séquence multimodale pour chaque chemin attractif est résolu par le Task Agent avec l'approche exacte proposée (cf. section III.5.3) sur le graphe temps-étendu. Au cours de ce processus, il communique avec les Query Agents correspondants afin d'acquérir les offres sur la section de route considérée. Les combinaisons optimales des services de transport sont visualisées à l'aide des Interface Agents.

Les flottes de véhicules sont identifiées pour les sections de route grâce au processus de raffinement de fenêtre de temps sur les nœuds intermédiaires de chaque chemin attractif. Ensuite, le Task Agent génère aux Route Agents les modules de tronçons de routes avec les véhicules identifiés. Enfin, les solutions issues de la coalition des Route Agents sont transmises aux usagers à travers des Interface Agents.

Le comportement de ce type d'agent est décrit par le diagramme d'activité qui présenté par la Figure IV.6.

IV.2.2.5 Route Agent

Les Routes Agents sont mis en place pour représenter les tronçons de route sur lesquels les demandes et offres sont mises en relation. Ce type d'agent porte les informations

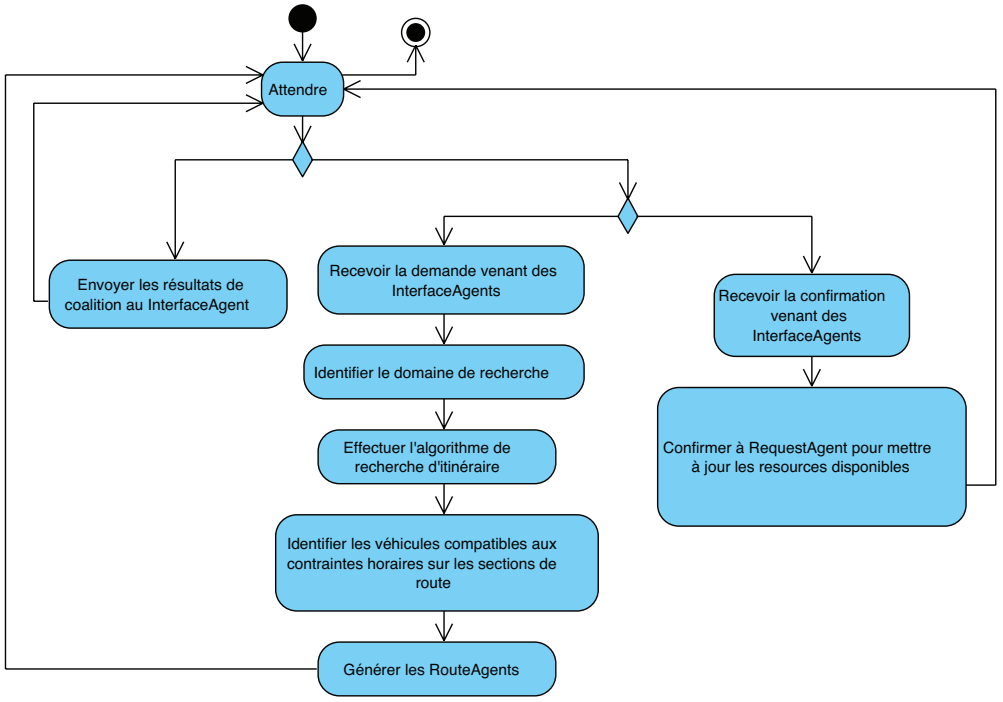


FIGURE IV.6: Diagramme d'activité d'un Task Agent

telles que les requêtes concernées, les renseignements sur les véhicules correspondants, ainsi qu'une matrice d'affectation des usagers aux véhicules.

Sur les tronçons de routes, les véhicules disponibles sont identifiés selon des fenêtres de temps correspondantes. Un chromosome sous la forme d'une matrice est obtenu pour chaque tronçon de route avec l'optimisation du processus d'affectation des requêtes concernées aux véhicules capables d'assurer ce tronçon de route. Les contraintes telles que les fenêtres de temps et les préférences utilisateurs au niveau du type de véhicules sont définies pendant ce processus d'affectation.

Après avoir fini le processus d'affectation, les Route Agents s'engagent dans les coalitions dont l'objectif est de combiner les routes et de reconstruire l'itinéraire avec les véhicules identifiés. La section IV.3 est dédiée à la présentation de la coalition entre les Route Agents. A l'aide d'Assist Agent, la coalition des Route Agents sera vérifiée et déterminée.

Le comportement de Route Agent est décrit par le diagramme d'activité visualisé par la Figure IV.7.

IV.2.2.6 Assist Agent

Cet agent permet de coordonner le processus de la coalition auprès de son architecture. L'Assist Agent est responsable aussi de la vérification de la viabilité de la coalition et de

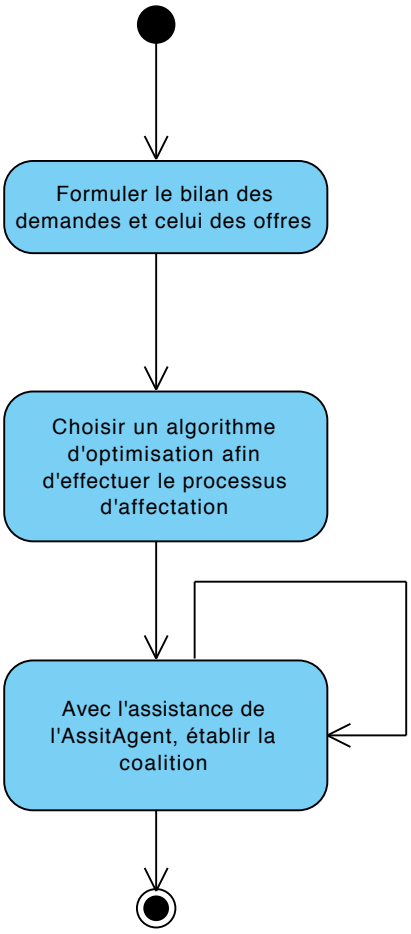


FIGURE IV.7: Diagramme d'activité d'un Route Agent

l'évaluation de chaque étape de cette coalition. Les résultats qu'il obtient seront transmis par la suite au Task Agent pour la transmission finale aux usagers.

Son comportement est illustré par le diagramme d'activité présenté par la Figure IV.7.

IV.2.3 Comportements agents cohérents

Le domaine de recherche est identifié pour une requête d'itinéraire avec le graphe de modules. Au sein d'un réseau de transport multimodal, il existe une multitude de combinaisons possibles permettant de se déplacer d'un point d'origine vers un point de destination. Nous présumons donc que les usagers choisissent un chemin parmi les chemins les plus attractifs. Pour calculer ces derniers, l'algorithme de k plus courts chemins est appliqué. Ces chemins attractifs génèrent un ensemble de structures de la coalition dans la phase suivante. Après l'extraction des itinéraires attractifs, l'approche proposée passe à la deuxième phase de son fonctionnement, à savoir, la recherche des meilleurs solutions compte tenu des critères d'optimisation.

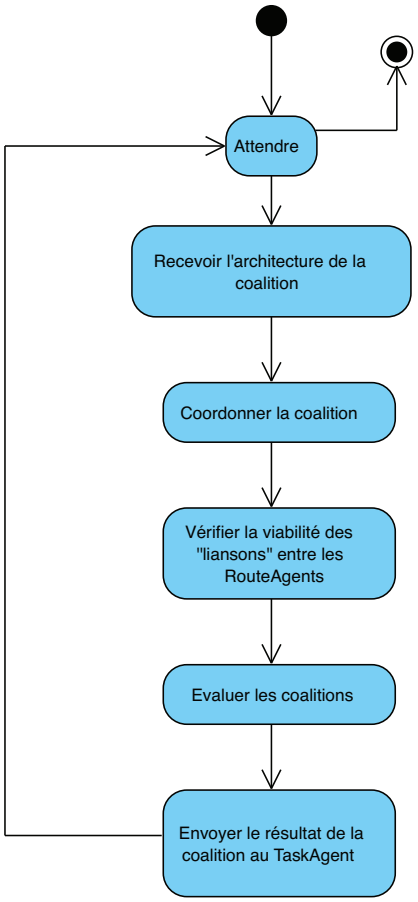


FIGURE IV.8: Diagramme d'activité d'un Assist Agent

Considérons un ensemble de requêtes reçues pendant une petite période de temps Δt. Pour chaque requête, le sous-graphe correspondant à la paire $O - D$ est obtenu à l'aide du graphe hiérarchisé construit préalablement. Par la suite, on obtient l'ensemble des itinéraires attractifs sur lesquels les flottes de véhicules compatibles aux contraintes horaires sont identifiées sur les nœuds. Ensuite, les processus d'affectation sont appliqués sur les morceaux de routes pour l'ensemble des requêtes. Enfin, nous passons à la phase permettant de relier les routes pour former les itinéraires optimaux répondant aux requêtes.

Du côté des agents au sein du SMA, les Interface Agents accueillent les requêtes durant la période de temps Δt et les transmettent aux Task Agents qui sont responsables d'identifier le domaine de recherche, de calculer les chemins attractifs, d'identifier les véhicules conformes aux contraintes temporelles sur chaque section de route en coopérant avec les Query Agents et générer les Route Agents. La coalition entre les Route Agents à l'aide des Assist Agents sont formées en suivant les structures de la coalition afin d'avoir les combinaisons optimales des services de transport.

Nous nous concentrons par la suite sur la coalition pour former les itinéraires.

IV.3 Coalition pour former les itinéraires

Nous détaillons dans cette partie le processus de la formation de coalition des services au sein d'un système d'information de transport comodal [Wang et al., 2014a] . La combinaison des services est réalisée à l'aide de la coalition pour obtenir un itinéraire complet avec des segments de route bien assignés pour les usagers. Parallèlement, la formation de coalition est menée à trouver la combinaison optimale en respectant les préférences des usagers et leurs contraintes temporelles. Compte tenu des multi-rôles des agents, leurs comportements et protocoles d'interaction sont introduits afin d'établir chaque étape du processus de la coalition.

IV.3.1 La formation des coalitions entre agents

Au sein d'un système multi-agents, les agents coopèrent lors de l'exécution des tâches qui sont en dehors de leurs domaines de compétences. Les coopérations entre les agents, sous forme de coalition, permettent d'exécuter des tâches plus compliquées.

Plusieurs modèles de formation des coalitions sont proposés dans la littérature visant à établir l'union des agents [Klusch and Shehory, 1996] [Zlotkin and Rosenschein, 1994] [Sandhlom and Lesser, 1997] [Sandholm et al., 1999] [Shehory and Kraus, 1998]. À partir des différentes hypothèses, ces modèles de la formation d'une coalition sont proposés dans des divers contextes. Les agents sont divisés en deux catégories : le premier type d'agent essaie de rendre son utilité individuelle maximale, en revanche, le second type d'agent vise à maximiser l'utilité du groupe [Shehory and Kraus, 1998].

Dans le domaine de la formation de coalition pour le marché électronique, le travail [Breban and Vassileva, 2002] a proposé des coalitions de long terme entre les clients et les commerçants basées sur la relation de confiance entre les agents. Dans le modèle proposé, les agents jouent les rôles des clients et ceux des commerçants. Le mécanisme de la formation d'une coalition est conçu au niveau des agents (microscopique) et analysé au niveau du système (macroscopique). L'évaluation permet de comparer la stratégie individuelle et sociale et à analyser les comportements sous les différentes circonstances.

Le travail cité plus haut a permis de franchir la limite des recherches antérieures qui focalisent sur la coalition d'une seule transaction, ainsi que sur la mémorisation des expériences précédentes. En considérant les relations entre les agents, le mécanisme proposé étend le concept existant de la coalition temporaire à celle à long terme formée des

clients et des commerçants. Le travail a aussi montré que le mécanisme de la formation d'une coalition apporte de la stabilité au système au niveau du nombre de coalitions et de la dynamique globale. Le mécanisme emploie la communication réduite qui augmente la capacité de monter en charge pour un grand nombre d'agents et d'interactions.

Dans le travail [Tong et al., 2009], l'auteur a proposé un modèle de service agent qui intègre le service Web et la technologie d'agent logiciel sous forme d'une entité cohésive. En se basant sur ce modèle proposé, un algorithme distribué de coalition d'agents pour la composition automatique des Web services, nommé DACA, est proposé. DACA est basé sur la prise de décision distribuée des services d'agents autonomes et adresse le caractère naturel de la composition des Web services. A partir de l'algorithme proposé, les solutions avec coût de communication relativement faible sont obtenues.

Dans le travail [Sandhlom and Lesser, 1997], l'auteur analyse les coalitions entre les agents qui sont équivalentes aux problèmes d'optimisation combinatoire pour opérer efficacement. Un modèle de rationalité bornée est adopté où les ressources de calcul sont coûteuses. Quand il n'est pas possible de résoudre le problème d'une manière optimale, une théorie universelle de coalitions entre des agents rationnels, indépendants des applications et protocoles est utilisée. La structure optimale de la coalition et sa stabilité est affectée par la performance de l'algorithme et le coût de la communication. L'auteur a abordé l'analyse de cette relation théoriquement pour la première fois.

Dans [Narayanan and McIlraith, 2002], le travail mené vise à permettre l'arrangement et la technologie de raisonnement automatique pour décrire, simuler, composer, tester et vérifier la composition des Web services. L'ontologie DAML-S DAML+OIL est prise comme point de départ pour décrire les capacités des Web service. Ensuite, la sémantique pour un sous-ensemble de DAML-S en termes d'un premier ordre de langage logique est défini. Une analyse de la complexité des tâches sous différentes contraintes aux DAML-S composition services est menée.

Pour le problème traité dans cette thèse, la formation d'une coalition a pour but la recherche des solutions itinéraires sous forme de coalition répondant aux requêtes d'utilisateurs.

IV.3.2 Formulation du problème de coalition

Avant de détailler les spécification du problème, nous proposons une formulation du problème général de coalition entre les agents.

IV.3.2.1 Description du problème de la formulation de la coalition

Considérons un ensemble d'agents $A = \{a_1, a_2, \ldots, a_n\}$ et un ensemble des tâches $K = \{k_1, k_2, \ldots, k_m\}$, les agents dans A sont capables d'exécuter les tâches dans K et peuvent négocier sur leur coalition afin de réaliser une combinaison de tâches dans K [Aknine and Shehory, 2005].

Chaque agent a_i dispose d'un ensemble de ressources $s_{a_i} = \{s^1_{a_i}, s^2_{a_i}, \ldots, s^p_{a_i}\}$ qui se compose de p différents types de ressources. Ces agents ont la même capacité pour réaliser une tâche mais avec des résultats non identiques. Si a_i est capable d'exécuter une tâche k_j, la ressource s_{k_j} de $s^l_{a_i}(1 \leq l \leq p)$ est demandée. Afin d'exécuter une liste de tâches qui sont semblables avec a_i en même temps, le même agent a_i est demandé. La ressource disponible pour un agent est toujours limitée. Certaines tâches forment une union qui ne peut pas être divisée. Ainsi, ces tâches dans l'union doivent être exécutées dans un ordre bien défini. Plusieurs agents vont combiner et mettre leurs ressources en commun afin de réaliser cette union de tâches ensemble.

L'ensemble des ressources demandées pour exécuter le groupe g de m tâches est l'agrégation des ressources demandées pour chaque tâche individuelle dans g, c'est à dire, $S_g = \{s_{g_1}, s_{g_2}, \ldots, s_{g_m}\}$ avec s_{g_l} la ressource nécessaire pour la tâche $g_l(\forall l, 1 \leq l \leq m)$.

L'agent a_i correspond à la route R et ceci peut être interprété de la façon suivante : un voyage en la route R est considéré comme une tâche k qui peut être exécutée par l'agent a_i. La formation d'une coalition pour la composition des itinéraires devient alors la coalition des agents pour réaliser les tâches correspondantes. L'objectif d'un agent est de trouver les coalitions auxquelles il appartient, de tel sorte que l'utilité globale soit maximale. À l'issue du processus de formation d'une coalition, un ensemble d'agents vont être combinés pour exécuter l'ensemble des tâches, en même temps, ce dernier forme un itinéraire entier pour une requête.

Le but d'employer une telle méthode dans ce contexte est de trouver des bonnes solutions du problème en se basant sur les préférences des usagers exprimées avec le système d'information.

D'une manière globale, les tâches et les agents forment deux ensembles pour les requêtes lancées simultanément. Il s'agit d'une allocation des ressources des agents aux tâches. La formation d'une coalition est appliquée pour former les itinéraires pour toutes les requêtes. Comme un agent dispose des ressources, il peut être utilisé à plusieurs reprises et participe aux coalitions dans la limite de ses ressources.

D'un point de vue plus détaillé, au niveau d'une requête, les agents visent à former un ensemble et se regroupent pour combiner leurs ressources afin d'exécuter un ensemble de tâches. Lorsque la coalition est formulée et prise définitivement, les ressources consommées doivent être prises en compte et la quantité des ressources restantes est mise à jour.

L'intérêt de la coalition des agents réside dans le fait qu'elle permet encore de profiter de l'alliance entre l'optimisation et les SMA pour générer des itinéraires comodaux optimisés. Comme chaque morceau de route est géré par un Route Agent, ce dernier est responsable de gérer les véhicules sur la route. Un processus d'optimisation locale permet d'attribuer les véhicules (ressources) aux usagers selon leurs besoins, ensuite la coalition des agents permet de relier les morceaux de routes. Pour chaque coalition, un processus d'évaluation sur l'ensemble des possibilités avec des véhicules attribués permet d'obtenir des itinéraires optimisés selon les critères définis. Les agents coopèrent ensemble et forment des combinaisons de routes qui seront proposées comme solutions aux différents usagers. En applicant l'approche de la coalition, des requêtes simultanées et nombreuses peuvent être prises et traitées efficacement, même en cas de perturbation où les ressources de transport sont insuffisantes.

IV.3.2.2 Définitions

Par la suite nous détaillons les définitions des comportements et du protocole d'interaction des agents pour la formation des coalitions.

Route Agent : Cet agent est associé à une section de route avec ses informations détaillées.(cf. section IV.2.2.5)

Le Route Agent porte les informations suivantes : les voyageurs sur cette route, les informations de la route (l'origine, la destination, la distance etc.), les véhicules sur cette route et le résultat d'affectation de l'étape précédente.

Coalition : Un ensemble d'agents qui s'allient entre eux pour exécuter des tâches en vue de former un itinéraire.

formation d'une coalition : Le processus pour établir une coalition des agents.

Structure de coalition : Une structure de tâches qui conclut les tâches à réaliser et les relations entre eux. D'après la structure de coalition, les Route Agents se combinent et forme une coalition.

Dans ce travail, la structure de coalition se présente sous forme d'une séquence multimodale. En effet, il existe deux stratégies pour effectuer la coalition : de l'origine à

la destination et de la destination à l'origine. La première approche est adoptée avec les conditions temporelles de départ au plus tôt. En revanche, la seconde approche est utilisée en cas de contrainte sur l'arrivée au plus tard.

Comportement et protocole d'interaction : Un comportement décrit comment se comportent les agents et comment ils changent leurs rôles tout au long du processus de la coalition. Le protocole est respecté par les agents pendant le processus de la formation d'une coalition. En effet, le protocole se compose de plusieurs phases : le commencement de la coalition, l'ajout d'un nouveau membre à la coalition, et la fin de la coalition. À travers ces phases, les Route Agents acquièrent différents rôles.

Candidat : Le rôle d'un *Route Agent* quand il n'appartient à aucune coalition. Avant de s'engager dans la coalition, chaque agent joue le rôle de candidat et est prêt à en rejoindre une.

Ready_up : Le rôle d'un agent quand il est accepté pour rejoindre une coalition.

Membre : Le rôle d'un agent quand il a rejoint une coalition.

Leader : Le rôle d'un agent quand il lance une coalition.

Terminateur : Le rôle d'un agent lorsqu'il est dernier membre d'une coalition.

Un agent peut changer le rôle qu'il joue pendant la coalition, à titre d'exemple, un agent candidat peut devenir leader s'il porte un profil pour lancer une coalition, un agent membre peut rejouer le rôle de candidat quand il a quitté une coalition.

IV.4 Approche collaborative pour la formation des coalitions

Pour aller d'un point d'origine vers un point de destination, il existe plusieurs itinéraires possibles dans un réseau de transport comodal. Avec des préférences et critères exprimés par les usagers, nous obtenons un ensemble d'itinéraires attractifs qui servent comme structures de coalition.

Les structures de coalition sont des séquences multimodales dont les nœuds sont caractérisés par une fenêtre horaire compatibles aux contraintes temporelles à l'aide d'une étape de manipulation de fenêtre de temps(cf. section III.5.3.3). Pour un Route Agent qui représente un morceau de route, les véhicules compatibles aux conditions temporelles sont les offres et les requêtes d'itinéraires et les demandes. En effet, le résultat d'affectation entre les offres et les requêtes est obtenu avec un algorithme génétique, comme le

montre la matrice $\mathbf{M}_{(u,v)}^{[8h,12h]}$. Le Route Agent de cette matrice peut participer à toutes les formations de coalition où (u, v) fait part de la structure de coalition et la fenêtre de temps pour (u, v) est incluse dans [8h,12h].

$$\mathbf{M}_{(u,v)}^{[8h,12h]} = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \end{array} \begin{array}{c} \begin{array}{ccccc} V1_{(u_1,v_1)}^{[8h,9h]} & V2_{(u_2,v_2)}^{[9h,10h]}(2,3) & V3_{(u_1,v_1)}^{[10h,11h]} & V4_{(u_3,v_3)}^{[t_0,t_0+2h]}(2)^1 & V5_{(u_1,v_1)}^{[11h,12h]} \end{array} \\ \left(\begin{array}{ccccc} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array} \right) \end{array}$$

Pour une requête donnée, la formation d'une coalition se lance après que l'affectation des véhicules soit terminée pour chaque section de la route. L'algorithme global pour ce processus est Algorithme *ACS*.

Le traitement parallèle des requêtes, que ce soit d'une seule requête avec plusieurs Route Agents avec processus d'affectation ou de plusieurs requêtes reliées avec le même Agent Route, est effectué de façon synchrone entre tous les agents intervenants au niveau du processus global.

Algorithm 16 Algorithme de la coalition pour une Structure (**ACS**)

1: $g = \{g_1, \ldots, g_m\} \leftarrow$ Structure de coalition pour requête I
2: $A = \{a_1, \ldots, a_n\} \leftarrow$ Ensemble de Route Agents
3: // Étape 1 : Formuler la coalition
4: Sélectionner le *Leader* agent a_{g_1} pour la tâche g_1
5: **Pour** $i \in [2, m]$
6: $\quad a_{g_{i-1}}$ envoie proposition de la coalition aux agents éligibles dans A
7: $\quad$ **Si** g_i peut être exécuté par a_k avec la ressource s_{a_k}
8: $\quad\quad a_{g_i} \leftarrow a_k$ ▷ Selon la matrice d'affectation $\mathbf{M}$ de l'agent a_k
9: $\quad$ **Fin Si**
10: **Fin Pour**
11: // Étape 2 : Évaluer les solutions
12: Évaluer les différentes possibilités de la coalition $C_I = \{a_{g_1}, \ldots, a_{g_m}\}$ selon les critères
13: Déterminer la solution adoptée
14: Mettre à jour les ressources de chaque Route Agent

1. t_0 est le temps de départ de u avec véhicule V4.

IV.4.1 Formation des coalitions pour la recherche des solutions

Pour identifier le domaine de recherche, nous utilisons un graphe de modules qui est obtenu à partir du graphe hiérarchisé du réseau de transport considéré. Chaque Module Agent est responsable d'un module dans le SMA. Pour une requête, la chaîne de modules identifiée correspond donc à un ensemble de Module Agents selon lesquels les Route Agents sont rassemblés pour la formation des coalitions. L'organisation des Module Agents pour le graphe de la Figure III.9 est visualisée par la Figure IV.9.

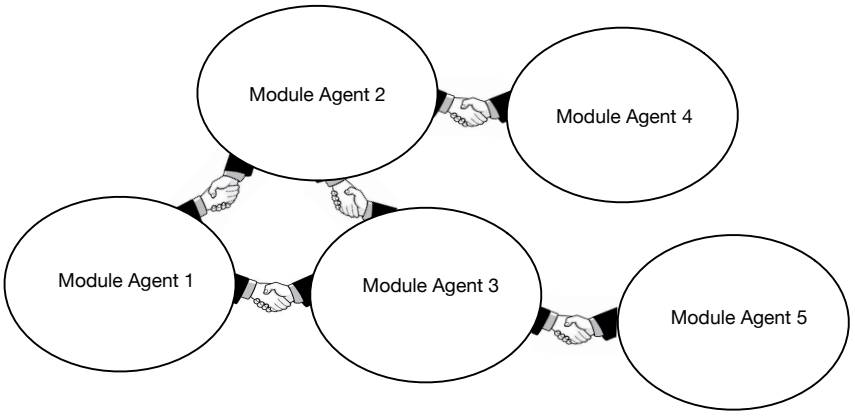


FIGURE IV.9: Les Modules Agents sur le plan de la coalition

Avec le graphe identifié pour calculer les itinéraires qui servent comme les structures de la coalition, les Route Agents sont présents au sein de chaque module du graphe hiérarchisé.

Une structure de la formation d'une coalition est l'ensemble des tronçons qui relient les extrémités et les points intermédiaires. En effet, il existe deux stratégies pour effectuer la coalition pour les agents. Le premier choix est de commencer la coalition par le point de départ et de continuer le processus avec chaînage avant. Le deuxième choix concerne d'initialiser la coalition par le point d'arrivée et poursuit la procédure avec chaînage arrière. Un agent peut appartenir dans plusieurs coalitions à la fois.

IV.4.2 Processus de formation des coalitions

Les agents d'un SMA sont des entités autonomes lorsqu'ils interagissent et communiquent entre eux. Par exemple, la négociation entre les agents est réalisée d'une façon autonome en cas de conflit durant le processus de la formation d'une coalition, autrement dit, les agents peuvent prendre la décision sans intervention venant de l'extérieur. Avec les conditions imposées, comme les préférences et critères temporels, les agents font les meilleurs choix. Pour guider la procédure de la coalition suivant les structures,

les protocoles d'interaction et les comportements des agents sont menées pour établir la coopération entre ces entités.

La formation d'une coalition pour ce problème spécifique consiste en plusieurs phases décrivant le processus : commencement, ajout d'un nouveau membre et achèvement.

IV.4.2.1 Commencement de la formation d'une coalition

La formation d'une coalition est initialisée par un agent *leader*. À partir du moment où un agent envoie une demande d'une formation de coalition, le processus est lancé par cet agent *leader*. En effet, l'agent *leader* représente une section de route ainsi qu'un sous-itinéraire. Tous les autres agents disponibles qui disposent d'un profil permettant de répondre à cette demande publiée adoptent le rôle de *ready_up*. Ce dernier va répondre aux demandes de façon réactive [Kowalczyk et al., 2006].

Les agents sont capables d'envoyer différents types de propositions aux autres agents et reçoivent les réponses réciproquement. Trois termes sont utilisés pour illustrer ces comportements :

- envoyer les propositions ;
- recevoir les messages ;
- répondre aux messages.

Le comportement de la demande d'une coalition par un agent *leader* est visualisé par la Figure IV.10.

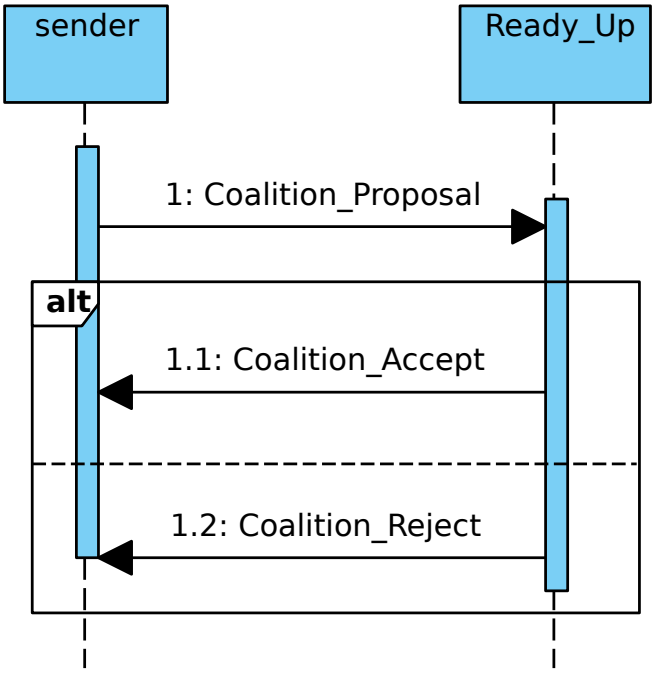


FIGURE IV.10: La conversation pour proposer une coalition

La coalition suit l'ordre interne des morceaux de route au sein des structures d'une coalition. Ce dernier est défini préalablement.

Pour la section de route S_m, représentant le morceau de route (S_{mO}, S_{mD}), S_{mO} et S_{mD} sont l'origine et la destination respectivement. D'après la structure de la coalition, la section suivante est S_{m+1}.

en plus des contraintes imposées par les utilisateurs, la condition importante sur le temps est l'ordre chronologique : $t_{LastArrival} < t_{Departure}$.

L'émetteur transmet une proposition de la coalition aux agents éligibles qui satisfont les conditions précédentes sur le temps et le transfert. Les agents éligibles se nomment ready_up agents aussi.

Si la proposition de la coalition est acceptée par un agent ready_up, ce dernier va renvoyer une confirmation de l'acceptation à l'émetteur. En même temps, l'agent ready_up devient un membre agent de la coalition. Dans le cas contraire, l'agent renvoie un rejet de la coalition comme réponse. Cet agent reprend son rôle de candidat.

Pour pouvoir devenir un membre agent de la coalition, les conditions suivantes sont exigées :

— Les nœuds de la route coïncident avec les nœuds correspondants dans la structure de la coalition.
— Le voyageur est bien affecté à un véhicule sur cette route dans la matrice d'affectation.
— Le temps d'arrivée du véhicule n'est pas plus tard que le temps de départ pour la route suivante.

Après que le statut de membre est confirmé, les ressources des agents route se mettent à jour.

IV.4.2.2 Ajout d'un nouveau membre à la coalition

L'activité de trouver le membre suivant de la coalition fait référence à l'ajout d'un nouveau membre, ceci est décrit dans la Figure IV.11.

Sans perte de généralité, nous supposons que la coalition se déroule du point de départ vert la destination.

En ce qui concerne l'identification des agents *leader* d'une coalition, il faut faire référence à la structure de la coalition. En cas d'une coalition en chaînage avant, l'agent représentant

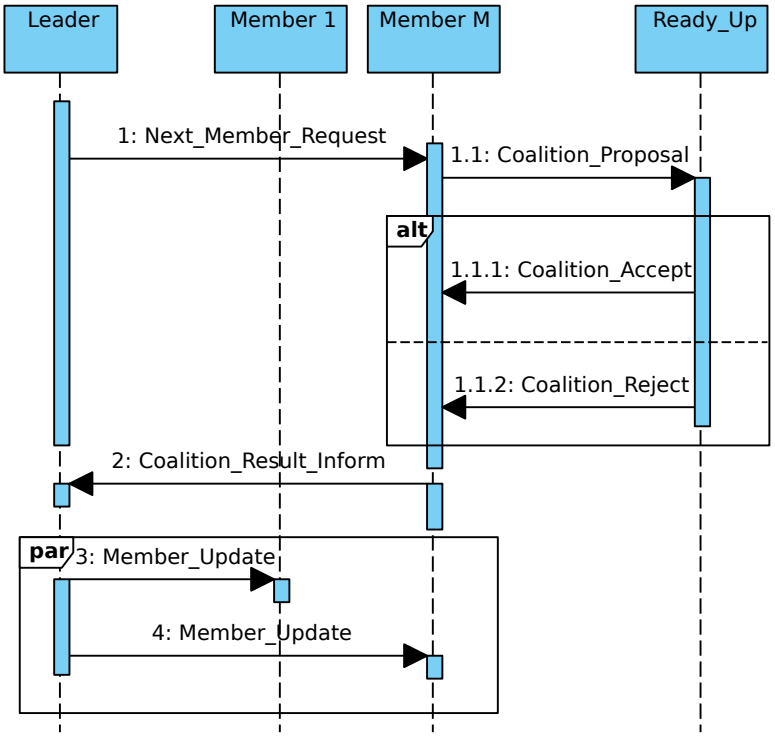


FIGURE IV.11: Ajout d'un nouveau membre à la coalition

la première section de route joue ce rôle de *leader*. Dans le cas contraire, l'agent représentant la dernière section de route est pris comme *leader* de la coalition.

Pour insérer un nouveau membre à la coalition, l'agent *leader* transmet la demande d'ajout d'un nouveau membre à ses membres appartenant à la même coalition. En suite, l'agent membre concerné envoie la proposition de la coalition aux *candidats* qualifiés en suivant la structure de la coalition. L'agent *ready_up* va y répondre d'une manière proactive. La conversation pour la proposition de la coalition est répétée jusqu'au moment où un nouveau membre est intégré. Après avoir reçu le message de confirmation de prendre un agent *ready_up* comme un *membre*, l'agent *leader* informe tous les agents de ce nouveau membre.

Lorsque un agent est ajouté comme un nouveau membre de la coalition, on ajoute une section aux sous-itinéraires. Parmi les principaux rôles de l'Assist Agent dans ce contexte : la mise à jour des sous-itinéraires avec un nouveau morceau de route que le nouveau membre porte. Ayant validé l'optimalité des sous-itinéraires (non dominé par les autres), ceux-ci sont gardés avec une structure de données appropriée. C'est l'Assist Agent qui porte ces sous-itinéraires à partir desquels les solutions de l'étape suivante sont obtenues en intégrant le nouveau membre de la coalition. Cette structure de données est gardée en mémoire durant la formation d'une coalition.

IV.4.2.3 Achèvement de la formation d'une coalition

L'ajout du dernier membre *terminateur* à une coalition signifie la fin de la procédure. Le dialogue pour terminer la formation d'une coalition est illustré par la Figure IV.12.

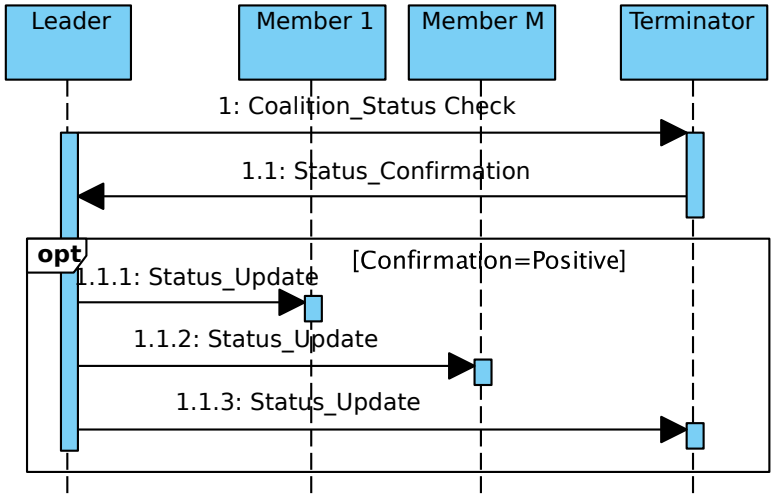


FIGURE IV.12: Conversation pour terminer la formation d'une coalition

L'agent *leader* de la coalition envoie une demande au dernier membre afin d'effectuer la vérification du status de la coalition. Après avoir vérifié les conditions, celui-ci y répond avec un message contenant le status de la coalition. Si la réponse est une confirmation de la fin de la coalition, le dernier membre porte le rôle d'agent *terminateur*. En même temps, tous les membres agents sont informés de la terminaison de la coalition dont ils font partie.

Pour choisir un terminateur de la coalition, les conditions suivantes doivent être satisfaites :

— l'extrémité du coté arrivé de la route représentée par l'agent terminateur est identique à celle de la destination de la structure de la coalition ;
— la condition sur le temps d'arrivée : $t_{arrival} < t_{arrivalBefore}$ où $t_{arrival}$ signifie le temps d'arrivée réel et $t_{arrivalBefore}$ signifie le temps d'arrivée imposé.

Durant tout le processus de formation d'une coalition, chaque étape de la combinaison est évaluée. Les préférences des usagers peuvent varier d'un utilisateur à un autre. Il peut exister plusieurs résultats finaux éligibles à la fois, donc, l'optimisation devient nécessaire pour prendre la solution la plus convenable.

La formation d'une coalition prend fin quand une condition parmi les suivantes est remplie :

— la coalition comprend un agent *terminateur* ;

— l'agent *leader* abandonne la coalition ;
— la limite de temps prédéfinie est atteinte.

À l'issue d'une formation de coalition pour une structure de la coalition de la requête I, on obtient $C_I = \{a_I^1(s_1), \ldots, a_I^n(s_n)\}$ qui est une coalition permettant d'avoir un ensemble d'itinéraires optimaux au sens de Pareto. Lorsque toutes les formations de coalition d'une requête s'achèvent, l'union de tous les itinéraires non-dominés fait l'ensemble de solutions. En effet, comme nous l'avons expliqué, cet ensemble de solutions est calculé avec les résultats de la dernière étape sauvegardés par l'Assist Agent. Une fois que la formation d'une coalition se termine et qu'une solution est prise par l'usager, cela entraîne une modification automatique de la situation générale comme la disponibilité des véhicules pour en assurer la cohérence et l'actualisation.

Au niveau du système, les comportements des agents sont illustrés par le diagramme de séquence de la Figure IV.13.

IV.4.2.4 Compromis de la formation d'une coalition

Dans certains cas, les conditions imposées par l'usager peuvent entraîner la fin de la coalition sans solution valable. Par exemple, la plage horaire entre le départ au plus tôt et l'arrivée au plus tard pour la requête n'est pas assez large pour avoir un arrangement d'itinéraire faisable entre l'origine et la destination. Comme dans la société humaine, les gens tendent à compromettre leurs profits si tous les participants se mettent d'accord sur une solution. Dans le monde des agents, les agents peuvent faire un compromis en vue d'obtenir des solutions et de les fournir aux usagers.

Dans notre problème, les causes pouvant conduire au manque de solution sont variées. Nous citons dans la suite celles qui sont principales :

— le critère mis sur le mode de transport est trop sévère ;
— la plage horaire est excessivement limitée.

IV.4.3 Alliance de l'optimisation et SMA pour une méthode efficace

Afin d'avoir des résultats optimisés pour les problèmes combinatoires tel que les problèmes du plus court chemin ou ceux des affectations, des approches d'optimisation sont utilisées et implémentées comme par exemple les algorithmes génétiques.

Les SMA sont dotés d'intelligence artificielle distribuée. Compte tenu du fait que les données des sous-systèmes intégrés (les sources informatiques) sont distants, nous pouvons profiter de l'intelligence artificielle distribuée de SMA pour notre problème. Les

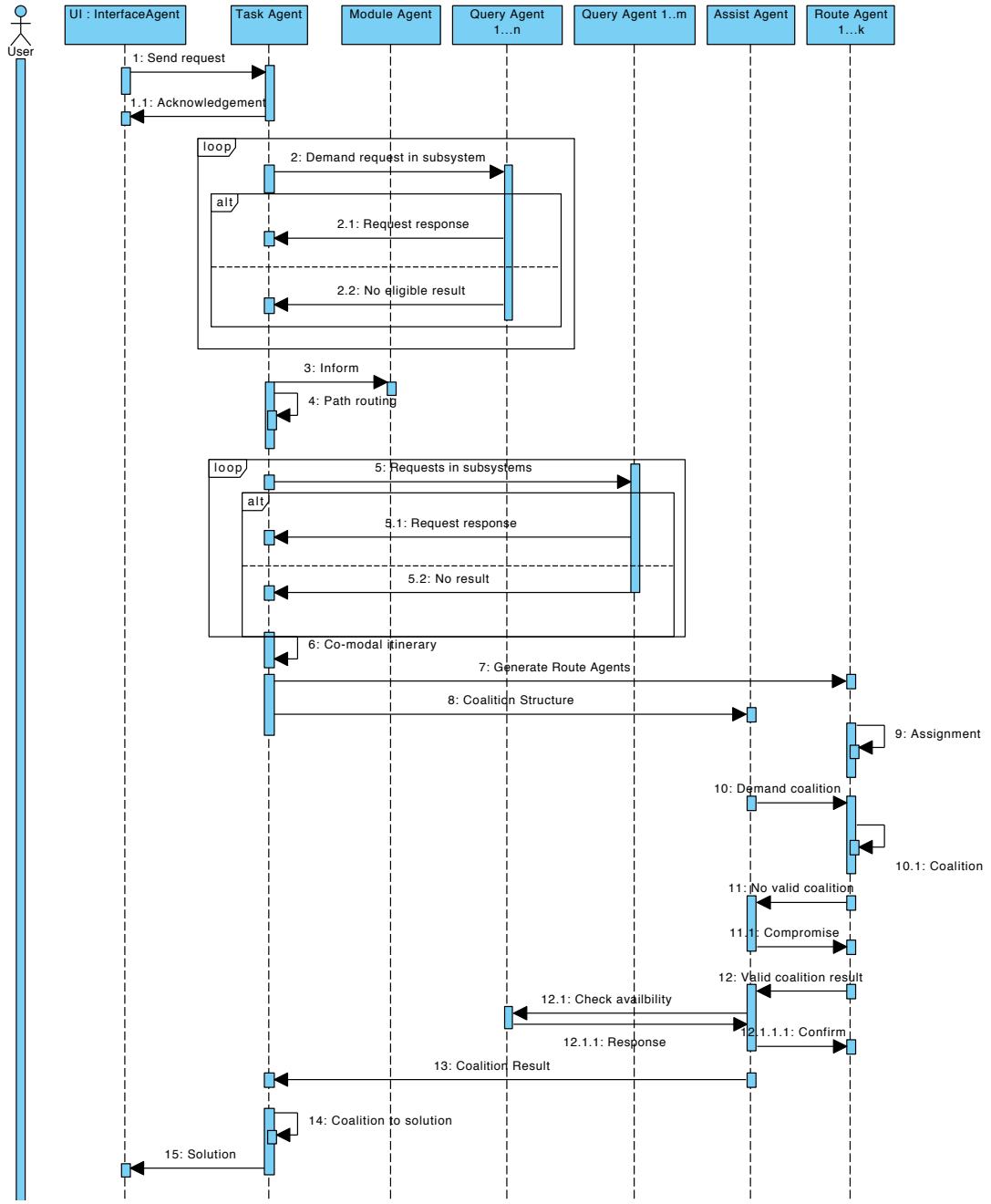


FIGURE IV.13: Illustration du diagramme de séquence au niveau du système

processus de traitement des requêtes d'itinéraires pourront donc être répartis et traitées en parallèle.

Nous avons appliqué l'approche basée sur le paradigme agent, couplée à des méthodes d'optimisation telles que celles reposant sur des algorithmes génétiques. Cette utilisation conjointe des approches d'optimisation et de SMA est intéressante, surtout en présence de perturbations sur le réseaux de transport et/ou devant l'arrivée simultanée d'un grand nombre de requêtes. Cette méthode permet de combiner les avantages de l'optimisation et des SMA. Le processus d'affectation sur un morceau de route est considéré comme une optimisation locale. Cependant, la coalition des Route Agents à l'aide des Assist Agents permet d'assembler des résultats à l'issue des optimisations locales et propose des combinaisons de routes optimales. La méthode est plus efficace que l'optimisation ou SMA seuls sur ces problèmes.

IV.5 Système d'information orienté agent

Au sein de notre système, les entités logicielles du SMA mises en places ont pour mission d'effectuer les tâches dans une synchronisation totale, évitant toute sorte de conflit et profitant d'une intelligence distribuée. Pour ce faire, les canaux de communication entre les agents sont ainsi établis. À l'aide des diagrammes d'activité et des diagrammes de séquence, nous avons présenté les comportements des agents et les interactions inter-agent.

Dans le cadre de la mise en place d'un système basé sur des agents communicants et de la coalition entre les agents, nous nous focalisons sur le fonctionnement du système au niveau du flux de communication.

IV.5.1 Communication entre les agents

Le système proposé s'appuie sur le principe de la distribution des missions pour réduire la complexité du problème. Outre que les agents se partagent les tâches, les coalitions entre les Route Agents souscrivent également à ce contexte. Comme le monte la Figure IV.13, les différents agents coopèrent et exécutent des activités distinctes qui dépendent de leurs missions et des objectifs spécifiques afin d'aboutir à un objectif global au niveau du système :

— Les Interface Agents ont pour mission de transmettre les informations collectées aux Task Agents pour leurs traitements et de recevoir les résultats obtenus grâce à la coopération inter-agent ;

— Les Task Agents ayant pour plusieurs missions d'établir la coopération avec les autres agents tout au long du processus ;
— Les Assist Agents coopèrent avec le Task Agent et guident la coalition entre les Route Agents afin de trouver les combinaisons optimales de services de transport ;
— Les Route Agents représentant les morceaux de routes ainsi que les services de transport desservis sur ces derniers et forment les coalitions correspondantes aux meilleurs itinéraires pour répondre aux attentes des usagers.

IV.5.2 Communication avec l'extérieur

Étant le support de communication des agents, le SMA mis en place favorise aussi continuellement la communication avec l'environnement où est déployé notre système de transport comodal.

Les Interface Agents ont pour objectif de collecter toutes les informations saisies par les usagers et d'assurer la bonne configuration et la synchronisation des données sur celles-ci ; de plus, ils sont aussi responsables de communiquer les résultats des requêtes d'itinéraire aux utilisateurs. Les Query Agents sont concernés par des sous-systèmes intégrés en consultant les données ; lorsque les perturbations se présentent au sein d'un sous-système, le Query Agent en est informé.

IV.6 Conclusion

Dans ce chapitre, nous avons présenté le système multi-agents adopté qui intègre les approches proposées. Cela nous permet de profiter de la combinaison entre le paradigme agent et les méthodes d'optimisation en vue d'avoir un système plus performant et plus efficace. Puis l'approche de la coalition entre les agents est établie pour la combinaison des routes afin d'obtenir les solutions optimisées même en cas de perturbation dans le réseau de transport.

Dans ce chapitre nous avons présenté notre approche pour le problème défini, dans le chapitre suivant, nous présentons son implémentation. Différents scénarios d'application ainsi que des détails du déploiement seront fournies.

Chapitre V

Implémentation et déploiement de la solution proposée

Sommaire

V.1 Introduction

Dans ce chapitre, nous présentons notre solution en mettant l'accent sur les domaines d'intérêts scientifiques en premier lieu. Par la suite, nous exposons les différentes plateformes SMA existantes et qui sont susceptibles d'être utilisées pour le déploiement de notre application, nous présenterons celle que nous avons choisie et nous donnerons les raisons de notre choix. Puis nous présentons le prototype du système proposé ainsi que les scénarios de tests utilisés pour valider notre approche.

V.2 Une plateforme de mobilité avancée optimisée

Pour présenter notre système sur le plan pragmatique, nous évoquons dans cette section les aspects liés à notre méthode de résolution.

V.2.1 Une solution dynamique

L'évolution du covoiturage à été assez rapide avec l'émergence des plateformes internet qui dans la plupart des cas se contente de mettre en relation les offres et les demandes. Le covoiturage et l'autopartage sont deux formes de la nouvelle mobilité partagée qui véhiculent "des valeurs positives", et prennent aujourd'hui une place importante dans cette économie dite collaborative.

Face aux besoins très variés, nous nous sommes intéressés à combiner les services de transport en commun et ceux des véhicules partagés afin de fournir aux usagers des solutions pouvant répondre à leurs critères et exigences. Nous nous sommes placés dans le contexte où les aspects temps réel, les perturbations de transport subies occasionnellement, la disponibilité des véhicules partagés sont pris en compte pour considérer la dynamique des environnements de transport.

V.2.2 Domaines d'intérêts scientifiques

Nos travaux profitent de l'alliance de deux domaines, celui de l'optimisation et celui du paradigme agent afin de trouver une solution optimale au problème traité. Ce choix est également dicté par le fait que ces travaux sont réalisés au sein d'une équipe de recherche où les thèmes de SMA et d'optimisation sont importants. La mise en place de la combinaison de l'optimisation et de SMA est à nos yeux un choix judicieux. De plus, plusieurs domaines scientifiques sont utilisés pour élaborer notre solution. Ils sont utilisés tout au long de la démarche de résolution afin d'aboutir à la réalisation des objectifs fixés. Nous citons principalement :

- **Recherche opérationnelle** : théorie des graphes, problème d'affectation, ordonnancement en temps réel, problème de plus court chemin, optimisation multicritère, méthode exacte, métaheuristique, algorithme génétique
- **Aide à la décision** : intelligence artificielle distribuée, optimisation distribuée, implémentation de l'alliance entre SMA et optimisation
- **Transport** : transport multimodal, véhicule partagé, perturbation

V.3 Programmation orientée agent

Après avoir fixé l'architecture du système pour l'élaboration du système d'information de transport, nous nous intéressons à la sélection de l'outil pour le développement et la mise en place de ce dernier. En effet, nous avons présenté dans les chapitres précédents nos principaux sujets en soulignant l'union des méthodes d'optimisation et les systèmes multi-agents. Nous avons conçu une architecture multi-agents spécifique à notre problème donnant lieu à un ensemble d'agents communicants. À partir des caractéristiques spécifiques aux agents et aux SMA qui nécessitent des différents outils pour le développement du système basé sur ce paradigme, la programmation orientée agent symbolise une progression de celle orientée objet. Nous allons, dans cette section, mettre en évidence la relation entre ces deux types de programmation, ainsi qu'argumenter notre choix d'outil parmi les plateformes de développement de SMA existantes.

V.3.1 Processus de développement : agent vs objet

Afin de pouvoir comparer agent et objet, nous présentons dans un premier temps ces deux concepts. En programmation structurée, un programme consiste en différentes procédures et structures de données indépendantes de ces procédures. Autrement dit, l'accès aux données peut être réalisé directement. Avec le paradigme objet, différents objets sont mis en place au sein d'un programme. Chaque objet allie des données à des méthodes qui agissent uniquement sur les données de l'objet. Il convient de noter que les méthodes jouent toujours le rôle d'interface par laquelle sont manipulées les données d'un objet. Étant donné que les objets sont passifs, ils sont prêts à répondre à l'appel, au lieu d'être capable de décider de leurs propres fonctionnements.

Cependant, les agents d'un SMA sont en possession de leurs propres objectifs et des plans d'exécution. La plus grande différence entre un agent et un objet réside dans le niveau d'autonomie du choix d'exécuter ou pas une méthode. Dans le cas d'un objet, l'objet appelant la méthode est responsable de la prise de décision. Au contraire, l'agent qui reçoit l'appel prend la décision au sein d'un SMA.

De plus, en incluant toutes les notions relatives au concept objet, le concept agent intègre les fonctionnalités de communications via l'établissement de collaborations ou encore négociations entre les agents au sein du système. Les agents peuvent être créés ou détruits dynamiquement, selon les besoins. Notre approche profite de toutes ces avantages par la proposition d'une structure multi-agents en concordance avec la stratégie de résolution du problème ciblé.

V.3.2 Plateforme de développement multi-agents

La plateforme multi-agents, caractérisée par des modèles de mise en place, des formalités d'interaction et de communication, est le support permettant de concevoir, créer, gérer et expérimenter des SMA. Avant de justifier notre choix spécifique, il convient de décrire des principales plateformes multi-agents actuelles.

V.3.2.1 Les plateformes de SMA

Nous allons présenter les plateformes multi-agents existantes pour la mise en œuvre du système que nous avons conçu. Parmi toutes les plateformes candidates, nous en détaillons quelques-unes et en choisissons une comme la plateforme de développement.

— CORMAS. CORMAS est une plateforme libre (ou *open source*) dédiée au développement de systèmes multi-agents, basée sur SmallTalk comme langage de programmation. En tant que plateforme spécialisée, celle-ci est destinée aux problématiques de recherche scientifique de développement et de négociation entre les acteurs.

— Zeus. C'est une autre plateforme avec laquelle le développement des systèmes collaboratifs peuvent être conduit en quatre étapes : analyse, conception, réalisation et support à l'exécution. Malgré que elle présente davantage de points forts pour la définition des rôles sur l'élaboration, l'organisation et la coordination, la difficulté de la maîtriser de la complexité empêche sa mise en œuvre.

— Madkit. Madkit, abréviation de Multi-Agent Development Kit, est une plateforme basée sur un modèle organisationnel de type (Agent/Groupe/Rôle). Le fait qu'elle est basée sur les principes du paradigme agent permet des applications rapides pour les problèmes distribués.

— JADE (Java Agent Development Environment)[1] . JADE est une plateforme de développement des applications orienté agent fondée sur le langage Java. En tant que support de développement, elle facilite la mise en œuvre d'un système multi-agents répondant aux spécifications de *FIPA* [2] par l'intermédiaire de plusieurs outils, et utilise donc le langage *FIPA ACL* pour établir des liens de communication entre les agents pour transmettre des messages. Jade est un projet open source offrant des composants de développement des agents légers.

1. `http://jade.tilab.com`
2. Foundation for Intelligent Physical Agents (`http://www.fipa.org`)

V.3.2.2 JADE : comme choix de la plateforme de développement

Dans l'optique de réaliser l'implémentation, nous décidons d'adopter la plateforme JADE comme outil de développement.

Chaque instance de JADE qui s'exécute sur un ou plusieurs postes est nommée comme conteneur (*container* en anglais). Il existe un et un seul conteneur de type *conteneur principal*. Cependant, le reste des conteneurs doit être déclaré au sein de conteneur principal. La totalité de ces conteneurs constitue la plateforme multi-agents. Dans le cadre d'une conformité à la norme *FIPA*, le conteneur principal héberge trois modules composants de JADE, à savoir :

- *Agent Management System* (*AMS*) : Il est chargé de la supervision des agents : leur enregistrement, authentification, accès et utilisation du système.
- *Director Facilitator* (*DF*) : Ce module sert comme un registre centralisé d'entrées qui correspond souvent au service de pages jaunes.
- *Agent Communication Channel* (*ACC*) : Il a pour fonction de conduire les communications entre les entités.

De point de vue de la mise en place de nos travaux, JADE est un outil qui s'accorde avec nos objectifs et méthodes de résolution que nous avons décidée d'adopter. Avec l'usage de JADE, les normes essentielles pour la communication nous permettent d'élaborer notre approche avec les bonnes pratiques au niveau de la coordination et la synchronisation des actions des agents. c'est une plateforme distribuée pouvant être répartie sur plusieurs machines. En tant que processus Java, les agents sont implémentés en s'assurant que la communication sur la même machine est effectuée à l'aide des événements Java. Cette communication inter-agent est réalisée via les messages ACL qui sont des objets Java au niveau de la programmation pour des échanges légers. JADE offre une interface et un agent RMA permettant la gestion de la plateforme et des agents, telles que la création des nouveaux agents, la suppression des agents etc.

De point de vue de l'interopérabilité et compatibilité, nous ne dérogeons pas à la pratique au sein de notre équipe OPTIMA du laboratoire CRIStAL où les travaux précédemment élaborés et futurs utilisent une plateforme unifiée et générique.

De plus, JADE offre des outils de la gestion de la plateforme, parmi lesquels nous citons les plus intéressants :

- *Dummy Agent* : visualiser des messages et envoyer des messages aux agents de la plateforme et recevoir des réponses ;
- *Sniffer Agent* : visualiser l'enchaînement des messages et vérifier le processus de communication ;

— *Introspector Agent* : visualiser des messages envoyés et reçus et contrôler l'état des agents.

V.4 Prototype du système

L'architecture du système est visualisée sur la Figure V.1. Les données pour les différents modes de transports et les transit sont stockées avec la base de données. Les mapping APIs sont également intégrés pour afficher les cartes et les itinéraires obtenus. Au sein du SMA, trois couches appliquées à notre modélisation multi-agents se présentent : interface (vue), application et données. Cette représentation du modèle à couches concorde avec la modélisation agent proposée (cf. IV.2.1) : les Interface Agents pour la couche de vue, les Query Agents pour la couche de données et les autres types d'agents pour la couche d'application.

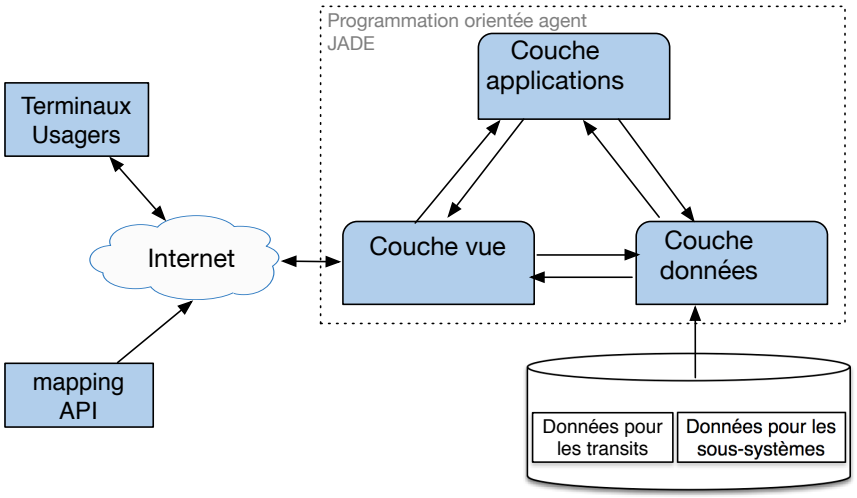


FIGURE V.1: Architecture du système

V.4.1 Choix des outils

V.4.1.1 Java

Afin de mettre en œuvre les approches et les algorithmes développés pour l'optimisation, nous faisons usage du langage de programmation Java. Ce dernier privilégie l'approche orientée objet de sorte que tout soit objet à part des types primitifs (comme nombres entiers, doubles, etc.). L'avantage principal de ce langage, et qui nous a insité à l'utiliser est que les applications développées en Java sont exécutables pour les SI (Systèmes d'Information) qui possèdent un JRE (Java Runtime Environment ou Environnement

d'exécution Java). Les deux constituants d'un JRE principaux sont la JVM (Java Virtuel Machine) qui interprète le code Java et le convertit en code natif et une bibliothèque standard avec laquelle les programmes sont développés.

V.4.1.2 L'API JDBC

Le Système de Gestion de Base de Données (SGBD) est utilisé pour stocker le jeu de données de notre application. Nous avons fait usage d'un Système libre de Gestion de Base de Données relationnelles (MySQL version 5.6.23) comme notre SGBD. Les bases de données sont accessibles à l'aide d'une interface spécifique pour Java. L'API JDBC (Java DataBase connectivity) permet à l'application d'accéder à la base de données où les informations requises pour le traitement sont stockées. Afin de pourvoir opérer sur la base de données, nous profitons d'un pilote JDBC nommé ODBC (version 5.3.4) adéquat qui convertit les appels de données en appels ODBC pour les exécuter par la suite.

V.4.2 Présentation du système

V.4.2.1 Interface pour la saisie des requêtes et la visualisation des résultats

L'utilisateur est amené à saisir un certain nombres d'information concernant sa requête, ceci est possible grâce à l'interface que nous avons mis à disposition :

- point origine(à choisir dans une liste) ;
- point destination(à choisir dans une liste) ;
- fenêtre de temps pour le départ : $[T_1, T_2]$ où T_2 est optionnel ;
- fenêtre de temps pour l'arrivée : $[T_3, T_4]$ où T_3 est optionnel ;
- préférence au niveau du type de services (les modes de transport autorisés) ;
- préférence au niveau des critères d'optimisation.

Dans le système développé, l'*Interface Homme-Machine* (IHM) permet de saisir les éléments des requêtes d'itinéraire. Un aperçu est donné dans la Figure V.2.

Notre système développé permet également de visualiser les résultats des requêtes d'itinéraire :

- sous forme d'une feuille de route : le bilan d'itinéraire, les détails de chaque morceau de route, les correspondances à effectuer etc ;
- sous forme cartographique : Avec les cartes, les solutions peuvent être visualisées. Pour fournir la possibilité d'afficher les itinéraires sur la carte, nous avons profité des mapping APIs. En effet, Google Maps est disponible pour afficher les itinéraires sur la carte avec Google Maps API.

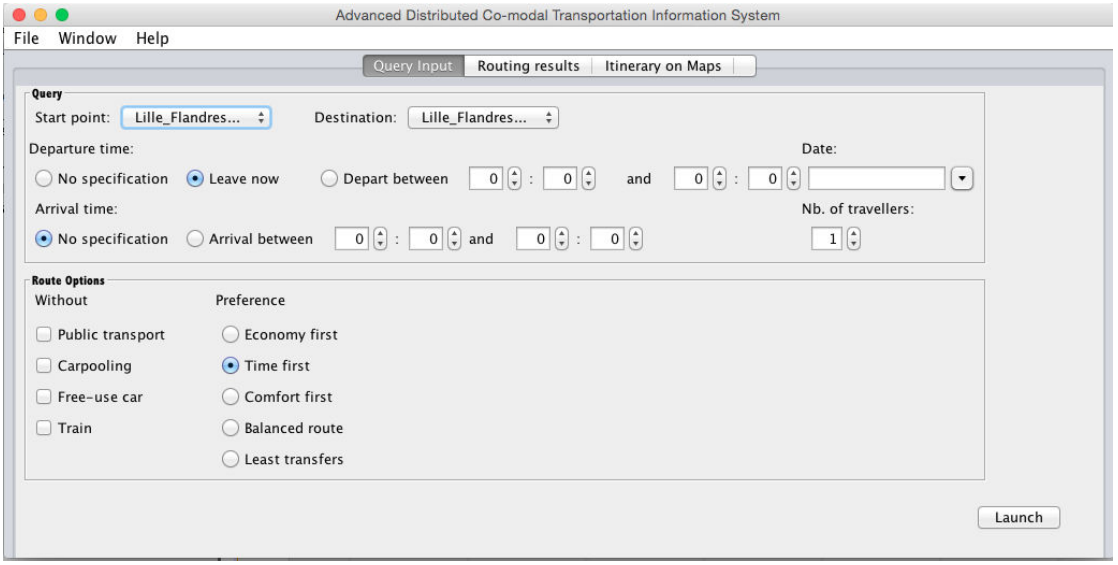


FIGURE V.2: Interface permettant la saisie des requêtes d'itinéraire

V.4.3 Système de transport intégré

Notre structure du système permet d'intégrer les services de différents opérateurs. Les approches présentées dans les chapitres précédents consistent à chercher les itinéraires à travers les sous-systèmes en respectant les critères imposés.

Nous décidons de lancer les scénarios dans un réseau relativement grand où se trouvent plusieurs fournisseurs de service. Parmi ces derniers, différents moyens de transport se présentent :

— au niveau de type de véhicule : métro, bus, train, covoiturage, autopartage et vélo-partage ;
— au niveau de type de service : collectif et privé ;
— au niveau de type de fonctionnement : service urbain et service extra-urbain.

V.4.4 Jeu de données pour les scénarios

V.4.4.1 Modèle de données

Pour décrire le réseau de transport, en particulier le coût de trajet, le temps de parcours ou la distance parcourue, on a besoin de plusieurs types de données de différentes natures :

— des informations logiques permettant de construire des liens entre les objets ;
— des renseignements sur la correspondance pour la construction de *table de transit* ;
— des informations géographiques pour la représentation des objets sur la carte ;
— des informations horaires ;

— des renseignements sur le coût de trajet.

V.4.4.2 Collecte des données et la génération de graphe hiérarchisé

Plutôt que “créer” des données, nous avons récupéré celles des lignes principales et représentatives de certains opérateurs de transport de différents services pour les scénarios de la simulation.

Avec les données que nous avons récupérées, le système offre aux usagers trois types de service de transport, à savoir ; les transports publics, les véhicules en libre service et le covoiturage. Une synthèse de ces services intégrés est fournie dans le Tableau V.1.

Tableau V.1: Les sous-systèmes intégrés

Numéro	**Sous-système**	**Opérateur**	**Région**	**Modes**	**Type de fonctionnement**
1	Transpole	Transpole	Lille Métropole	Métro, Bus, Tramway, Vélo	Public
2	DK’BUS	DK’BUS	Dunkerque	Bus, Vélo	Public
3	sitac	sitac	Calais	Bus	Public
4	Ligne-BCD	Véolia	Dunkerque, Calais	Bus	Privé
5	Tadao	Tadao	Lens	Bus, Tramway	Public
6	évéole	évéole	Douai	Bus	Public
7	Transvilles	Transvilles	Valenciennes	Bus, Tramway	Public
8	Lilas-autopartage	Lilas	Lille Métropole	Autopartage	Privé
9	Covoiturage	Covoiturage	Global	Covoiturage	Privé
11	SNCF	SNCF	Global	Train	Public

On classe les transferts en deux niveaux. Le graphe de modules est visualisé par la Figure V.3.

Nous montrons les quantités des arcs et des nœuds pour les graphes des différents réseaux de service dans le Tableau V.2.

V.4.5 Tests et scénarios d’exécution

Nous visons dans ce travail la recherche et la composition de services dans un contexte de transport comodal et distribué. Pour montrer cette faisabilité, il faut accéder aux différents systèmes d’information pour l’aide au déplacement de différents services de

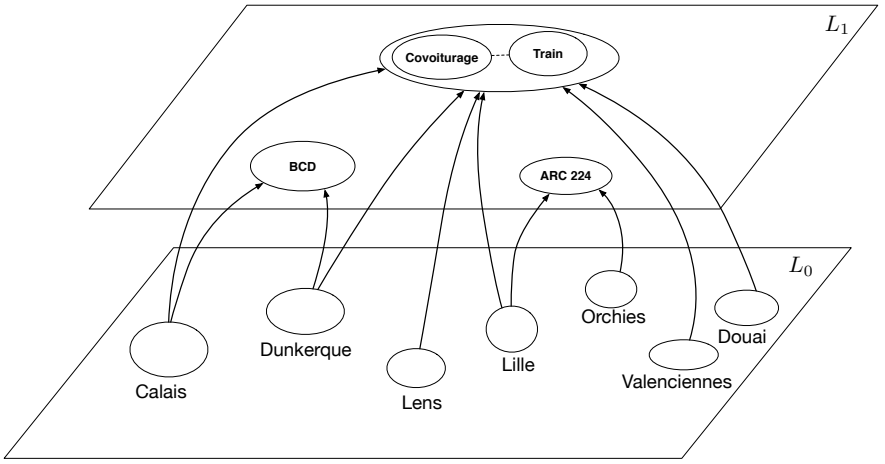


FIGURE V.3: Le graphe de modules pour les scénarios

TABLEAU V.2: Réduction des graphes pour différents réseaux de transport

Réseau	Graphe du réseau			Graphe de transfert		
	Nœuds	Arcs	Modes	Nœuds	Arcs	Composants
Lille	2106	6205	5	192	617	5
Dunkerque	523	1409	2	47	165	2
Calais	641	1907	2	62	177	2
Douai	514	1476	2	52	145	2
Valenciennes	612	194	2	67	178	2
Lens	778	2364	2	82	234	2
BCD	100	100	1	10	10	1
Train	30	50	1	30	50	1
Covoiturage	35	105	1	35	105	1

transport. Or, les conditions pour ce faire ne sont pas présentes. Nous décidons donc de réaliser la simulation sur des données théoriques qui sont conformes au contexte réel des modes de transport utilisés.

Dans cette section, nous mettons en exergue le fonctionnement du système élaboré par des scénarios d'exécution. Avec les jeux de données que nous avons collectés, nous donnons des résultats générés pour plusieurs scénarios réalistes avec des préférences variées. Nous allons remettre les résultats sous la forme de feuille de route et les visualiser sur les cartes. Nous utilisons par la suite les abréviations des modes de transport qui sont listées dans le Tableau V.3.

Le premier scénario porte sur un exemple d'une seule requête traitée en même temps. Le deuxième scénario porte sur le processus de traitement de plusieurs requêtes simultanées et en présence de perturbations qui affectent les réseaux de transport. Le troisième scénario s'intéresse à l'étude des performances du système face à la montée en charge des requêtes utlisateurs (grand nombre d'utilisateurs).

Tableau V.3: Les modes de transport

Mode de transport	Abréviation
Métro	M
Bus	B
Train	T
Marche à pied	W
Autopartage	A
Covoiturage	C
Vélo-partage	V

V.4.5.1 Scénario 1 : Une seule requête

Dans ce scénario, on considère le cas d'une seule requête. Le voyageur lance une requête d'itinéraire allant de 4 Cantons (Villeneuve d'Ascq) vers Impressionniste (Dunkerque). C'est l'Interface Agent qui assure l'acquisition comme il est spécifié dans la description de son fonctionnement (cf. section IV.2.2.1). À l'aide d'interface qui permet la saisie des requêtes, les éléments suivants sont pris en compte :

— point de l'origine : 4 Cantons ;
— point de la destination : Impressionnistes ;
— fenêtre de temps pour le départ : $T_1 = 7$h comme le temps de départ au plus tôt ;
— fenêtre de temps pour l'arrivée : $T_4 = 10$h comme le temps d'arrivée au plus tard ;
— préférence au niveau du type de service : tous les modes de transport acceptés ;
— préférence au niveau de critères d'optimisation : le coût du trajet et le temps de parcours.

La première étape après la réception de la requête est d'identifier le domaine de recherche, à savoir une chaîne de modules. Comme le montre la Figure V.4, la chaîne de modules trouvé pour cette requête est $\mathscr{M}(10)$-$\mathscr{M}(6)$-$\mathscr{M}(2)$. Le module $\mathscr{M}(10)$ comprend trois composants : C_1 (service vélo-partage de V'Lille), C_2(service métro, tramway et bus de Transpole) et C_3 (service autopartage de Lilas). Le module $\mathscr{M}(6)$ couvre deux composants C_4(service covoiturage) et C_5 (service train de SNCF). Deux composants coexistent dans le module $\mathscr{M}(2)$: C_6(service vélo-partage de DK'BUS) et C_4(service bus de DK'BUS).

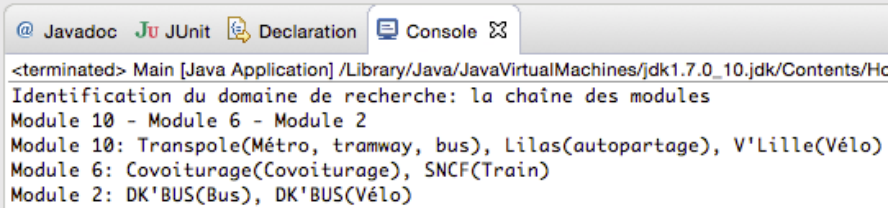


Figure V.4: Identification du domaine de recherche

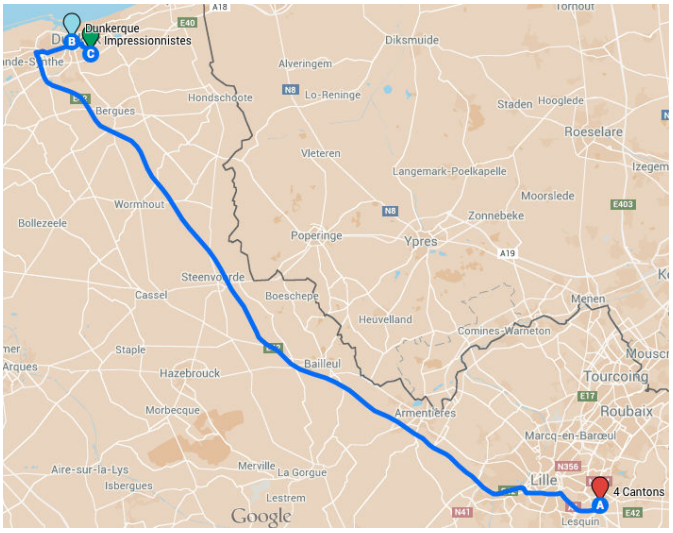
Par la suite, trois trajets attractifs sont obtenus dans le domaine de recherche identifié pour cette requête avec notre système. Afin de faciliter la notation, nous désignons les nœuds suivants : 4 Cantons, Lille Flandres, Porte des Postes, Dunkerque, Impressionnistes comme n_1, n_2, n_3, n_4, n_5, respectivement. Les trajets attractifs, considérés comme les séquences de nœuds, sont montrés dans le Tableau V.4 et visualisé sur Google Maps (Figure V.5).

Tableau V.4: Les séquences de nœuds des chemins attractifs

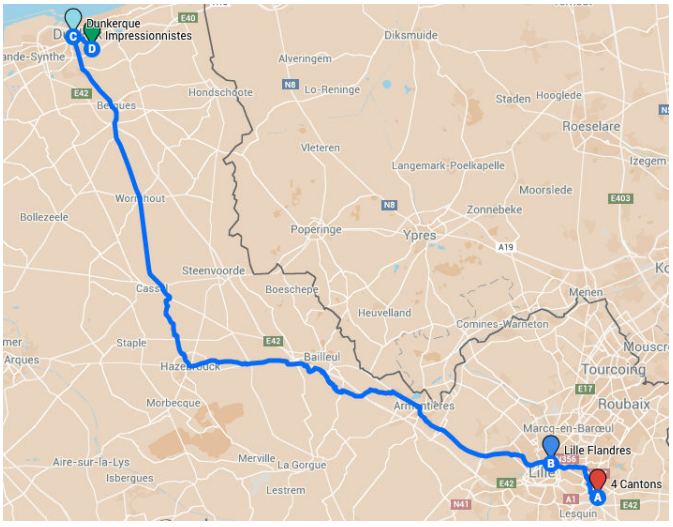
Option	**Séquence modale**	**Séquence de nœuds**	**Détail de route**
1	$(C,W,B/V)$	(n_1, n_4, n_5)	Figure V.5A
2	$(M,W,T/C,W,B/V)$	(n_1, n_2, n_4, n_5)	Figure V.5B
3	$(M,W,C,W,B/V)$	(n_1, n_3, n_4, n_5)	Figure V.5C

Selon les conditions imposées, le but est de trouver les itinéraires équilibrés pour le temps de parcours et le coût de trajet. Les manipulations de la fenêtre de temps permettent de calculer les fenêtres temporelles pour tous les nœuds de la séquence et, ainsi, d'identifier les véhicules "intéressants" dont les horaires correspondent aux exigences de la requête. La Figure V.6 montre ces fenêtres temporelles et les véhicules desservant les tronçons de route pour la séquence de nœuds de l'option 3.

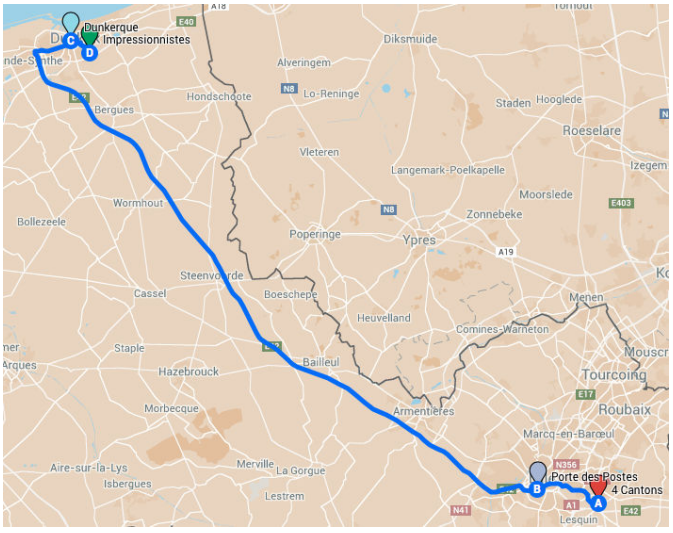
Pour chaque séquence de nœuds, trouver les itinéraires équilibrés, c'est résoudre le problème de séquence multimodale avec le critère de temps de parcours et le coût du trajet (cf. section III.5.1). Les arrangements d'itinéraires optimaux au sens Pareto sont montrés sur la Figure V.7. Les itinéraires sont mis sous forme de la feuille de route détaillée, et indiqués dans le Tableau V.5.



(A) Chemin attractif 1



(B) Chemin attractif 2



(C) Chemin attractif 3

FIGURE V.5: Visualisation des chemins avec Google Maps

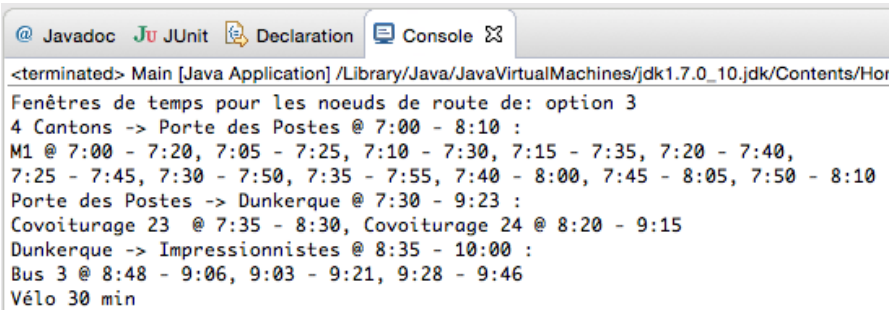


FIGURE V.6: Les fenêtres temporelles et les véhicules identifiés pour les nœuds de l'option 3

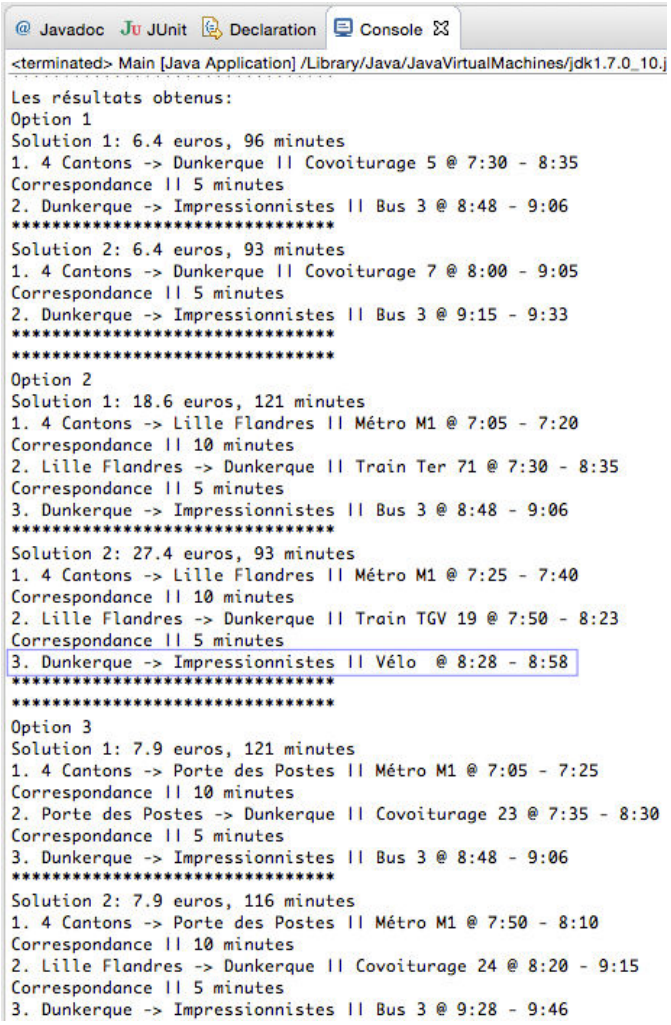


FIGURE V.7: Les résultats itinéraires pour le scénario 1

Tableau V.5: La feuille de route

Résultats	**Étape**	**Mode**	**Tronçon route**	**Horaires**	**Durée**	**Coût**	**Temps**
1-1	1	*C* (Covoiturage 5)	4 Cantons	7h30	1h05	6,4 euros	96 min
			Dunkerque (covoiturage)	8h35			
	1'	*W*	Correspondance	8h35	5 min		
	2	*B* (Bus 3)	Dunkerque (bus)	8h48	26 min		
			Impressionnistes	9h06			
1-2	1	*C* (Covoiturage 7)	4 Cantons	8h00	1h05	6,4 euros	93 min
			Dunkerque (covoiturage)	9h05			
	1'	*W*	Correspondance	9h05	5 min		
	2	*B* (Bus 3)	Dunkerque (bus)	9h15	18 min		
			Impressionnistes	9h33			
2-1	1	*M* (M1)	4 Cantons	7h05	15 min	18,6 euros	121 min
			Lille Flandres (Métro)	7h20			
	1'	*W*	Correspondance	7h20	10 min		
	2	*T*(Ter 71)	Lille Flandres (Gare train)	7h30	1h05		
			Dunkerque (Gare train)	8h35			
	2'	*W*	Correspondance	8h35	5 min		
	3	*B* (Bus 3)	Dunkerque (bus)	8h48	18 min		
			Impressionnistes	9h06			
2-2	1	*M* (M1)	4 Cantons	7h25	15 min	27,4 euros	93 min
			Lille Flandres (Métro)	7h40			
	1'	*W*	Correspondance	7h40	10 min		

	2	T(TGV 19)	Lille Flandres (Gare train)	7h50	33 min		
			Dunkerque (Gare train)	8h23			
	2'	W	Correspondance	8h23	5 min		
	3	V (Vélo-partage)	Dunkerque (Vélo)	8h28	30 min		
			Impressionnistes	8h58			
3-1	1	M (M1)	4 Cantons	7h05	20 min	7,9 euros	121 min
			Porte des Postes (Métro)	7h25			
	1'	W	Correspondance	7h25	10 min		
	2	C (Covoiturage 23)	Porte des Postes (Covoiturage)	7h35	55 min		
			Dunkerque (Covoiturage)	8h30			
	2'	W	Correspondance	8h30	5 min		
	3	B (Bus 3)	Dunkerque (Bus)	8h48	18 min		
			Impressionnistes	9h06			
3-2	1	M (M1)	4 Cantons	7h50	20 min	7,9 euros	116 min
			Porte des Postes (Métro)	8h10			
	1'	W	Correspondance	8h10	10 min		
	2	C (Covoiturage 24)	Porte des Postes (Covoiturage)	8h20	55 min		
			Dunkerque (Covoiturage)	9h15			
	2'	W	Correspondance	9h15	5 min		
	3	B (Bus 3)	Dunkerque (Bus)	9h28	18 min		
			Impressionnistes	9h46			

Pour la deuxième solution de l'option 2, on constate que le dernier morceau de route est effectué à vélo au lieu de bus pour les autres itinéraires. Ce choix permet désormais au voyageur d'arriver à la destination plus tôt qu'attendre et prendre le bus. Si la condition de temps d'arrivée au plus tard est 9h, celle-ci sera la seule solution valable.

Sur les trois trajets attractifs, nous avons mesuré le nombre de nœuds exploités pour obtenir les arrangements Pareto optimaux. Les nœuds exploités sont ceux sur la séquence de nœuds qui sont étiquetés dans le graphe temps-étendu par notre algorithme, soit avec l'étiquetage vers l'avant, soit avec l'étiquetage vers l'arrière, soit avec une recherche bidirectionnelle. Une comparaison de ces trois cas est montrée dans la Figure V.8. La recherche bidirectionnelle permet de diminuer davantage le nombre de nœuds exploités par rapport à celles avec l'étiquetage vers l'avant ou l'arrière.

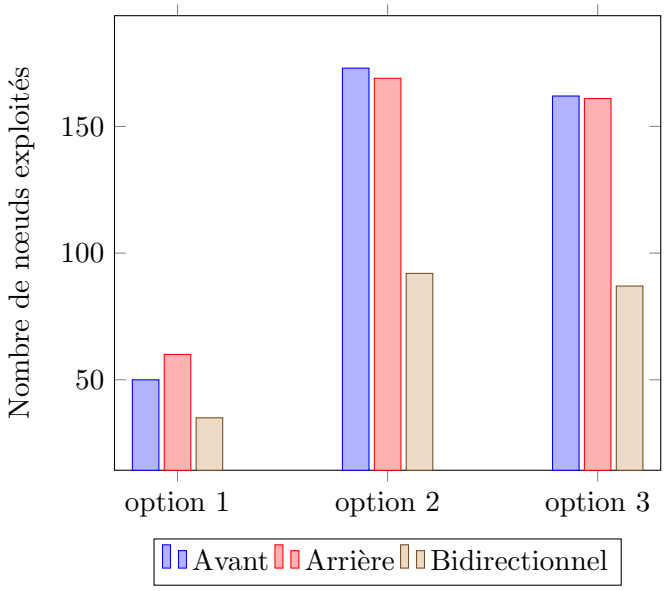


FIGURE V.8: Impact du type de recherche sur le nombre de nœuds exploités

V.4.5.2 Scénario 2 : les requêtes simultanées

Dans ce jeu de test, nous étudions le cas de requêtes simultanées. À l'instant t, le système accueille un ensemble de requêtes d'itinéraire $I = \{I_1, I_2, \ldots, I_5\}$. Dans le Tableau V.6 figurent les éléments des requêtes spécifiées via l'interface du système proposé.

Identification du domaine de recherche

Le Task Agent cherche à identifier le domaine de recherche après avoir reçu les requêtes d'itinéraire. Les points de départs et d'arrivées sont repérés dans un premier temps,

3. Sans train
4. Transport en commun

Tableau V.6: Les requêtes reçues

Requête	Départ	Arrivée	T.D.	T.A.	Pré.rence	Critère
I_1	4 Cantons	Stade Calais	7h00	10h30	-(T)[3]	Plus rapide
I_2	V. d'Ascq	Phare de Calais	9h00	12h00	tous	Moins cher
I_3	Cité Scientifique	Centre Universitaire	7h00	10h00	T.C.[4]	Plus rapide
I_4	Porte de Douai	Calais Musée BA	-	12h00	tous	Moins cher
I_5	Wazemmes	Stade Calais	8h00	-	tous	Plus rapide

ensuite le Task Agent exécute l'algorithme de la section III.4.1 pour trouver le domaine de recherche et les Module Agents correspondants. Pour chaque requête reçue, son domaine de recherche est identifié, comme le montre la Figure V.9.

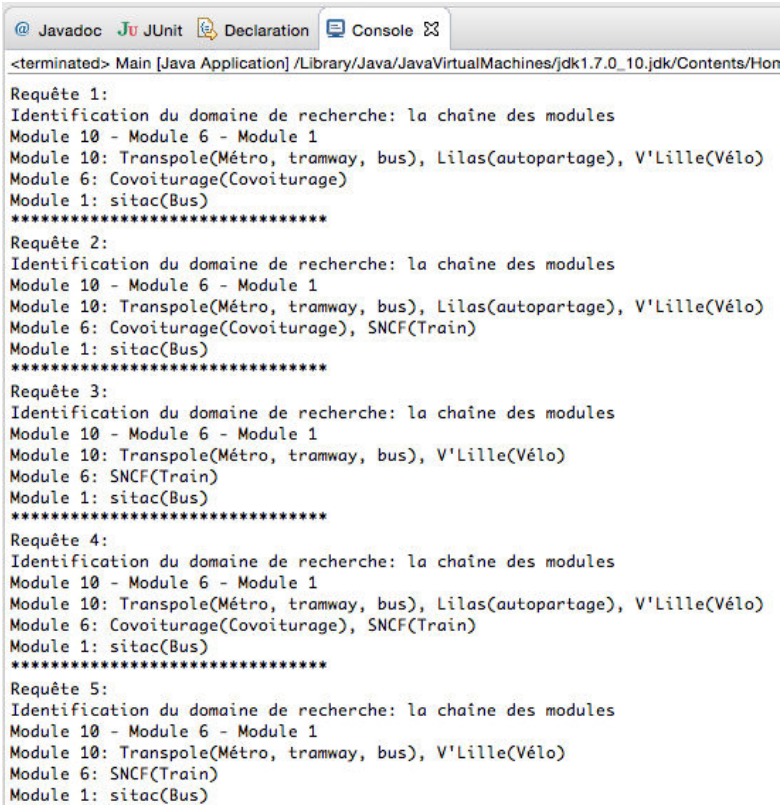


Figure V.9: Identification du domaine de recherche du scénario 2

Les Module Agents pour l'ensemble du réseau sont organisés selon la Figure V.10. Les Module Agents envoient pour chaque requête les opérateurs de transport qui vont participer dans la prochaine étape pour répondre à la requête.

Selon les résultats de l'étape d'identification du domaine de recherche, il s'agit de Module Agent 1, Module Agent 6 et Module Agent 10 dans ce scénario. Les services et les opérateurs de transport pour les différents Module Agents sont listés comme suivant :

- Module Agent 1. Query Agent sitac : Opérateur bus à Calais.
- Module Agent 6. Query Agent Covoiturage : service de covoiturage ; Query Agent SNCF : Opérateur SNCF.

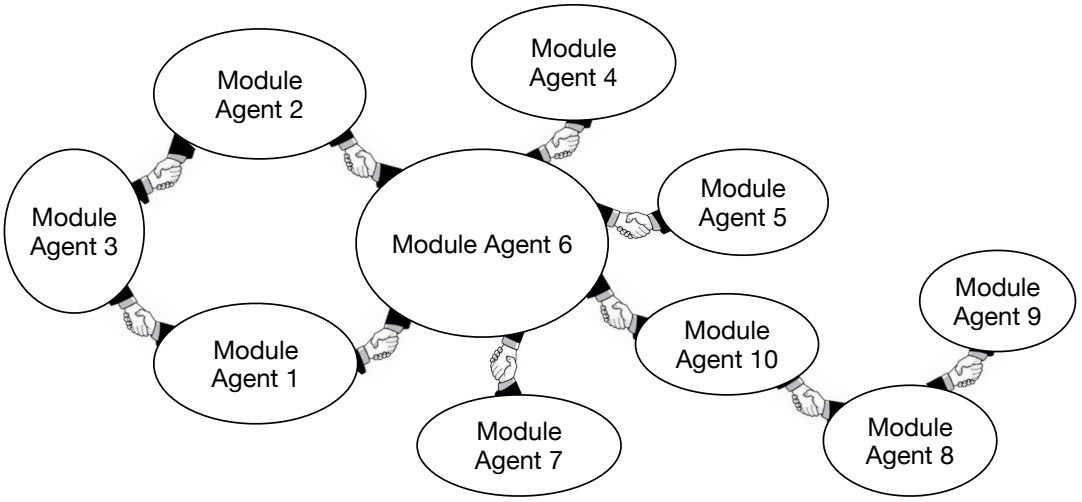


FIGURE V.10: Organisations des Modules Agents

— Module Agent 10. Query Agent V'Lille Vélo : service de vélo-partage ; Query Agent Transpole public : service de transport en commun à Lille ; Query Agent Lilas : service d'autopartage.

L'outil Sniffer est disponible pour afficher les communications inter-agents du système (avec des messages *FIPA-ACL*). À l'aide de cet outil, la Figure V.11 visualise les communications entre les Task Agents, les Modules Agents et les Interface Agents.

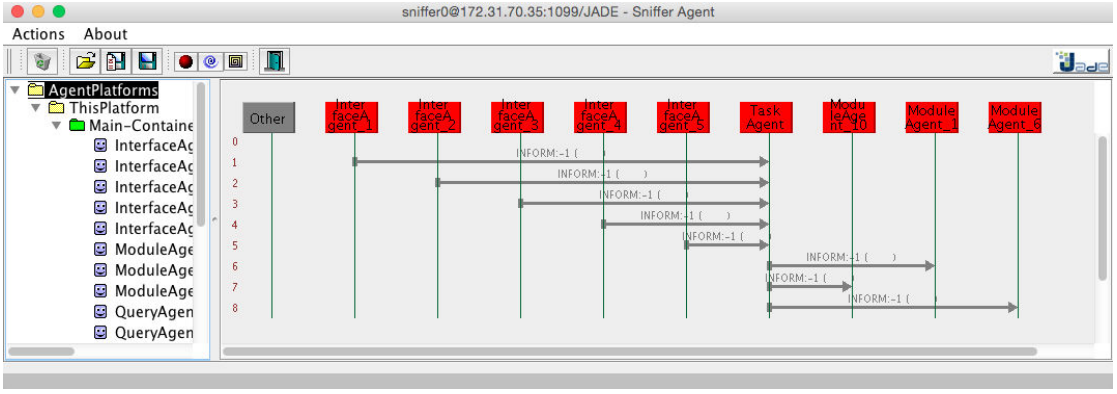


FIGURE V.11: Communications entre les Interface Agents, Task Agents et Module Agents

Avec les opérateurs de transport identifiés pour chaque requête, le Task Agent s'engage par la suite à calculer les chemins attractifs en envoyant les requêtes aux Query Agents des opérateurs correspondants. L'algorithme présenté dans la section III.4.2.1 permet d'obtenir les chemins attractifs pour les requêtes d'itinéraires.

Dans le but de faciliter la notation, nous notons les nœuds suivants 4 Cantons, Lille Flandres, Porte des Postes, Gare Calais, Stade Calais, V. d'Ascq, Phare de Calais, Cité Scientifique, Centre Universitaire Calais, Porte de Douai, Calai Musée BA et Wazemmes

comme n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8, n_9, n_{10}, n_{11} et n_{12}. Les chemins attractifs pour les requêtes sont montrés avec le Tableau V.7.

TABLEAU V.7: Les séquences de nœuds des chemins attractifs pour les requêtes I_i

Requête	Options	Séquence modale	Séquence de nœuds
I_1	1	(C,B)	(n_1, n_4, n_5)
	2	(M,C,B)	(n_1, n_3, n_4, n_5)
	3	(M,C,B)	(n_1, n_2, n_4, n_5)
I_2	1	(M,C,B)	(n_6, n_3, n_4, n_7)
	2	$(M,C/T,B)$	(n_6, n_2, n_4, n_7)
I_3	1	(M,T,B)	(n_8, n_2, n_4, n_9)
	2	(B,T,B)	(n_8, n_2, n_4, n_9)
I_4	1	$(M,C/T,B)$	$(n_{10}, n_2, n_4, n_{11})$
	2	(M,C,B)	$(n_{10}, n_3, n_4, n_{11})$
	3	$(A,C/T,B)$	$(n_{10}, n_2, n_4, n_{11})$
I_5	1	(V,C,B)	(n_{12}, n_3, n_4, n_5)
	2	(M,C,B)	(n_{12}, n_3, n_4, n_5)
	3	$(M,C/T,B)$	(n_{12}, n_2, n_4, n_5)

Affectation entre les offres et les demandes

Selon les séquences de nœuds pour les requêtes, le Task Agent va ensuite générer des Route Agents chacun représente un morceau de route.

On prend la section de route Lille Flandres-Calais comme exemple car deux modes de transport se présentent pour assurer ce tronçon, à savoir le train et le covoiturage. Sur ce tronçon de route, les fenêtres temporelles sont déterminées, et visualisées dans la Figure V.12. Pour le Route Agent Lille Flandres-Calais, il s'agit des requêtes I_1, I_2, I_3, I_4 et I_5 sur la fenêtre de temps [7h, 12h] qui couvre toutes ces requêtes.

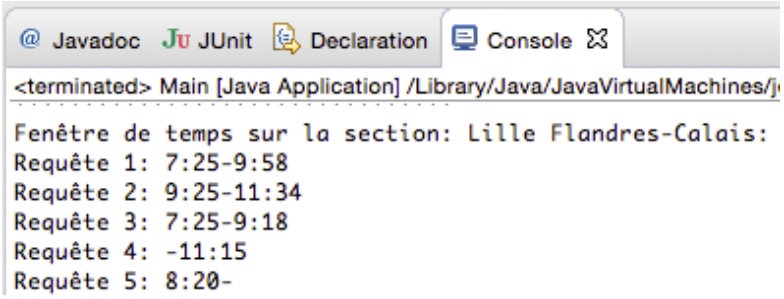


FIGURE V.12: Fenêtre de temps pour la section de route Lille Flandres-Calais

Avec les conditions temporelles, le Route Agent Lille Flandres-Calais s'engage à identifier les véhicules desservant ce tronçon de route.

Les véhicules respectant les conditions de temps sur ce tronçon de route sont affichés par la Figure V.13.

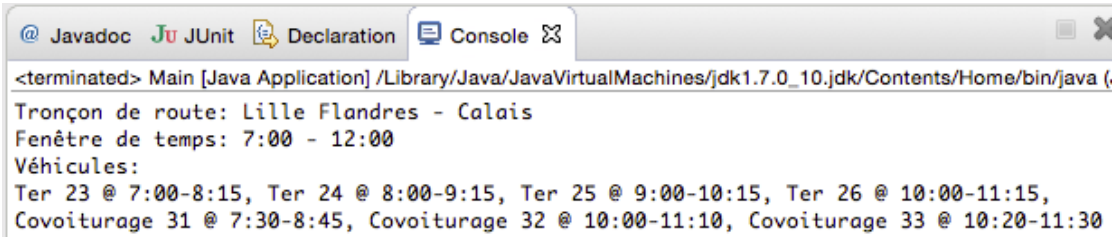


FIGURE V.13: Les véhicules identifiés pour Lille Flandres-Calais

Le Tableau V.8 montre le schéma d'affectation qui représente les préférences de modes et les contraintes temporelles de chaque requêtes.

TABLEAU V.8: Schéma d'affectation pour le Route Agent Lille Flandres-Calais

	Ter 23	Cov 31(1,3)	Ter 24	Ter 25	Ter 26	Cov 32(1,3)	Cov 33(1,4)
I_1	0	*	0	0	0	*	0
I_2	0	0	0	0	*	*	*
I_3	0	0	*	0	0	0	0
I_4	*	*	*	*	*	*	*
I_5	0	0	0	*	*	*	*

Par la suite, le processus d'affectation avec le Route Agent Lille Flandres-Calais cherche la matrice d'affectation selon l'algorithme génétique introduit dans la section III.7. Les probabilités de croisement et de mutation pour l'algorithme génétique utlisé sont respectivement 0,9 et 0,1. Un chromosome relatif au Route Agent Lille Flandres-Calais est montré dans le Tableau V.9.

TABLEAU V.9: Résultat d'affectation pour le Route Agent Lille Flandres-Calais

	Ter 23	Cov 31(0,3)	Ter 24	Ter 25	Ter 26	Cov 32(0,3)	Cov 33(0,4)
I_1	0	1	0	0	0	*	0
I_2	0	0	0	0	*	*	1
I_3	0	0	1	0	0	0	0
I_4	*	*	*	*	*	1	*
I_5	0	0	0	1	*	*	*

Supposons qu'une perturbation se produit (i.e. la grève) où n'y a qu'un train pour assurer un service minimum à 8h pour le trajet Lille Flandres-Calais. Les véhicules pour le Route Agent Lille Flandres-Calais sont indiqués dans la Figure V.14.

Les préférences sur les modes, les contraintes temporelles et les ressources disponibles permettent d'établir le schéma d'affectation montré dans le Tableau V.10.

À partir du schéma d'affectation en cas de perturbation, le Route Agent Lille Flandres-Calais a trouvé la matrice d'affectation présentée dans le Tableau V.11 avec l'algorithme génétique.

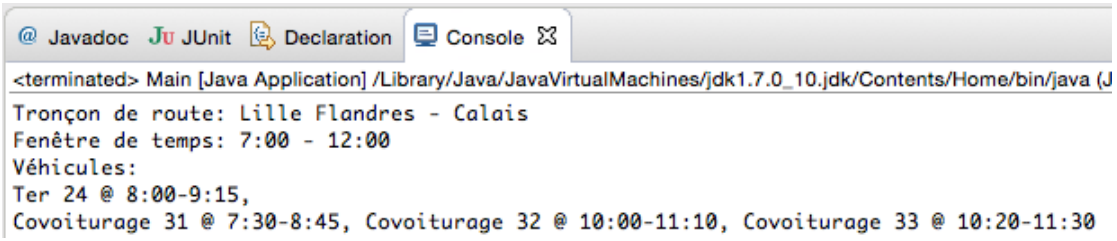


FIGURE V.14: Les véhicules identifiés sur Lille Flandres-Calais en cas de perturbation

TABLEAU V.10: Schéma d'affectation pour le Route Agent Lille Flandres-Calais en cas de perturbation

	Cov 31(1,3)	Ter 24	Cov 32(1,3)	Cov 33(1,4)
I_1	*	0	*	0
I_2	0	0	*	*
I_3	0	*	0	0
I_4	*	*	*	*
I_5	0	0	*	*

TABLEAU V.11: Chromosome pour le Route Agent Lille Flandres-Calais en cas de perturbation

	Cov 31(0,3)	Ter 24	Cov 32(0,3)	Cov 33(0,4)
I_1	1	0	*	0
I_2	0	0	*	1
I_3	0	1	0	0
I_4	*	1	*	*
I_5	0	0	1	*

Coalition entre Route Agents

Les Route Agents s'engagent ensuite dans la coalition pour composer les itinéraires auprès des structures de coalition selon les critères des usagers.

À l'aide de l'outil Sniffer Agent, une instance de coalition entre les agents pour composer les itinéraires est visualisée sous cette forme dans la Figure V.15.

La coalition des Route Agents a pour objectif de composer les itinéraires qui sont évalués par l'Assist Agent. Nous avons obtenu les itinéraires contenant le morceau Lille Flandres-Calais. Les itinéraires sont obtenus dans le cas normal (Figure V.16) et dans le cas perturbé (Figure V.17).

Nous constatons que les requêtes I_1, I_2 et I_3 ont les mêmes solutions dans les deux cas (avec ou sans perturbation). En revanche, la place de Covoiturage 32 est attribuée à I_5 car le Ter 25 n'est plus disponible dans le deuxième cas. I_4 prend le Ter 24 au lieu de covoiturage parce qu'il ne reste plus de place de covoiturage sur le tronçon Lille Flandres-Calais, en même temps, les préférences sont également respectées.

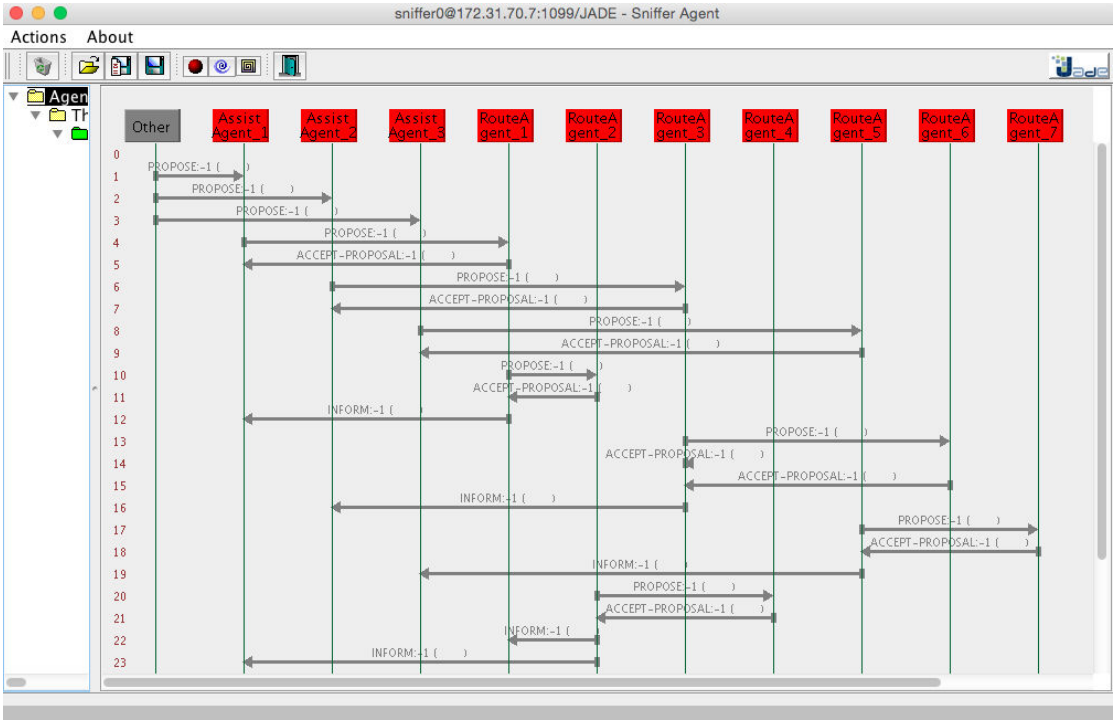


FIGURE V.15: Illustration de la communication pour la formation de coalition entre les Route Agents à l'aide de l'outil SNIFFER

Avec notre approche de la coalition, les itinéraires comodaux trouvés (Figure V.16 et Figure V.17) sont efficaces et conformes aux préférences des requêtes.

V.4.5.3 Scénario 3 : Grand nombre de requêtes

Dans ce scénario, nous allons tester la robustesse du système face à un grand nombre de requêtes simultanées. Tout d'abord, nous allons comparer les deux approches : programmation linéaire et algorithme génétique contrôlé pour le problème d'affectation. Ensuite, nous allons montrer les performances du système face à un grand nombre de requêtes.

Nous testons deux implantations différentes : l'une avec l'algorithme génétique implémenté en Java, et l'autre, avec CPLEX interfacé avec Java. Ces tests sont basés sur les instances d'une fenêtre de temps de 2 heures et une section de route avec 10 véhicules. Nous avons mis en évidence l'influence du nombre de requêtes sur le temps de calcul et le taux de réussite de l'affectation optimale. Pour chaque série de tests, chaque algorithme est exécuté 20 fois et le résultat retenu est basé sur la moyenne de résultats obtenus.

Toujours dans la même configuration du réseau de transport, l'algorithme génétique est paramétré avec la population de 100 individus, la probabilité de croisement de 0,9 et la

FIGURE V.16: Les itinéraires contenant le morceau Lille Flandres-Calais pour les requêtes

FIGURE V.17: Les itinéraires contenant le morceau Lille Flandres-Calais pour les requêtes en cas de perturbation

probabilité de mutation de 0,1. Les calculs sont réalisés sur un ordinateur de processeur Core 2 à 2,67GHz et 4 Go de RAM.

Nous constatons d'abord que l'algorithme génétique est meilleur que CPLEX sur le temps de calcul, surtout quand le nombre de requêtes traitées est importante (Figure V.18). En revanche, la programmation linéaire est bien meilleure en terme de taux de réussite d'affectation optimale (Figure V.19). Pour interpréter ces écarts de performances, on peut noter que l'algorithme génétique est une métaheuristique efficace en terme de temps de calcul et que le CPLEX est un outil spécialisé en PL (Programmation Linéaire) pour chercher les solutions optimales.

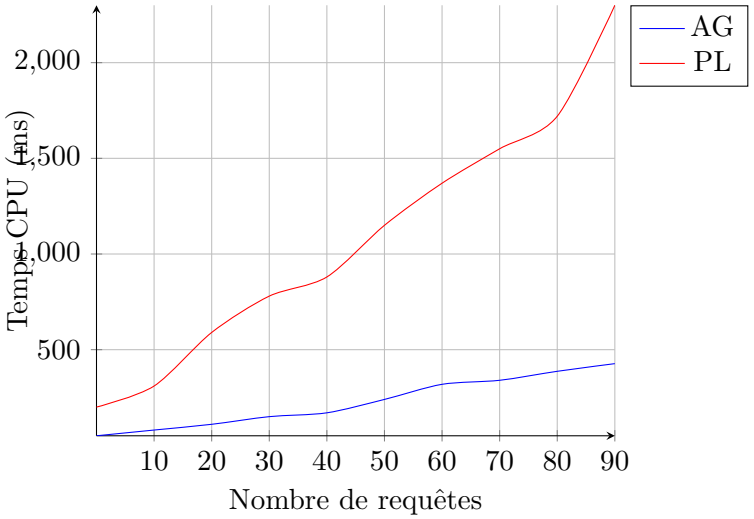


FIGURE V.18: Influence du nombre de requêtes : temps de calcul avec AG et PL(moyenne sur 20 tests)

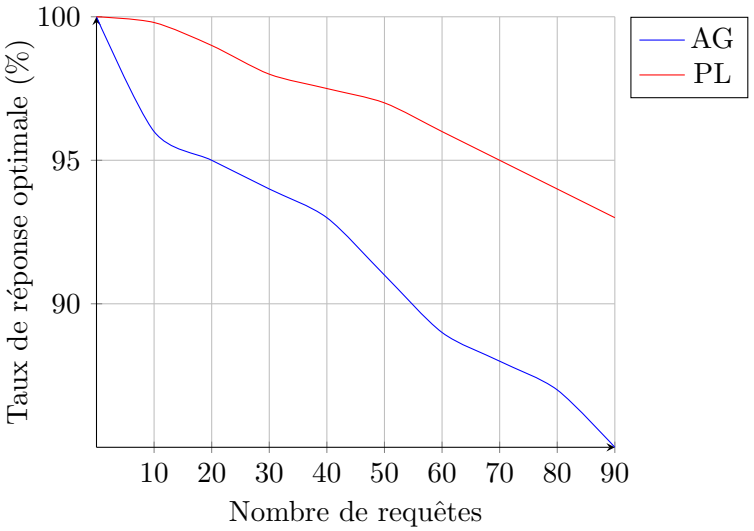


FIGURE V.19: Influence du nombre de requêtes : taux de résultat optimal(moyenne sur 20 tests)

D'après la Figure V.20, notre système continue toujours à répondre aux requêtes avec temps de réponse optimisé et ceci grâce à notre approche d'alliance du système multi-agents et de l'optimisation. Malgré le nombre très élevé de requêtes simultanées, le système est capable de gérer les nombreuses demandes efficacement.

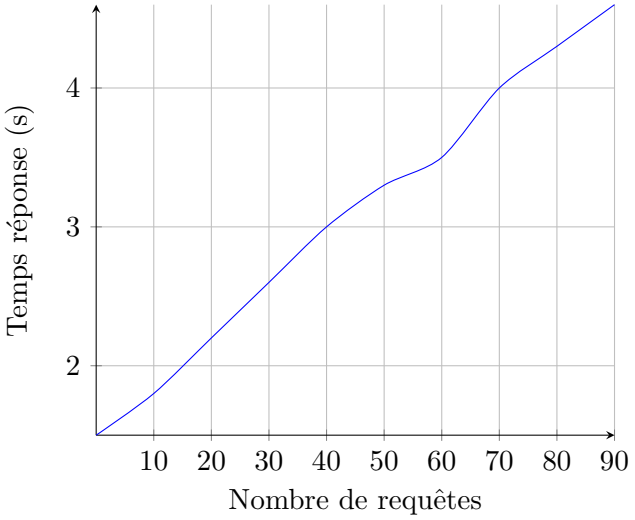


FIGURE V.20: Influence de nombre de requêtes : temps de réponse pour les requêtes d'itinéraires(moyenne sur 20 tests)

V.5 Conclusion

Dans ce chapitre, nous avons présenté les différents aspects concernant la mise en œuvre de notre application relative à la planification d'itinéraire et le management de ressources de transports :

- les plateformes et outils de développement ;
- la collecte de données ;
- la saisie des requêtes ;
- le traitement des requêtes d'itinéraire ;
- la coalition de service pour la recomposition des itinéraires.

Pour les requêtes simples, les itinéraires optimaux au sens de Pareto sont déterminés à l'aide de décision des usagers. Pour les requêtes simultanées, nous cherchons à identifier les itinéraires intéressants via la combinaison des solutions optimisées en faisant usage des coalitions. Les intérêts pratiques des approches proposées ont été exposés à travers deux types d'applications. Les jeux de test ont également montré que le système donne les réponses adéquates en cas de perturbation du réseau de transport. Selon les études sur les rapports entre le nombre de requêtes et le temps de réponse, nous pouvons assurer que notre système est robuste et capable de faire face à l'augmentation de la charge(nombre de requêtes important).

Avec les résultats et scénarios de simulation obtenus, le système manifeste la capacité à fournir les solutions pour répondre aux attentes des usagers, que ce soit pour la requête simple ou l'ensemble de demandes d'itinéraires simultanées. En cas de perturbation du réseau de transport, une allocation optimale des offres limitées aux demandes est réalisée pour satisfaire toutes les requêtes reçues.

Conclusion générale

Les travaux réalisés durant cette thèse se sont focalisés sur le développement d'un système d'information de transport comodal qui adopte une approche multi-agents intégrant un aspect optimisation. Les recherches se sont concentrées sur trois problèmes emboîtés :

1. la planification d'itinéraire dans le réseau de transport multimodal avec la prise en compte des véhicules partagés ;
2. le problème d'affectation avec l'optimisation multiobjectif ;
3. le problème de recomposition d'itinéraires formant une coalition au sein d'un système multi-agents.

L'étude du premier problème consiste à trouver les itinéraires attractifs dans un réseau de transport multimodal qui est modélisé par un graphe hiérarchique. Le domaine de recherche est identifié à l'aide de ce dernier afin de limiter le nombre de sous-systèmes demandées pour calculer les itinéraires. Avec une méthode exacte, les arrangements optimaux d'un trajet avec les nœuds intermédiaires peuvent être déterminés en respectant les contraintes horaires sur le départ et l'arrivée.

L'étude du deuxième problème a permis de formuler et de résoudre le problème d'affectation entre les offres et les demandes sur les tronçons de route. À partir des résultats du premier problème, les offres des véhicules conformes aux conditions temporelles identifiées pour les requêtes sont mises en relation avec les demandes. Pour ce faire, nous avons appliqué l'optimisation multicritère basée sur l'algorithme génétique.

L'étude du troisième problème a permis de recomposer les tronçons de routes en suivant les structures de coalition afin d'obtenir les itinéraires optimaux. Pour ce faire, une formation de coalition est effectuée entre les agents représentants les tronçons de routes. Un ensemble d'itinéraires intéressants répondant aux attentes des usagers est fourni tout en gardant une certaine flexibilité à l'usager.

Pour la mise en œuvre des approches, nous avons appliqué la programmation orientée agent qui nous permet éventuellement de profiter également de la combinaison entre la technologie des systèmes multi-agents et l'optimisation distribuée tout au long de la résolution du problème. Cette mise en œuvre a permet de mettre en évidence les intérêts pratiques pour l'information des usagers où plus de modes de transport tel que les véhicules partagés sont intégrés.

Perspectives

Les perspectives de recherche pour étendre nos travaux sont assez diverses.

Comme notre système possède une grande flexibilité, il est toujours possible d'intégrer les véhicules particuliers pour la planification d'itinéraire. Sachant que l'usage des véhicules personnels reste encore très répandu pour les voyages occasionnels et les trajets fréquents, il est donc possible d'ajouter cette option afin d'avoir une plateforme pouvant répondre aux besoins des usagers. Il convient de noter que *park-and-ride* joue un rôle très important pour la complémentarité entre les voitures personnelles et les transports en commun, surtout dans les régions urbaines. Cependant, il faut aussi prendre en compte le fait que les approches de calculs des plus court chemins dans les transports en commun ne sont pas forcément efficaces, et les méthodes pour le réseau routier doivent ainsi être reconsidérées. L'affichage des résultats peut être plus intelligent. Comme les résultats de la planification d'itinéraires contiennent des informations très riches telle que les sections de routes de différents modes avec des coûts variés, et des actions de correspondance. Visualiser tous ces détails sur la carte d'une façon adéquate peut rendre l'outil plus intéressant pour l'usage quotidien, par exemple la visualisation interactive, l'effet zoom pour les actions de correspondance, etc.

Comme le système proposé concerne la planification d'itinéraire *avant* le voyage, il serait intéressant de l'étendre pour qu'il puisse fournir des itinéraires pendant le voyage (*en route*) avec la prise en compte des informations en temps réels. Avec les informations sur des perturbations ponctuelles (panne, accident, embouteillage, grève, etc.), le problème d'anticipation peut être également abordé pour proposer aux voyageurs par exemple des itinéraires alternatifs, le temps de voyage supplémentaire.

Bibliographie

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

[Ahn and Ramakrishna, 2002] Ahn, C. W. and Ramakrishna, R. S. (2002). A genetic algorithm for shortest path routing problem and the sizing of populations. *Evolutionary Computation, IEEE Transactions on*, 6(6) :566–579.

[Aknine and Shehory, 2005] Aknine, S. and Shehory, O. (2005). Coalition formation : Concessions, task relationships and complexity reduction. *arXiv preprint cs/0502094*.

[Amey, 2010] Amey, A. M. (2010). *Real-time ridesharing : exploring the opportunities and challenges of designing a technology-based rideshare trial for the MIT community.* PhD thesis, Massachusetts Institute of Technology.

[Bäck, 1996] Bäck, T. (1996). Evolution strategies : An alternative evolutionary algorithm. In *Artificial evolution*, pages 1–20. Springer.

[Bäck and Schwefel, 1993] Bäck, T. and Schwefel, H.-P. (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary computation*, 1(1) :1–23.

[Bast et al., 2010] Bast, H., Carlsson, E., Eigenwillig, A., Geisberger, R., Harrelson, C., Raychev, V., and Viger, F. (2010). Fast routing in very large public transportation networks using transfer patterns. In Berg, M. and Meyer, U., editors, *Algorithms – ESA 2010*, volume 6346 of *Lecture Notes in Computer Science*, pages 290–301. Springer Berlin Heidelberg.

[Bast et al., 2009] Bast, H., Funke, S., and Matijevic, D. (2009). Ultrafast shortest-path queries via transit nodes. *The Shortest Path Problem : Ninth DIMACS Implementation Challenge*, 74 :175–192.

[Bast et al., 2007] Bast, H., Funke, S., Matijevic, D., Sanders, P., and Schultes, D. (2007). In transit to constant time shortest-path queries in road networks. In *ALENEX*. SIAM.

[Bauer et al., 2013] Bauer, R., Columbus, T., Rutter, I., and Wagner, D. (2013). Search-space size in contraction hierarchies. In *Automata, Languages, and Programming*, pages 93–104. Springer.

[Bauer et al., 2010] Bauer, R., Delling, D., Sanders, P., Schieferdecker, D., Schultes, D., and Wagner, D. (2010). Combining hierarchical and goal-directed speed-up techniques for dijkstra's algorithm. *Journal of Experimental Algorithmics (JEA)*, 15 :2–3.

[Bauer et al., 2011] Bauer, R., Delling, D., and Wagner, D. (2011). Experimental study of speed up techniques for timetable information systems. *Networks*, 57(1) :38–52.

[Bellman, 1957] Bellman, R. (1957). *Dynamic Programming*. Princeton Univ. Press, Princeton, N. J.

[Bellman, 1958] Bellman, R. (1958). On a routing problem. *Quarterly of Applied Mathematics*, 16 :87–90.

[Bielli et al., 2006] Bielli, M., Boulmakoul, A., and Mouncif, H. (2006). Object modeling and path computation for multimodal travel systems. *European Journal of Operational Research*, 175(3) :1705–1730.

[Breban and Vassileva, 2002] Breban, S. and Vassileva, J. (2002). A coalition formation mechanism based on inter-agent trust relationships. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems : part 1*, pages 306–307. ACM.

[Chaib-draa, 1996] Chaib-draa, B. (1996). Interaction between agents in routine, familiar and unfamiliar situation. *International journal of cooperative information systems*, 5(01) :1–25.

[Chakraborty et al., 1996] Chakraborty, U., Deb, K., and Chakraborty, M. (1996). Analysis of selection algorithms : A markov chain approach. *Evolutionary Computation*, 4(2) :133–167.

[Clavel et al., 2008] Clavel, R., Mariotto, M., Arsac, B., et al. (2008). L'autopartage en france et en europe : état des lieux et perspectives.

[Colorni et al., 1991] Colorni, A., Dorigo, M., Maniezzo, V., et al. (1991). Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life*, volume 142, pages 134–142. Paris, France.

[Commission, 2009] Commission, E. (2009). A sustainable future for transport : toward an integrated, technology-led and user friendly system. European Commission, Luxembourg.

[Cormen et al., 1996] Cormen, T. H., Leiserson, C. E., Rivest, R. L., Cormen, T. H., and Cormen, T. H. (1996). *Introduction à l'algorithmique*. Dunod.

[Crina and Ajith, 2006] Crina, G. and Ajith, A. (2006). Stigmergic optimization : Inspiration, technologies and perspectives. In *Stigmergic optimization*, pages 1–24. Springer.

[Dailey et al., 1999] Dailey, D., Loseff, D., and Meyers, D. (1999). Seattle smart traveler : dynamic ridematching on the world wide web. *Transportation Research Part C : Emerging Technologies*, 7(1) :17–32.

[Dantzig, 1998] Dantzig, G. B. (1998). *Linear programming and extensions*. Princeton university press.

[Davis et al., 1991] Davis, L. et al. (1991). *Handbook of genetic algorithms*, volume 115. Van Nostrand Reinhold New York.

[Deakin et al., 2010] Deakin, E., Frick, K. T., and Shively, K. M. (2010). Markets for dynamic ridesharing ? *Transportation Research Record : Journal of the Transportation Research Board*, 2187(1) :131–137.

[Deb, 1999] Deb, K. (1999). Multi-objective genetic algorithms : Problem difficulties and construction of test problems. *Evolutionary computation*, 7(3) :205–230.

[Deb et al., 2002] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm : Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2) :182–197.

[Dechter and Pearl, 1985] Dechter, R. and Pearl, J. (1985). Generalized best-first search strategies and the optimality of a*. *J. ACM*, 32(3) :505–536.

[Delling et al., 2013] Delling, D., Pajor, T., and Werneck, R. F. (2013). Round-based public transit routing. In *Sixth Annual Symposium on Combinatorial Search*.

[Diana and Dessouky, 2004] Diana, M. and Dessouky, M. M. (2004). A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows. *Transportation Research Part B : Methodological*, 38(6) :539–557.

[Diestel, 2005] Diestel, R. (2005). Graph theory. 2005. *Grad. Texts in Math.*

[Dijkstra, 1959] Dijkstra, E. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1) :269–271.

[Dorigo et al., 1999] Dorigo, M., Caro, G. D., and Gambardella, L. M. (1999). Ant algorithms for discrete optimization. *Artificial life*, 5(2) :137–172.

[Dorigo and Gambardella, 1997] Dorigo, M. and Gambardella, L. M. (1997). Ant colonies for the travelling salesman problem. *BioSystems*, 43(2) :73–81.

[Durfee and Lesser, 1991] Durfee, E. H. and Lesser, V. R. (1991). Partial global planning : A coordination framework for distributed hypothesis formation. *Systems, Man and Cybernetics, IEEE Transactions on*, 21(5) :1167–1183.

[Efentakis et al., 2011] Efentakis, A., Pfoser, D., and Voisard, A. (2011). Efficient data management in support of shortest-path computation. In *Proceedings of the 4th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, pages 28–33. ACM.

[Fagin and Williams, 1983] Fagin, R. and Williams, J. H. (1983). A fair carpool scheduling algorithm. *IBM Journal of Research and development*, 27(2) :133–139.

[Fayech, 2003] Fayech, B. (2003). *Régulation des réseaux de transport multimodal : Systèmes multi-agents et algorithmes évolutionnistes*. PhD thesis, Université de Lille 1 et École Centrale de Lille.

[Feki, 2010] Feki, M. (2010). *Distributed optimization for multi-operator routes search in co-modal transport network*. PhD thesis, Ecole Centrale de Lille.

[Ferber, 1995] Ferber, J. (1995). *Les système multi-agent : vers une intelligence collective*, volume 1. Paris, France : InterEdition.

[Ferber, 1999] Ferber, J. (1999). *Multi-agent systems : an introduction to distributed artificial intelligence*, volume 1. Addison-Wesley Reading.

[Fischer et al., 1995] Fischer, K., Müller, J. P., Pischel, M., and Schier, D. (1995). A model for cooperative transportation scheduling. In *ICMAS*, pages 109–116.

[Fonseca et al., 1993] Fonseca, C. M., Fleming, P. J., et al. (1993). Genetic algorithms for multiobjective optimization : Formulationdiscussion and generalization. In *ICGA*, volume 93, pages 416–423.

[Ford, 1956] Ford, L. R. (1956). Network flow theory.

[Fouchal et al., 2011] Fouchal, H., Gandibleux, X., and Lehuédé, F. (2011). Preferred solutions computed with a label setting algorithm based on choquet integral for multiobjective shortest paths. In *Computational Intelligence in Multicriteria Decision-Making (MDCM), 2011 IEEE Symposium on*, pages 143–150. IEEE.

[Fredman and Tarjan, 1987] Fredman, M. L. and Tarjan, R. E. (1987). Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3) :596–615.

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

[Galand et al., 2010] Galand, L., Perny, P., and Spanjaard, O. (2010). Choquet-based optimisation in multiobjective shortest path and spanning tree problems. *European Journal of Operational Research*, 204(2) :303–315.

[Galvez-Fernandez et al., 2009] Galvez-Fernandez, C., Khadraoui, D., Ayed, H., Habbas, Z., and Alba, E. (2009). Distributed Approach for Solving Time-Dependent Problems in Multimodal Transport Networks. *Advances in Operations Research*, 2009.

[Geisberger et al., 2008] Geisberger, R., Sanders, P., Schultes, D., and Delling, D. (2008). Contraction hierarchies : Faster and simpler hierarchical routing in road networks. In *Experimental Algorithms*, pages 319–333. Springer.

[Geisberger et al., 2012] Geisberger, R., Sanders, P., Schultes, D., and Vetter, C. (2012). Exact routing in large road networks using contraction hierarchies. *Transportation Science*, 46(3) :388–404.

[Gen and Cheng, 2000] Gen, M. and Cheng, R. (2000). *Genetic algorithms and engineering optimization*, volume 7. John Wiley & Sons.

[Giannopoulos, 2008] Giannopoulos, G. (2008). The application of co-modality in greece : A critical appraisal of progress in the development of co-modal freight centres and logistics services. *Transition Studies Review*, 15(2) :289–301.

[Gille, 2006] Gille, A. (2006). La comodalité outil du développement durable. *Transports*, page 16.

[Glover, 1986] Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, 13(5) :533–549.

[Glover, 1989] Glover, F. (1989). Tabu search-part i. *ORSA Journal on computing*, 1(3) :190–206.

[Glover, 1990] Glover, F. (1990). Tabu search—part ii. *ORSA Journal on computing*, 2(1) :4–32.

[Goldberg and Harrelson, 2005] Goldberg, A. V. and Harrelson, C. (2005). Computing the shortest path : A search meets graph theory. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 156–165. Society for Industrial and Applied Mathematics.

[Goldberg, 1989] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition.

[Goldberg et al., 1991] Goldberg, D. E., Deb, K., and Clark, J. H. (1991). Genetic algorithms, noise, and the sizing of populations. *Complex systems*, 6 :333–362.

[Goldberg et al., 1992] Goldberg, D. E., Deb, K., and Thierens, D. (1992). Toward a better understanding of mixing in genetic algorithms. *Urbana*, 51 :61801.

[Hahne et al., 2008] Hahne, F., Nowak, C., and Ambrosi, K. (2008). Acceleration of the a*-algorithm for the shortest path problem in digital road maps. In *Operations Research Proceedings 2007*, pages 455–460. Springer.

[Hansen and Mladenović, 2001] Hansen, P. and Mladenović, N. (2001). Variable neighborhood search : Principles and applications. *European journal of operational research*, 130(3) :449–467.

[Harik et al., 1999] Harik, G., Cantú-Paz, E., Goldberg, D. E., and Miller, B. L. (1999). The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3) :231–253.

[Hart et al., 1968] Hart, P. E., Nilsson, N. J., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *Systems Science and Cybernetics, IEEE Transactions on*, 4(2) :100–107.

[Herbawi, 2012] Herbawi, W. (2012). *Solving the ridematching problem in dynamic ridesharing*. PhD thesis, Ulm, Universität Ulm, Diss., 2012.

[Herbawi and Weber, 2012] Herbawi, W. M. and Weber, M. (2012). A genetic and insertion heuristic algorithm for solving the dynamic ridematching problem with time windows. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pages 385–392. ACM.

[Hizem, 2008] Hizem, M. (2008). *Recherche de chemins dans un grpahe à pondération dynamique*. PhD thesis, Ecole Centrale de Lille.

[Hliněnỳ and Moriš, 2011] Hliněnỳ, P. and Moriš, O. (2011). Scope-based route planning. In *Algorithms–ESA 2011*, pages 445–456. Springer.

[Holand, 1975] Holand, J. H. (1975). Adaptation in natural and artificial systems. *Ann Arbor : The University of Michigan Press*.

[Holland, 1975] Holland, J. H. (1975). *Adaptation in natural and artificial systems : An introductory analysis with applications to biology, control, and artificial intelligence*. U Michigan Press.

[Holzer et al., 2009] Holzer, M., Schulz, F., and Wagner, D. (2009). Engineering multilevel overlay graphs for shortest-path queries. *Journal of Experimental Algorithmics (JEA)*, 13 :5.

[Holzer et al., 2004] Holzer, M., Schulz, F., and Willhalm, T. (2004). Combining speed-up techniques for shortest-path computations. In *Experimental and Efficient Algorithms*, pages 269–284. Springer.

[Horn, 2002] Horn, M. (2002). Multi-modal and demand-responsive passenger transport systems : a modelling framework with embedded control systems. *Transportation Research Part A : Policy and Practice*, 36(2) :167–188.

[Ishibuchi and Murata, 1998] Ishibuchi, H. and Murata, T. (1998). A multi-objective genetic local search algorithm and its application to flowshop scheduling. *Systems, Man, and Cybernetics, Part C : Applications and Reviews, IEEE Transactions on*, 28(3) :392–403.

[Jagadeesh et al., 2002] Jagadeesh, G. R., Srikanthan, T., and Quek, K. (2002). Heuristic techniques for accelerating hierarchical routing on road networks. *Intelligent Transportation Systems, IEEE Transactions on*, 3(4) :301–309.

[Jennings et al., 1998] Jennings, N. R., Sycara, K., and Wooldridge, M. (1998). A roadmap of agent research and development. *Autonomous agents and multi-agent systems*, 1(1) :7–38.

[Jung and Pramanik, 2002] Jung, S. and Pramanik, S. (2002). An efficient path computation model for hierarchically structured topographical road maps. *Knowledge and Data Engineering, IEEE Transactions on*, 14(5) :1029–1046.

[Kammoun, 2007] Kammoun, M. A. (2007). *Conception d'un système d'information pour l'aide au déplacement multimodal : Une approche multi-agents pour la recherche et la composition des itinéraires en ligne*. PhD thesis, Ecole Centrale de Lille.

[Klusch and Shehory, 1996] Klusch, M. and Shehory, O. (1996). A polynomial kernel-oriented coalition algorithm for rational information agents. *Tokoro, ed*, pages 157–164.

[Kowalczyk et al., 2006] Kowalczyk, R., Braun, P., et al. (2006). Towards agent-based coalition formation for service composition. In *Proceedings of the IEEE/WIC/ACM international conference on Intelligent Agent Technology*, pages 73–80. IEEE Computer Society.

[Laichour et al., 2001] Laichour, H., Maouche, S., and Mandiau, R. (2001). Traffic control assistance in connection nodes : multi-agent applications in urban transport systems. In *Intelligent Data Acquisition and Advanced Computing Systems : Technology and Applications, International Workshop on, 2001.*, pages 133–137. IEEE.

[Mahfoud and Goldberg, 1995] Mahfoud, S. W. and Goldberg, D. E. (1995). Parallel recombinative simulated annealing : a genetic algorithm. *Parallel computing*, 21(1) :1–28.

[Mandiau et al., 2002] Mandiau, R., Grislin-Le Strugeon, E., and Péninou, A. (2002). Organisation et applications des sma.

[Martins and Pascoal, 2003] Martins, E. Q. and Pascoal, M. M. (2003). A new implementation of yen's ranking loopless paths algorithm. *Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, 1(2) :121–133.

[Meng et al., 2007] Meng, A., Ye, L., Roy, D., and Padilla, P. (2007). Genetic algorithm based multi-agent system applied to test generation. *Computers & Education*, 49(4) :1205–1223.

[Mesghouni, 1999] Mesghouni, K. (1999). *Application des algorithmes évolutionnistes dans les problèmes d'optimisation en ordonnancement de production*. PhD thesis, Université de Lille 1.

[Michalewicz et al., 1992] Michalewicz, Z., Janikow, C. Z., and Krawczyk, J. B. (1992). A modified genetic algorithm for optimal control problems. *Computers & Mathematics with Applications*, 23(12) :83–94.

[Moore, 1959] Moore, E. F. (1959). *The shortest path through a maze*. Bell Telephone System.

[Narayanan and McIlraith, 2002] Narayanan, S. and McIlraith, S. A. (2002). Simulation, verification and automated composition of web services. In *Proceedings of the 11th international conference on World Wide Web*, pages 77–88. ACM.

[Nuortio et al., 2006] Nuortio, T., Kytöjoki, J., Niska, H., and Bräysy, O. (2006). Improved route planning and scheduling of waste collection and transport. *Expert systems with applications*, 30(2) :223–232.

[of the european communities, 2001] of the european communities, C. (2001). European transport policy for 2010 : time to decide. White paper, Bruxelles.

[officiel de l'Union européenne, 2007] officiel de l'Union européenne, J. (2007). Examen à mi-parcours du livre blanc sur les transports publié en 2001.

[Pacheco et al., 2009] Pacheco, J., Alvarez, A., Casado, S., and González-Velarde, J. L. (2009). A tabu search approach to an urban transport problem in northern spain. *Computers & Operations Research*, 36(3) :967–979.

[Pallottino and Scutella, 1998] Pallottino, S. and Scutella, M. G. (1998). Shortest path algorithms in transportation models : classical and innovative aspects. In *Equilibrium and advanced transportation modelling*, pages 245–281. Springer.

[Paraskevopoulos et al., 2008] Paraskevopoulos, D. C., Repoussis, P. P., Tarantilis, C. D., Ioannou, G., and Prastacos, G. P. (2008). A reactive variable neighborhood tabu search for the heterogeneous fleet vehicle routing problem with time windows. *Journal of Heuristics*, 14(5) :425–455.

[Pohl, 1969] Pohl, I. (1969). *Bi-directional and heuristic search in path problems*. Number 104. Department of Computer Science, Stanford University.

[Rizzoli et al., 2007] Rizzoli, A. E., Montemanni, R., Lucibello, E., and Gambardella, L. M. (2007). Ant colony optimization for real-world vehicle routing problems. *Swarm Intelligence*, 1(2) :135–151.

[Russell et al., 1995] Russell, S., Norvig, P., and Intelligence, A. (1995). A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, 25.

[Salhi, 2002] Salhi, S. (2002). Defining tabu list size and aspiration criterion within tabu search methods. *Computers & Operations Research*, 29(1) :67–86.

[Sandhlom and Lesser, 1997] Sandhlom, T. W. and Lesser, V. R. (1997). Coalitions among computationally bounded agents. *Artificial intelligence*, 94(1) :99–137.

[Sandholm et al., 1999] Sandholm, T., Larson, K., Andersson, M., Shehory, O., and Tohmé, F. (1999). Coalition structure generation with worst case guarantees. *Artificial Intelligence*, 111(1) :209–238.

[Satunin and Babkin, 2014] Satunin, S. and Babkin, E. (2014). A multi-agent approach to intelligent transportation systems modeling with combinatorial auctions. *Expert Systems with Applications*, 41(15) :6622–6633.

[Schulz et al., 2002] Schulz, F., Wagner, D., and Zaroliagis, C. (2002). Using multi-level graphs for timetable information in railway systems. In *Algorithm Engineering and Experiments*, pages 43–59. Springer.

[Sghaier, 2011] Sghaier, M. (2011). *A combination of optimization and distributed artificial intelli- gence techniques to set up a dynamic carpooling service.* PhD thesis, Ecole Centrale de Lille.

[Shehory and Kraus, 1998] Shehory, O. and Kraus, S. (1998). Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1) :165–200.

[Shen and Kwan, 2002] Shen, Y. and Kwan, R. S. (2002). Tabu search for time windowed public transport driver scheduling. *RESEARCH REPORT SERIES-UNIVERSITY OF LEEDS SCHOOL OF COMPUTER STUDIES LU SCS RR*, (13).

[Sim and Sun, 2003] Sim, K. M. and Sun, W. H. (2003). Ant colony optimization for routing and load-balancing : survey and new directions. *Systems, Man and Cybernetics, Part A : Systems and Humans, IEEE Transactions on*, 33(5) :560–572.

[Sycara, 1998] Sycara, K. P. (1998). Multiagent systems. *AI magazine*, 19(2) :79.

[Teal, 1987] Teal, R. F. (1987). Carpooling : who, how and why. *Transportation Research Part A : General*, 21(3) :203–214.

[Thierens and Goldberg, 1993] Thierens, D. and Goldberg, D. E. (1993). Mixing in genetic algorithms. *Urbana*, 51 :61801.

[Tong et al., 2009] Tong, H., Cao, J., Zhang, S., and Li, M. (2009). A distributed agent coalition algorithm for web service composition. In *Services-I, 2009 World Conference on*, pages 62–69. IEEE.

[Tsubakitani and Evans, 1998] Tsubakitani, S. and Evans, J. R. (1998). Optimizing tabu list size for the traveling salesman problem. *Computers & Operations Research*, 25(2) :91–97.

[Ulungu and Teghem, 1994] Ulungu, E. and Teghem, J. (1994). Multi-objective combinatorial optimization problems : A survey. *Journal of Multi-Criteria Decision Analysis*, 3(2) :83–104.

[Van Nes, 2002] Van Nes, R. (2002). Design of multimodal transport networks. *Civil Engineering. Delft Technical University, Delft*, page 304.

[Wang and Kaempke, 2004] Wang, J. and Kaempke, T. (2004). Shortest route computation in distributed systems. *Computers & Operations Research*, 31(10) :1621 – 1633.

[Wang et al., 2006] Wang, X.-b., Jun, L. Y., and Ying, S. J. (2006). Research on vrp of optimization based on improved two phase algorithm under electronic commerce. In *Management Science and Engineering, 2006. ICMSE'06. 2006 International Conference on*, pages 493–497. IEEE.

[Wang et al., 2014a] Wang, Z., Mesghouni, K., and Hammadi, S. (2014a). Agent-based coalition formation in a co-modal transport system. In *Proceedings of the IEEE/-WIC/ACM international conference on Intelligent Agent Technology*, pages 310–317. IEEE Computer Society.

[Wang et al., 2014b] Wang, Z., Mesghouni, K., and Hammadi, S. (2014b). A co-modal transport information system in a distributed environment. *International Journal of Advanced Computer Science and Information Technology*, 3 :83–99.

[Williams, 1964] Williams, J. W. J. (1964). Algorithm-232-heapsort.

[Wooldridge and Jennings, 1995] Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents : Theory and practice. *The knowledge engineering review*, 10(02) :115–152.

[Yen, 1971] Yen, J. Y. (1971). Finding the k shortest loopless paths in a network. *management Science*, 17(11) :712–716.

[Yu and Lu, 2012] Yu, H. and Lu, F. (2012). A multi-modal route planning approach with an improved genetic algorithm. *Adv. Geo-Spatial Inform. Sci*, 38 :193–202.

[Yuan and Wang, 2009] Yuan, Y. and Wang, D. (2009). Path selection model and algorithm for emergency logistics management. *Computers & Industrial Engineering*, 56(3) :1081–1094.

[Zidi, 2006] Zidi, K. (2006). *Système Interactif d'Aide au Déplacement Multimodal*. PhD thesis, Ecole Centrale de Lille.

[Zlotkin and Rosenschein, 1994] Zlotkin, G. and Rosenschein, J. S. (1994). Coalition, cryptography, and stability : Mechanisms for coalition formation in task oriented domains. In *AAAI*, volume 432, page 437.

Titre : Optimisation avancée pour la recherche et la composition des itinéraires comodaux au profit des clients de transport

Résumé : Avec les problèmes présents dans le secteur de transport, qu'ils soient financiers ou environnementaux, la mobilité avancée peut y remédier avec la mise à profit de la complémentarité entre les différents modes de transport. Dans ce contexte, nous nous focalisons dans cette thèse à la mise en œuvre d'un système d'information de transport avec la recherche et la composition des itinéraires comodaux pour les clients. L'enjeu est d'être capable de répondre aux attentes des usagers avec des solutions satisfaisantes permettant de proposer des itinéraires optimaux pour gérer efficacement l'intermodalité. Dans un souci pratique, nous fournirons des itinéraires attractifs respectant les contraintes imposées même pour les requêtes simultanées. Nous utilisons des techniques d'accélération permettant de réduire l'espace de recherche pour la planification d'itinéraire. Les itinéraires attractifs sont décomposés en sections de route sur lesquelles les différentes demandes et les offres disponibles sont mises en relation. Les combinaisons des sections de route permettent d'aboutir à un ensemble de solutions intéressantes. L'aspect distribué et dynamique du problème nous a permis d'employer une modélisation basée sur le paradigme agent. Ainsi, l'alliance entre les systèmes multi-agents et les algorithmes génétiques que nous avons mis en place s'avère très utile pour gérer l'articulation de l'intermodalité entre ces différents modes de transport. Les résultats de simulation présentés montrent l'efficacité des méthodes proposées.

Mots clés : théorie de graphe, plus court chemin, optimisation, Système Multi-Agents, coalition des agents, mobilité avancée

Title : Design and implementation of a traveller information system : an agent-based method for searching and composing itineraries

Abstract : Nowadays, the environment impact of transport is significant. In an attempt to address these problems, in this work, we are interested in the implementation of a transport information system, which integrates the existing means of transport to respond users' requests, including public transport and the shared transport like carpooling and car-sharing. In this context of application, we elaborate algorithms to provide attractive paths with respect to the imposed constraints, even for simultaneous requests. Different acceleration techniques for path planning are used to reduce the search space for a better performance. The attractive paths are divided into route sections on which the available offers are allocated to different requests, which is treated as one resource allocation problem using metaheuristics algorithms. With consideration of the distributed and dynamic aspects of the problem, the solving strategy makes use of several concepts like multi-agents system and different optimization methods. The proposed methods are tested with realistic scenarios with instances extracted from real world transport networks. The obtained results indicate that our proposed approaches can efficiently solve the itinerary planning problems by providing good and complete solutions.

Key words : graph theory, shortest path problem, optimization, Multi-Agent System, agent coalition, shared transport