Vers une sémantique floue : application à la géolocalisation

Mohammed-Amine Abchir

Université Paris 8 – Vincennes Saint-Denis

CHArt – EA4004
(Cognition Humaine et Artificielle)

Deveryware
Département technique

Vers une sémantique floue : application à la géolocalisation

THÈSE

présentée et soutenue publiquement le 25 novembre 2013

pour l'obtention du

Doctorat de l'université Paris 8 – Vincennes Saint-Denis

(spécialité informatique)

par

Mohammed Amine Abchir

Composition du jury

Président :	Ivan Lavallée	Professeur émérite, Université Paris 8
Rapporteurs :	Luis Martínez-López	Professeur, Université de Jaén, Espagne
	Violaine Prince	Professeur, Université Montpellier 2
Directeurs :	Isis Truck	Maître de Conférences HDR, Université Paris 8
	Anna Pappa	Maître de Conférences, Université Paris 8
Examinateurs :	Marc Bui	Professeur, Université Paris 8
	Max Silberztein	Professeur, Université de Franche-Comté
Encadrant industriel :	Thierry Bathias	Directeur technique, Deveryware

École Doctorale CLI (Cognition – Langage – Interaction)

Remerciements

Ce travail n'aurait jamais abouti sans l'aide et le soutien au quotidien de plusieurs personnes que je tiens tout particulièrement à remercier ici.

Tout d'abord, je tiens à exprimer mon infinie gratitude à ma directrice de thèse Isis Truck et mon encadrante Anna Pappa. Je les remercie pour leur gentillesse, le professionnalisme dont elles ont fait preuve à mon égard et pour la bonne humeur qu'elles ont su maintenir tout au long de la thèse. Elle m'ont beaucoup apporté aussi bien sur le plan scientifique que sur le plan humain et nous avons tissé des liens d'amitié inestimables à mes yeux.

Je remercie sincèrement mes deux rapporteurs, les Professeurs Violaine Prince et Luis Martínez-López pour avoir accepté cette lourde tâche, et je souhaite en particulier exprimer ma profonde reconnaissance au Professeur Violaine Prince pour ses conseils extrêmement précieux, sa bienveillance, sa relecture en profondeur de mon manuscrit et ses remarques particulièrement pertinentes qui ont permis de rendre lisible, fluide et cohérent ce manuscrit.

Je remercie également l'ensemble des membres de mon laboratoire pour l'accueil qu'ils m'ont réservé ainsi que pour les discussions enrichissantes que j'ai pu entretenir avec eux.

Je tiens aussi à remercier l'ensemble des employés de la société Deveryware. Je remercie en particulier la direction générale en la personne de Jacques Salognon et Stéphane Schmoll pour m'avoir accepté au sein de Deveryware. J'adresse mes remerciements les plus chaleureux à l'ensemble de l'équipe technique au sein de laquelle j'ai évolué durant cette thèse. En particulier, Thierry Bathias, Romuald du Song, William Delanoue, Alexandre Ardhuin et Sylvain Maucourt qui ont supporté, sourire aux lèvres, mes interminables sollicitations. Enfin, je remercie Florence Kossoff, Sihame El Amine et Laurent Cellier qui ont su faire de mes journées chez Deveryware un vrai bonheur au quotidien.

Finalement, j'adresse mes sentiments les plus chers à ma famille et mes amis pour leur soutien inestimable et pour avoir supporté avec patience mes interminables monologues. J'adresse mon plus grand respect à ma mère Fatima et à mon défunt père Ali envers qui j'ai une grande pensée en écrivant ces lignes. Je tiens également à remercier les membres de ma famille sans qui je ne serais pas là où je suis aujourd'hui : Touria, Hiba, Mostapha, Yacine, Dounia, Nizar, Camilia, Sophia et Sara. Je n'oublie pas non plus de remercier mes amis : Nabil, Mourad, 7amza, Sma3il, Samir, Kamel, Nada, Sandra et Salsabil.

"Ce n'est pas l'abondance des biens qui fait la richesse, mais la vraie richesse est celle de l'âme"
- Mohammed

Table des matières

Table des figures

Liste des tableaux

Introduction

Le raisonnement humain a toujours été une source d'inspiration dans le monde de la recherche. En particulier, une grande partie des travaux de recherche en Intelligence Artificielle (IA) [1] vise à représenter les connaissances, imiter le raisonnement humain, comprendre les langues naturelles ou pourvoir les machines d'une capacité de perception. Le but de tous ces travaux est de concevoir une machine, à l'image de l'homme qui soit dotée d'une intelligence, capable de raisonner et ayant une "conscience", et notamment une *conscience de soi*.

Depuis les premiers travaux remontant à Alan Turing sur la machine de Turing universelle, l'histoire a enregistré plusieurs avancées dans diverses branches de l'IA qui ont donné lieu à des projets aussi notables qu'ambitieux. Le programme MYCIN développé dans les années 1970 a constitué un grand pas pour la recherche en IA dans le domaine médical. MYCIN était capable de diagnostiquer un grand nombre d'infections en déduisant la bactérie qui en était responsable ainsi que l'antibiotique adéquat selon le patient dont il était question. Il était fondé sur un système d'inférence et des règles prédéfinies afin d'assister le médecin. MYCIN fut un des premiers systèmes experts connus.

Le jeu vidéo est un domaine où beaucoup de travaux ont été menés, soit afin d'améliorer l'expérience utilisateur (comme pour les jeux de simulation ou les Role Playing Game (RPG) (jeux de rôle) qui sont un type de jeux où le joueur incarne un héros dont il fait évoluer les capacités durant une quête), soit pour créer des programmes capables de battre des joueurs humains. C'est ainsi qu'en 1997, Deep Blue, un superordinateur spécialisé dans le jeu d'échecs, a réussi à battre le champion du monde Garry Kasparov dans une partie à six manches. Et malgré plusieurs contestations de Kasparov, ce fait a constitué une première mondiale qui a nécessité un supercalculateur d'une puissance de 11.38 GFLOPS [2]. Mais pour d'autres jeux comme le *go*, il n'existe toujours pas de programme capable de battre un joueur professionnel et ce, malgré la nette amélioration apportée par l'utilisation de la méthode de Monte-Carlo. Ceci est dû à la grande complexité induite par le nombre conséquent de coups possibles à chaque tour.

Malgré toutes ces avancées, le rêve des chercheurs en IA était loin d'être atteint. En effet, même si MYCIN et Deep Blue ont aidé à diagnostiquer des infections pour l'un, et à battre un joueur humain pour l'autre, ils n'avaient aucune "conscience" de leurs actions. Deep Blue, par exemple, n'avait pas conscience, lors de la partie, qu'il jouait

1. Tous les acronymes sont repris dans une table spécifique, page 119.
2. 1 GFLOPS correspond à 1 gigaFLOPS. Le FLOPS (FLoating point Operations Per Second) est une unité de mesure de la puissance de calcul brute d'un ordinateur.

contre le champion du monde en titre. Aussi, ses capacités n'évoluaient pas au fil des matchs et il ne pouvait en aucun cas apprendre de nouveaux coups de ses adversaires. Ainsi, l'apprentissage automatique et l'exploration de données ont évolué en parallèle afin d'apporter des réponses à ces attentes. L'objectif est de donner aux programmes (et donc aux machines) une capacité d'analyse, une faculté à emmagasiner de la connaissance et à être capable d'adapter son comportement en fonction de nouveaux facteurs qu'il est susceptible de rencontrer. En 2012, le *X lab* de Google a présenté un réseau de neurones, appelé *Brain* [Le et al., 2012], capable de détecter la présence de schémas significatifs (visage ou corps humain, chats...) sur des images. La particularité de *Brain* est que pendant la phase d'apprentissage, aucune information sur la présence ou non de schémas ne lui a été donnée sur les dix millions d'images qu'il a parcourues. Suite à l'apprentissage, *Brain* a été capable de détecter "de lui même" la présence de visages humains et de chats sur une série de vingt mille images avec un taux de succès allant de 74.8% à 81.7%. *Brain* a donc développé le *concept* de chat et lui a lié une certaine *sémantique* qui lui permet de le reconnaître par la suite. Ceci représente une avancée majeure puisque, jusque-là, l'apprentissage des notions sémantiques était toujours assisté par l'homme.

De nos jours, les téléphones intelligents[3] connaissent, et ce, depuis déjà quelques années, une très forte adoption par les utilisateurs (plus d'un milliard d'utilisateurs dans le monde en 2012) et, avec eux, sont venus les assistants personnels. Ces programmes intelligents sont l'aboutissement des travaux d'IA en divers domaines comme le Traitement Automatique du Langage Naturel (TALN), la synthèse vocale, la reconnaissance vocale, la fouille de données, etc. Ces travaux découlent de ceux initiés en 1966 par Joseph Weizenbaum avec le programme ELIZA [Weizenbaum, 1966]. ELIZA simulait un psychothérapeute *via* une conversation écrite. Mais n'ayant pas pour but de donner des réponses à l'utilisateur et n'ayant aucune "vraie" notion sémantique de ses conversations, ELIZA se contentait de reformuler les affirmations qu'elle recevait en question afin de relancer la discussion dans le cadre d'un stimulus réponse [Weizenbaum, 1966].

Ces travaux et les suivants se sont inspirés des approches des linguistes concernant la sémantique et la pragmatique. La notion de "sémantique" telle qu'elle est définie dans l'introduction de *The philosophy of Language* [Martinich, 1985] est l'étude des significations des expressions linguistiques, c'est-à-dire l'étude des relations entre les mots et le monde. Avec la sémantique, on cherche donc à établir des relations entre les mots et les choses. Or, la *signification* d'un mot est, par essence, vague et ambigüe puisque l'on peut donner différents types de sens à une même notion (sémantique).

Par ailleurs, les linguistes font référence à la *pragmatique* comme la discipline qui étudie *comment* le langage est utilisé. Les éléments du langage doivent être pris contextuellement puisqu'ils ne peuvent être compris qu'en connaissant le *contexte* dans lequel ils sont employés.

Dans le domaine de l'IA, les recherches en TALN se sont longtemps focalisées sur la volonté de donner aux machines des capacités de "compréhension" et la difficulté a toujours été de représenter la sémantique au niveau machine. La plupart des approches sont fondées sur l'encodage manuel des données textuelles avec des techniques statistiques pour créer des lexiques, mais sans résoudre les problèmes de polysémie ni de synonymie.

3. c'est-à-dire les *smartphones*.

Les assistants personnels actuels comme Siri d'Apple ou Google Now, et contrairement à ELIZA, "comprennent" le sens des phrases. Ils sont capables d'analyser les phrases sémantiquement et d'interagir en conséquence avec l'utilisateur. Ils peuvent ainsi lire ou envoyer des mails ou SMS, répondre à des questions en se fondant sur un moteur de recherche ou encore commander et paramétrer le téléphone. Google Now va encore plus loin en intégrant un moteur de recommandations capable, en prenant en considération les habitudes de l'utilisateur, d'anticiper ses besoins en lui proposant certaines actions comme la réservation d'un billet de cinéma, la programmation d'un itinéraire de navigation par Global Positioning System (GPS), la proposition d'un événement culturel intéressant, etc. Notons qu'il n'est pas question ici de juger de l'intrusion ou non dans la vie privée de ces appareils, mais simplement de comprendre les avancées actuelles de ce genre de technologies.

Malgré le fait que ces assistants personnels nous rapprochent encore plus du but premier de l'IA qui est la création d'un programme simulant un humain, la création d'un assistant polyvalent nécessite une base de connaissances et une puissance de calcul conséquentes afin de permettre de réaliser de façon transparente les traitements sémantiques nécessaires pour simuler le raisonnement humain [Singhal, 2012].

De plus, aux limitations en termes de ressources s'ajoutent également des limitations conceptuelles. En effet, le cerveau humain possède une capacité d'abstraction inégalée lui permettant de traiter des données y compris quand celles-ci sont imprécises voire incomplètes. Ainsi, la modélisation des connaissances doit tenir compte, au mieux, de ces imprécisions afin de rester la plus fidèle possible aux nuances et incertitudes utilisées chez l'homme. Il est donc préférable de privilégier une modélisation qualitative plutôt que quantitative en utilisant des valeurs linguistiques plutôt que des valeurs numériques précises. L'approche floue a apporté des réponses à ces aspirations en introduisant le concept de variable linguistique [Zadeh, 1975]. Une variable linguistique est une représentation d'une valeur linguistique par des mots qui puisent leur sémantique dans des sous-ensembles flous. Il a été ainsi possible de construire des raisonnements approximatifs où les notions, comme la distance par exemple, peuvent être décrites par des valeurs comme *faible*, *bas*, *moyen*, *haute*, *forte*, etc. Les sous-ensembles flous dans ce cas-là prennent en charge la modélisation de ces valeurs et les calculs sont rendus possibles grâce au modèle de calcul sous-jacent.

La question de la géolocalisation, déjà évoquée dans les logiciels de navigation offerts par les assistants personnels, est également très présente, et, de plus en plus, dans les applications actuelles. Cette technologie s'est très nettement popularisée et est même devenue sans doute l'un des enjeux majeurs de notre société, tant du point de vue scientifique, que commercial, éthique, etc.

On souhaite géolocaliser des véhicules, des personnes, des biens, ou plus généralement des mobiles pour gagner du temps, de l'argent, gagner en productivité, en communication, en sécurité, etc. Ainsi, depuis l'avènement des outils de géolocalisation, notamment le GPS grâce à la constellation de satellites NAVSTAR [4], le système russe GLONASS [5]

4. NAVSTAR pour *NAVigation Satellite Timing And Ranging*.
5. GLONASS, acronyme russe de "Système global de navigation satellitaire".

puis Galileo [6], préfiguré par EGNOS [7], le domaine des applications militaires n'a plus été le seul secteur concerné comme c'était initialement le cas. En effet, le domaine civil a largement bénéficié de ces avancées depuis 1995 avec des applications professionnelles et grand public comme la navigation maritime, la navigation routière, les opérations de secours et de sauvetage, les travaux publics, la logistique, le transport, la sécurité, la collecte des taxes environnementales, la publicité, etc. et également toutes les applications de la vie de tous les jours (recherche et suivi d'itinéraire, recherche d'un restaurant, d'un hôtel, repérage lors de randonnées, recommandation de services entourant l'utilisateur, promotions dans des magasins à proximité, etc.). Ainsi, progressivement sont apparues des entreprises spécialisées dans la géolocalisation et proposant différentes technologies pour localiser les personnes, les marchandises et les véhicules. Les solutions proposées répondent à la question "qui, où et quand ?", c'est-à-dire **qui** est **où** au **moment** t, et mettent en œuvre des fonctions géodépendantes comme des envois de messages ou d'alertes pour prévenir de la position du mobile, d'un éventuel problème décelé, d'un retard, etc. Parmi les problèmes que ces solutions doivent surmonter se trouve en particulier la gestion de ces envois et notamment la question cruciale du paramétrage du suivi pour qu'il soit effectif et efficace.

On note donc un certain nombre de verrous scientifiques et technologiques :
- Comment faire pour ne pas envoyer trop ou trop peu de messages ?
- Comment faire en sorte qu'un message soit toujours envoyé dès que c'est nécessaire et seulement à ce moment-là ?
- Comment éviter une panne de mobile, et en particulier, comment faire pour économiser les batteries des mobiles suivis ?
- Comment paramétrer de la façon la plus automatisée possible le suivi des mobiles ?
- Comment traduire correctement les *desiderata* des clients, en particulier lorsque ces derniers ne sont pas spécialistes du domaine ?

Un des éléments essentiels utilisés par la géolocalisation est la notion de **position**, exprimée dans un système (géodésique pour le GPS, relatif à un ensemble de stations appelées Base Transceiver Stations (BTSs) pour une localisation par la norme GSM [8], etc.). Cette position est nécessairement entâchée, peu ou prou, d'imprécision, et même, parfois, elle peut être carrément manquante. Si les théories relatives notamment à l'approche floue sont désormais bien ancrées et ont donné lieu à de multiples applications, elles ne semblent pas suffisantes pour résoudre intégralement les verrous évoqués. En effet, les imprécisions peuvent être dues au vocabulaire employé pour exprimer le lieu recherché, ou pour exprimer la distance à ce lieu. Elles peuvent aussi être liées au capteur ayant donné l'information de position. Et tout ceci est également relatif au contexte dans lequel on se trouve. Aucune solution n'est aujourd'hui disponible pour résoudre les problèmes évoqués dans le contexte de la géolocalisation.

De même, les travaux actuels sur le langage naturel et son traitement ne se sont jamais attaqués, à notre connaissance, à des questions de paramétrage automatique de systèmes

6. Galileo est un projet européen lancé en 2005 qui devrait proposer à partir de 2014 un système de positionnement et de datation par satellites européen de deuxième génération.

7. EGNOS pour *European geostationary navigation overlay system* a été lancé en 2009 principalement pour la navigation aérienne car supposé fournir un très bonne précision verticale.

8. *Global System for Mobile communications*

de géolocalisation, ni à la possibilité de traduire des besoins métiers de non spécialistes en configuration des systèmes de géolocalisation à bas niveau et pas davantage à la prise en compte de données imprécises ou manquantes dans un tel contexte.

Pour notre part et dans le cadre d'un travail en milieu industriel lié à ces problématiques de géolocalisation de mobiles, en combinant la **sémantique** avec la **logique floue**, nous cherchons à obtenir une approche dans laquelle un sous-ensemble flou représente, dans une théorie de la sémantique du langage naturel, la signification d'une expression vague. Ainsi, l'interprétation sémantique des données textuelles serait le processus d'analyse d'un texte balisé avec des marqueurs pour représenter sa signification (à l'aide des outils de la logique floue), où l'entrée du système est un arbre analysé syntaxiquement et grammaticalement et où la sortie est la signification de cet arbre. Ce travail doit permettre d'améliorer la qualité du suivi des mobiles, ainsi que proposer des interfaces de communication adéquates permettant le suivi, à bas niveau (*via* par exemple, des assistants personnels), afin de lever les verrous évoqués plus haut.

La suite du document est organisée en cinq chapitres.

Dans le chapitre 1, nous rappelons le contexte dans lequel s'est effectuée la thèse et les besoins industriels exprimés dans le cadre du projet SALTY dans lequel s'insèrent nos travaux.

Le chapitre 2 présente l'état de l'art des approches se rapportant à la logique floue et la modélisation de données imprécises, celles liées au traitement automatique du langage naturel ainsi que celles explorant les articulations entre les deux premiers types d'approches.

Le chapitre 3 décrit les actions préalables d'analyse et de constitution des corpus relatifs aux besoins exprimés. En particulier, nous présentons un lexique métier, une grammaire dédiée, ainsi qu'une approche d'agent intelligent de dialogue. Un prototype est également introduit.

Ensuite, le chapitre 4 présente notre proposition de modélisation de données imprécises fondée sur ce que nous avons appelé les 2-tuples sémantiques. Dans la première partie, nous présentons les 2-tuples en question, la façon de les utiliser pour obtenir un partitionnement flou ainsi que le modèle de calcul correspondant. Dans une deuxième partie, nous présentons une interprétation sémantique floue des termes linguistiques issus du dialogue, fondée sur les 2-tuples sémantiques. Nous finissons ensuite par une discussion autour du lien entre les divers outils de modélisation floue.

La chapitre 5 est consacré à la mise en œuvre et aux résultats du travail. D'abord, l'implémentation des 2-tuples sémantiques est explicitée à l'aide notamment d'exemples et de comparaison avec les 2-tuples linguistiques. Ensuite, après avoir étudié la complexité de l'algorithme principal de partitionnement, nous présentons la façon dont les travaux ont été intégrés dans l'entreprise dans laquelle s'est déroulée la thèse.

La dernière partie conclut nos travaux et donne quelques perspectives de recherche à plus ou moins long terme.

Chapitre 1

Contexte

Sommaire

Pour décrire le contexte scientifique et applicatif, général et particulier de cette thèse, il faut partir du besoin et du problème de l'entreprise avec laquelle nous avons travaillé. L'entreprise en question, Deveryware, est une société spécialisée dans la géolocalisation (ou chrono-localisation) qui offre la possibilité à ses clients de localiser en temps réel des personnes, biens, matériels ou marchandises. Le problème que nous devons traiter est celui de la création d'une *alerte* (pour *alerter* le client), non définie à l'avance, et qui doit se faire sur trois critères : sortie d'une zone physique, état du mobile (batterie) ou présence d'un autre mobile dans une zone.

1.1 Contexte scientifique général

Le contexte scientifique dans lequel on se place est celui du traitement des données imprécises. Ce que l'on appelle une donnée imprécise est une information approximative, vague, indéterminée, floue... qui ne donne qu'une idée générale de ce que l'on souhaite manipuler. Typiquement, une expression linguistique comme "Il est âgé" est imprécise car la donnée *âgé* l'est. Que signifie, en termes de nombre d'années, *âgé* ? 70 ans ? 80 ans ? 30 ans ? Il est évident que tout dépend du **contexte** et du point de vue dans lequel on se place.

Une personne de 70 ans est en principe, en retraite, donc retirée, toujours en principe, définitivement de la vie active, donc pouvant être considérée comme quelqu'un de trop avancé dans l'âge pour pouvoir être assez productif sur le marché du travail. Dans ce sens-là, quelqu'un de 70 ans peut sans doute être qualifié d'*âgé*.

Une personne de 80 ans, lorsqu'il s'agit d'un homme, a atteint et même dépassé l'espérance moyenne de vie d'une personne de sexe masculin en France, en 2010, qui est

de 78 ans environ. Ainsi, un homme français, vivant en 2013, et ayant 80 ans, peut être considéré comme *âgé* à ce titre.

Une personne de 30 ans, s'il s'agit d'un étudiant en master ou même en doctorat, peut être considéré comme ayant dépassé la moyenne d'âge des étudiants de son niveau d'études. A ce titre, il peut être considéré comme une personne *âgée*.

D'autres types de données peuvent être également imprécises, alors même qu'elles se voudraient précises. C'est typiquement le cas des données issues d'appareils de mesure comme les *capteurs*, pour lesquels trois types d'erreurs peuvent apparaître, dues :

- aux appareils eux-mêmes qui sont, par essence, imparfaits ;
- aux défauts de la méthode de mesure ;
- aux éventuelles erreurs de lecture.

On peut pallier les erreurs dues aux appareils en connaissant la classe de précision (c'est-à-dire l'incertitude relative sur une mesure égale au calibre [9]) et ainsi anticiper l'écart entre la valeur lue et la valeur vraie.

Les deux autres types d'erreur sont en général évités par des comparaisons entre la valeur et la moyenne, par exemple, des dernières valeurs reçues. Il reste qu'il est indispensable de faire subir un traitement aux données reçues avant de les transmettre au système.

Par ailleurs, les données reçues par captation peuvent, elles aussi, être fortement dépendantes du contexte et être interprétées de façon très différente selon les cas. Donc les imprécisions apparaissent à plusieurs niveaux pour cette sorte de données.

Un troisième type de données peut également être imprécis : il s'agit tout simplement des mots, au sens large. En effet, ils n'ont aucune (ou presque aucune) "bonne" propriété mathématique comme la transitivité, la bijection,... Par exemple, un mot a qui est synonyme d'un mot b n'est pas forcément synonyme du mot c, c étant pourtant un synonyme de b. Ou encore, un mot a synonyme de b n'implique pas forcément que b est également synonyme de a (relations d'inclusion, par exemple, "train" et "voiture"). La polysémie d'un mot est également source d'ambiguïté, donc d'imprécision puisque le sens du mot est, d'un certain point de vue, mal défini, puisque le mot est à plusieurs sens.

Or, comme on va le voir maintenant, le matériau que l'on doit manipuler dans le cadre de cette thèse est caractérisé, justement, et en particulier, par l'imprécision.

9. Le calibre d'un appareil est la plus forte intensité que ce dernier peut mesurer. Les appareils sont souvent multi-calibres et autocalibrables pour permettre la mesure la meilleure, quelle que soit l'intensité mesurée.

1.2 Contexte applicatif général

Ce travail s'inscrit dans un contexte applicatif qui est très fortement lié à la notion de géolocalisation, génératrice de données imprécises.

Schématiquement, la géolocalisation utilise les informations de positionnement des mobiles afin d'en assurer le suivi dans le temps et dans l'espace pour utiliser ces informations à des fins spécifiques à chaque application et chaque domaine métier.

La géolocalisation implique au moins deux entités : un dispositif de positionnement et un système de suivi. Quel que soit leur type, les dispositifs de positionnement suivent des modes opératoires assez similaires. Ils déterminent leurs positions puis transmettent l'information à un système de suivi. La plupart du temps, ils utilisent les réseaux mobiles à cet effet, mais d'autres alternatives existent (réseaux satellitaires ou encore Wifi).

La localisation d'un mobile se fait essentiellement par trois technologies qui peuvent être complémentaires :

- **GPS :** la localisation par GPS se fait à l'aide de boîtiers spécifiques équipés de puces GPS capables de calculer leur position en temps réel. Ces boîtiers sont embarqués sur les véhicules (en logistique par exemple) ou directement portés par les personnes suivies (personnes âgées ou travailleurs isolés par exemple). Ils sont le plus souvent munis d'une puce GSM qui leur permet d'être communicants, notamment en transmettant leur position à une plateforme (appelée Geohub, dans notre cas, voir plus loin) en temps réel, soit par SMS soit *via* le réseau GPRS. Cette méthode de localisation assure une **précision allant de 10 à 100 mètres** selon les conditions de réception du signal GPS ;
- **Cell-Id :** la localisation par Cell-Id concerne tout mobile connecté aux réseaux de téléphonie des opérateurs mobiles. Le mobile est géolocalisé par l'opérateur suivant l'antenne cellulaire à laquelle il est connecté. Selon la zone de couverture de l'antenne cellulaire, cette méthode offre une **précision allant de 300 mètres (en zone urbaine) à plusieurs kilomètres (en zone rurale)** ;
- **Wifi :** à l'image de la localisation par Cell-Id, la localisation par Wifi se fait par le biais de l'identifiant de la borne Wifi à laquelle le mobile est connecté. La localisation Wifi nécessite de créer et de tenir à jour de grandes bases de données où sont stockés les identifiants des bornes ainsi que leur position. Cette méthode de localisation offre une **précision allant de 30 à 100 mètres**.

Les dispositifs de positionnement peuvent être fixes ou mobiles. En effet, certains boîtiers GPS spécifiques peuvent être fixés sur un camion, par exemple, et alimentés par celui-ci. De tels dispositifs sont le plus souvent connectés aux différents capteurs du véhicule (température moteur, vitesse, niveau des réservoirs, ouverte/fermeture des portes, etc.) et peuvent en transmettre les données (en plus de données de positionnement).

Ceci dit, les boîtiers GPS mobiles sont de plus en plus utilisés grâce à la flexibilité et la facilité d'installation qu'ils offrent. De ce fait, étant donné qu'ils sont munis de batteries pour fonctionner, la consommation d'énergie devient une **ressource critique à gérer**.

Pour cela, deux paramètres principaux doit être pris en compte lors de la conception d'une application en géolocalisation :

- **fréquence de mesure :** représente la fréquence à laquelle le boîtier calcule sa position actuelle ;
- **fréquence de transmission :** représente la fréquence à laquelle le boîtier transmet sa position au système de suivi.

La géolocalisation, par conséquent, implique la prise de décision concernant ces deux fréquences car chaque mesure ou transmission a un coût énergétique. Ces décisions sont actuellement prises de façon statique et définitive, ce qui ne correspond pas toujours aux besoins des clients/utilisateurs finaux.

Ainsi, il existe certains boîtiers mobiles capables d'optimiser leur consommation d'énergie en allumant et en mettant en veille les systèmes de positionnement et de transmission de données selon les besoins. Ils peuvent également se mettre en veille et ne se remettre en marche que suivant un planning horaire défini, ou encore suite à la détection d'un événement particulier comme la détection de mouvement par exemple.

Le système de suivi, quant à lui, a la responsabilité de "surveillance" des positions des mobiles, de leur stockage et de leur traitement. Ces traitements incluent, par exemple, l'agrégation et le filtrage de positions.

Les systèmes de suivi de positions peuvent être embarqués directement dans les applications d'utilisateurs finaux, comme par exemple pour les boîtiers de guidage GPS, ou bien s'appuyer sur des plate-formes dédiées pour corréler les informations à partir de plusieurs dispositifs de positionnement et déclencher des événements particuliers envoyés aux applications des utilisateurs finaux. Ces plate-formes ont typiquement un rôle de CEP (pour *Complex Event Processing*) [Chakravarthy et Jiang, 2009].

Les apports du système de suivi peuvent être résumés en deux points essentiels :

- offrir aux applications reposant sur le système en question, une interface standard d'accès aux données les abstrayant ainsi que des particularités et spécificités de chaque type de boîtier et chaque technique de localisation ;
- corréler efficacement les informations de géolocalisation de plusieurs mobiles en même temps et permettre ainsi de détecter des événements de plus en plus complexes afin d'en avertir les applications.

Ainsi, la conception d'une application de géolocalisation, et vu tous les aspects que nous avons évoqués, se trouve confrontée à deux difficultés en particulier :

- la corrélation des positions des mobiles avec les événements à signaler ainsi que la configuration des appareils de positionnement pour les remontées des positions reposent sur une multitude de paramètres techniques. Ces paramètres ne sont pas toujours faciles à comprendre et à ajuster par les utilisateurs finaux ;
- une application peut comporter l'enchaînement d'un grand nombre de types d'événements différents à détecter, possédant chacun ses caractéristiques se déclinant en paramètres de géolocalisation particuliers. La définition d'une application correcte

et complète est rendue difficile par ce grand nombre et les interactions qui peuvent exister entre eux.

La difficulté est amplifiée par le fait que chaque requête de localisation ou chaque remontée de position a un coût, et donc que toute application se construit avec pour objectif de minimiser le coût global de géolocalisation tout en respectant les besoins et objectifs de l'application.

Pour cela, il faut savoir réduire la fréquence de ces opérations à chaque fois que cela est possible, ce qui demande de faire une adaptation dynamique des paramètres à la situation courante des mobiles à localiser.

Par ailleurs, des conflits ou incohérences peuvent surgir dans la mise à jour des paramètres ou même dans le choix des paramètres à mettre à jour, notamment lorsque la géolocalisation d'un même élément mobile intéresse plusieurs utilisateurs finaux (suivi de différents colis dans un même véhicule, par exemple). Il faut donc, dans ce cas, envisager des méthodes issues du *Multi Criteria Decision Making* (MCDM) (pour prise de décision multi-critères), visant à agréger les réponses locales pour fournir une réponse globale intégrant contraintes et préférences.

1.3 Contexte applicatif particulier

1.3.1 Métier de Deveryware

Cette thèse s'effectue dans le cadre d'une convention CIFRE [10] avec l'entreprise Deveryware.

Deveryware, pour localiser les mobiles, s'appuie principalement sur sa plate-forme multi-technologies, le Geohub, qui collecte les données (positions, niveau de batterie, vitesse du mobile, etc.) de diverses sources comme les boîtiers GPS, les opérateurs mobiles pour les données cellulaires (Cell-id GSM), les bornes Wifi ou encore les tags RFID. Ces données, après avoir été formatées et stockées, sont mises à disposition des clients *via* un service Web sécurisé. Le Geohub agit ainsi comme un *middleware* [11] entre les applications clientes et les mobiles localisés.

Le cœur de métier de Deveryware est la possibilité de créer des *alertes*. Les alertes sont programmées sur le Geohub par les utilisateurs afin que ces derniers soient notifiés (par SMS ou mail) lorsqu'un événement en particulier survient. La configuration d'une alerte se fait *via* une interface Web de type formulaire pré-rempli. Par exemple, un utilisateur peut demander à être prévenu par SMS qu'un mobile en particulier sort d'une zone géographique définie. Ainsi, le Geohub évalue si une notification doit être transmise à l'utilisateur ou non à chaque fois qu'une nouvelle position du mobile en question est disponible.

10. Convention Industrielle de Formation par la REcherche

11. Un *middleware* (aussi appelé *intergiciel*) est un logiciel agissant comme intermédiaire d'échange de données entre plusieurs logiciels tiers.

Dans ce rôle de gestionnaire d'alertes, le Geohub est un CEP.

Il existe plusieurs types d'alertes qui peuvent être créées par les utilisateurs clients de Deveryware. Les alertes les plus utilisées sont les suivantes :

- **entrée/sortie de zone :** le déclenchement de l'alerte se fait quand le mobile cible entre dans (ou sort d') une zone définie par l'utilisateur. Une zone est définie par un point central (en fournissant une adresse postale ou directement la longitude et la latitude) et un rayon ;
- **sortie de corridor :** l'alerte est déclenchée quand le mobile sort d'un corridor prédéfini par l'utilisateur. Le corridor est construit à partir d'un parcours (ou trajet) et d'une largeur de x mètres ;
- **batterie faible :** le déclenchement de l'alerte se fait quand le niveau de batterie du mobile est inférieur à une certaine valeur ;
- **rapprochement entre mobiles :** l'alerte est déclenchée quand le mobile en question se situe à moins de x mètres d'un autre mobile.

Le déclenchement des alertes étant binaire et ne prenant en aucun cas en considération les imprécisions et les erreurs dont peuvent êtres entachées les données reçues par le Geohub, Deveryware a besoin d'améliorer le système de déclenchement d'alertes (appelé *EventServer* pour, littéralement, serveur d'événements). C'est dans ce sens que nos travaux doivent permettre de rendre l'*EventServer* plus fiable en prenant en charge, *via* les techniques issues de la logique floue notamment, l'imprécision inhérente aux diverses techniques de localisation.

Par ailleurs, la configuration des alertes, complètement transparente pour un utilisateur, nécessite néanmoins une grande connaissance du système pour pouvoir transcrire les besoins en paramètres au niveau des mobiles eux-mêmes. La transcription se fait par des messages bas niveau écrits en langage Forth [12] et envoyés aux mobiles afin de configurer la fréquence de remontées des positions, la fréquence de remontée des autres types d'information (niveau de batterie du mobile, quantité d'essence du véhicule si le mobile est connecté avec celui-ci, panne éventuelle, etc.). Le souhait de l'entreprise est de donner davantage de souplesse aux utilisateurs en leur permettant d'exprimer leurs besoins, même s'ils ne connaissent pas le vocabulaire technique, mais sans, non plus, les obliger à passer par une longue, fastidieuse (et coûteuse pour Deveryware) conversation avec un expert technique de la société.

Aussi, nous nous posons la question suivante : comment améliorer les interfaces existantes pour passer d'un traitement quantitatif à un traitement qualitatif des occurrences d'événements et des contraintes de géolocalisation ? En particulier, pouvons-nous permettre aux experts du domaine et aux utilisateurs d'exprimer leurs besoins, préférences et objectifs métiers dans leurs propres mots, donc en **langage naturel**, puis ensuite de traduire ces éléments, le plus fidèlement possible en configurations et/ou alertes ? Nos travaux doivent donc permettre aux utilisateurs de travailler dans un contexte qualitatif plutôt que l'actuel contexte quantitatif, ce qui devrait avoir comme conséquence de réduire considérablement la complexité de programmation d'alertes et le paramétrage des boîtiers GPS.

12. Forth est un langage bas niveau, fondé essentiellement sur l'utilisation explicite de piles et ressemblant à l'Assembleur.

1.3.2 Cadre de la thèse : un projet ANR

Les travaux présentés dans cette thèse s'insèrent dans le cadre d'un projet financé par l'Agence Nationale de la Recherche (ANR) qui s'intitule SALTY [13] (Self-Adaptive very Large disTributed sYstems). Il vise à produire une plate-forme de gestion auto-adaptative pour des grands systèmes distribués. Ainsi, les systèmes peuvent se reconfigurer et s'adapter automatiquement en cas de pannes ou d'événements imprévus afin de garantir, à la fois la disponibilité du service qu'ils offrent et une optimisation des ressources utilisées. Pour y parvenir, SALTY s'appuie sur des techniques issues de diverses disciplines telles que l'informatique autonomique, l'aide à la décision, l'ingénierie dirigée par les modèles (*Model Driven Engineering*) ou encore l'architecture logicielle.

Le consortium du projet SALTY comporte huit partenaires (quatre académiques et quatre industriels) :

- *Deveryware*
- INRIA Lille (groupe ADAM)
- MAATG France
- Petals Link
- Thales
- **Université Nice Sophia Antipolis : groupe MODALIS (porteur)**
- *Université Paris 8*
- Université Pierre et Marie CURIE (LIP6-MoVe)

La partie que devaient traiter Deveryware et l'Université Paris 8 concernait surtout un des cas d'utilisation du projet : le suivi de camion (*Truck tracking use case*). Plusieurs *scenarii* ont été envisagés au sein de cette étude de cas que partageaient les quatre partenaires Deveryware, Thales, Université Pierre et Marie Curie (UPMC) et Université Paris 8. Le travail de Paris 8 devait servir à définir les différentes alertes sur le GeoHub permettant de remonter les données vers les gestionnaires autonomiques de l'application du cas d'utilisation en question, gestionnaires autonomiques étant traités par l'UPMC et Thales.

Comme pour tout projet ANR, SALTY implique la participation de tous les partenaires à diverses activités tout au long de la durée du projet (39 mois pour SALTY). Ces activités incluent entre autres la production des rapports, dits "livrables" internes et externes, l'implémentation et la publication de logiciels, la participation aux réunions techniques et à celles de suivi du projet. J'ai bien sûr participé à toutes ces activités en tant que membre des deux équipes : Deveryware et Université Paris 8. Sur douze livrables externes rendus au total, quatre livrables externes impliquaient Paris 8 et Deveryware et un livrable externe impliquait Deveryware mais pas Paris 8. De plus, neuf livrables

13. Projet SALTY, n° ANR-09-SEGI-012

internes impliquant Paris 8 ont été également rédigés. Ils sont tous disponibles à l'URL `https://salty.unice.fr`

Le contexte tant scientifique qu'applicatif du travail ayant été posé, il convient maintenant d'étudier les outils et solutions existantes (s'il y en a) pour répondre à nos besoins. Parmi ces outils, les techniques purement numériques ne sont pas pertinentes puisqu'il s'agit de travailler avec le matériau langagier, d'une part, et avec des données dont la caractéristique principale est l'imprécision, d'autre part. De ce fait, on est obligé de recourir à des démarches scientifiques de nature cognitive, et qui relèvent de l'intelligence artificielle et du langage naturel.

Chapitre 2

Etat de l'art

Sommaire

Le travail, dans le cadre de cette thèse, puisqu'il s'agit de manipuler notamment des données sous des contraintes spatiales et temporelles dues au dispositif matériel lui-même, touche ainsi à des problèmes liés aux systèmes d'information géographique et se situe par ailleurs à l'intersection de plusieurs thématiques en informatique :

1. la représentation des connaissances et le raisonnement en milieu incertain, domaines relevant de l'intelligence artificielle, et généralement modélisés par la logique floue et ses domaines dérivés ;
2. les sujets liés à la formulation et à l'énonciation ainsi qu'à la nature même de ces connaissances, et donc relevant du traitement du langage naturel.

Différentes questions émergent, de ce fait : quels sont les domaines ou disciplines qui traitent de l'imprécision ou de l'imperfection ? La logique floue, certes, mais encore ? Et quels sont ceux qui traitent des mots ? Le TALN, certes, mais encore ? Et quel lien existe aujourd'hui entre les deux ?

2.1 Modélisation des imperfections

Comme nous l'avons présenté dans l'introduction, les humains ont la faculté de raisonner avec des données qui sont souvent vagues, incertaines et imprécises. Cette imperfection se retrouve naturellement dans le langage naturel et représente la partie subjective dans les jugements et raisonnements de chacun. Ainsi, les mêmes termes peuvent avoir différentes sémantiques (ou en tous cas différentes nuances d'une même sémantique) selon les personnes et les contextes dans lesquels ils sont employés. C'est exactement cela qu'il nous faut traiter, vu les questions que nous avons soulevées dans le chapitre 1.

On distingue en général trois catégories d'imperfection des connaissances :
- les incertitudes qui portent sur le bien-fondé, l'exactitude de l'information. Par exemple, la phrase "Il *pense que* cette rue est en zone inondable" est incertaine ;
- les imprécisions qui concernent le périmètre sur lequel porte l'information. Par exemple, les phrases "La zone inondable s'étend sur *à peu près* dix kilomètres carrés" ou "La zone inondable s'étend sur une surface *assez vaste*" sont imprécises ;
- et les incomplétudes qui portent sur l'absence totale ou partielle d'information.

La catégorie qui nous concerne ici est principalement celle des imprécisions, et, à la marge, celle des incomplétudes, lorsque les données de position des mobiles sont manquantes.

Dans la littérature, il existe plusieurs approches proposant des formalismes permettant de gérer ce type de données imprécises et approximatives. Ces approches découlent des premiers travaux sur les sous-ensembles flous et également sur le raisonnement approximatif, menés par Lotfi Zadeh dès 1965 [Zadeh, 1965]. Dans ce qui suit, nous passons en revue les formalismes les plus pertinents par rapport à nos travaux.

2.1.1 Logique floue

La théorie des sous-ensembles flous introduite par Lotfi Zadeh peut être considérée comme une généralisation de la théorie des ensembles dits *classiques* (par opposition aux ensembles flous). Un élément peut y appartenir *plus ou moins* à un ensemble donné, modélisant ainsi une incertitude qui permet par la suite un raisonnement plus flexible. Les données y sont modélisées par des variables linguistiques qui sont l'association d'un terme linguistique à un sous-ensemble flou [Zadeh, 1975]. Ainsi, le modèle de calcul numérique "classique" devient un "calcul à l'aide de mots" (*Computing with Words* (CW))), concept que Zadeh liera un peu plus tard à l'essence-même de la logique floue [Zadeh, 1996].

Si l'on reprend l'exemple de la phrase citée plus haut "La zone inondable s'étend sur une surface *assez vaste*", c'est l'information sur la surface de la zone inondable qui est approximative, puisque décrite comme "assez vaste". Dans ce cas, on se donne un ensemble de termes linguistiques qui caractérise la surface, par exemple, l'ensemble : {très-petite, petite, assez-petite, moyenne, assez-vaste, vaste, très-vaste}. Selon la théorie des sous-ensembles flous, la sémantique des termes linguistiques est donnée par la fonction d'appartenance qui leur est associée. Ainsi, ces sept termes sont associés à sept sous-ensembles flous dont la fonction d'appartenance peut prendre différentes formes (singleton, triangulaire, trapézoïdale, gaussienne...).

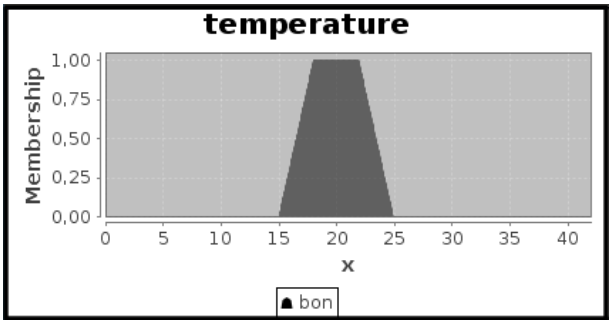


FIGURE 2.1 – Sous-ensemble flou pour une température douce (en ˚C) correspondant à l'expression "Il fait bon".

Pour simplifier les calculs, il est usuel de choisir une fonction de forme trapézoïdale pour représenter l'appartenance à un sous-ensemble. La figure 2.1 montre une fonction d'appartenance de ce type pour le sous-ensemble flou *"il fait bon"* qui exprime une température comprise entre 15 et 25˚C, correspondant exactement au terme "bon" lorsque la valeur est comprise entre 18 et 23˚C et *plus ou moins* à ce même terme lorsque la valeur est entre 15 et 18˚C ou entre 23 et 25˚C.

Pour mémoire, nous rappelons maintenant quelques définitions et principes fondamentaux de la théorie des sous-ensembles flous qui vont permettre de bien dresser ce cadre théorique.

Définition 1. *Un sous-ensemble flou A de l'ensemble X (l'univers de discours) est caractérisé par une fonction d'appartenance $f_A(x)$ telle que : $\forall x \in X, f_A(x) \in [0,1]$*

Dans l'exemple précédent, A peut être "il fait bon" ou bien "température douce".

L'univers de discours X peut être continu ou discret, mais dans le cadre de cette thèse, il est en tous cas toujours ordonné.

Dans l'exemple précédent, X est l'ensemble des valeurs que peut prendre la température, ici de 0 à 42˚C. C'est donc un univers continu.

Si l'on ne devait manipuler que des mots, l'univers de discours serait un ensemble (discret, donc) de termes linguistiques ou de symboles qui décrirait chaque valeur pouvant être prise par la variable.

Définition 2. *Une variable linguistique s'écrit sous la forme d'un triplet (V, X, T_V) où V est le nom de la variable, X est l'univers de discours et $T_V = \{A_1, A_2, \ldots\}$ est l'ensemble des sous-ensembles flous permettant de caractériser V.*

Par exemple, nous pouvons décrire une variable de température ambiante par : (température, [0,42], {basse, moyenne, douce, chaude}).

La valeur d'appartenance (ou le degré de vérité) d'un certain x à un ensemble donné est donc la valeur $f_A(x)$ prise par la fonction f_A. Par exemple, si nous prenons le sous-ensemble flou présenté dans la figure 2.1, la valeur 20 appartient complètement ($f_A(20) = 1.0$) au sous-ensemble flou *"il fait bon"* ou *"température douce"*.

Définition 3. *Soit A un sous-ensemble flou appartenant à X :*
- *Le support de A est $Supp(A) = \{x \in X \text{ tel que } f_A(x) \neq 0\}$*
- *Le noyau de A est $Ker(A) = \{x \in X \text{ tel que } f_A(x) = 1\}$*

Les éléments de $Supp(A)$ sont des éléments appartenant "au moins un peu" à A alors que ceux de $Ker(A)$ sont ceux qui appartiennent "pleinement" à A. Dans l'exemple de la figure 2.1, le noyau du sous-ensemble flou *"il fait bon"* correspond aux éléments appartenant à l'intervalle $[18, 23]$, tandis que son support correspond aux éléments appartenant à l'intervalle $]15, 25[$.

Nous donnons également la définition de quelques opérations sur les sous-ensembles flous en prenant deux ensembles A et B appartenant à X.

Pour comparer deux sous-ensembles flous, il faut comparer leurs fonctions d'appartenance respectives.

Définition 4. *Égalité : $A = B$ ssi $\forall x \in X, f_A(x) = f_B(x)$*

Pour connaître les points communs entre deux sous-ensembles flous, il faut également regarder leurs fonctions d'appartenance respectives et estimer à quel point elles se ressemblent. On prend souvent l'opérateur min pour cela.

Définition 5. *Intersection : $A \cap B \Rightarrow f_{A \cap B}(x) = min(f_A(x), f_B(x))$*

Pour réunir deux concepts en un seul (réunir deux sous-ensembles flous), on prend souvent l'opérateur max.

Définition 6. *Union : $A \cup B \Rightarrow f_{A \cup B}(x) = max(f_A(x), f_B(x))$*

Pour exprimer une notion contraire (le "non"), on utilise le complément de la fonction d'appartenance du sous-ensemble flou correspondant à la notion.

Définition 7. *Le complémentaire A^C est défini par $f_{A^C}(x) = 1 - f_A(x)$ avec $A^C \cup A \neq X$ et $A^C \cap A \neq \emptyset$*

Il est à noter que le max et le min ne sont pas les seuls opérateurs possibles pour définir respectivement l'union et l'intersection de deux sous-ensembles flous. De manière générale, nous pouvons définir l'intersection par une norme triangulaire (t-norme) et l'union par une conorme triangulaire (t-conorme). Une t-(co)norme prend en entrée deux valeurs d'appartenance, donc deux valeurs comprises entre 0 et 1, notées chacune x et y, et renvoie en sortie une seule valeur d'appartenance. La t-(co)norme est donc à ce titre une fonction d'agrégation. Pour définir un nouveau sous-ensemble flou à partir d'une t-(co)norme entre deux sous-ensembles flous, il convient donc d'appliquer les formules ci-dessous, de manière itérative, pour toutes les valeurs d'appartenance que peuvent prendre les deux sous-ensembles flous en question.

Définition 8. *Une t-norme est une fonction $\top : [0,1] \times [0,1] \to [0,1]$ qui satisfait les propriétés suivantes :*
- *commutativité : $\top(x, y) = \top(y, x)$*
- *associativité : $\top(x, \top(y, z)) = \top(\top(x, y), z)$*

- *monotonie :* $\top(x, y) \leq \top(z, t)$ *si* $x \leq z$ *et* $y \leq t$
- *1 est élément neutre :* $\top(x, 1) = x$

Il est ainsi possible de définir de manière générique l'intersection par :

$$A \cap^{\top} B \Rightarrow f_{A \cap^{\top} B}(x) = \top(f_A(x), f_B(x))$$

Définition 9. *Une t-conorme est une fonction* $\perp : [0, 1] \times [0, 1] \rightarrow [0, 1]$ *qui satisfait les propriétés suivantes :*

- *commutativité :* $\perp(x, y) = \perp(y, x)$
- *associativité :* $\perp(x, \perp(y, z)) = \perp(\perp(x, y), z)$
- *monotonie :* $\perp(x, y) \leq \perp(z, t)$ *si* $x \leq z$ *et* $y \leq t$
- *0 est élément neutre :* $\perp(x, 0) = x$

L'union peut être définie de manière générique par :

$$A \cup^{\perp} B \Rightarrow f_{A \cup^{\perp} B}(x) = \perp(f_A(x), f_B(x))$$

Le choix d'une t-norme et celui d'une t-conorme est lié par un lien de complémentation. En effet, dans un calcul, si une t-norme et une t-conorme doivent être appliquées, il convient de choisir la t-conorme en fonction de la t-norme ou bien la t-conorme en fonction de la t-norme. Elles sont ainsi dites duales :

Définition 10. *Une t-norme et une t-conorme sont duales si et seulement si :*

- $1 - \top(x, y) = \perp(1 - x, 1 - y)$
- $1 - \perp(x, y) = \top(1 - x, 1 - y)$

La dualité entre t-normes et t-conormes permet de conserver les lois de De Morgan. Autrement dit, le complément de l'intersection de deux sous-ensembles flous doit être égal à l'union des deux compléments des sous-ensembles flous. Et vice-versa, c'est-à-dire que le complément de l'union de deux sous-ensembles flous doit être égal à l'intersection des deux compléments des sous-ensembles flous. Le tableau 2.1 présente une liste de t-normes et t-conormes associées, où p est choisi tel que $p > 1$.

t-norme	t-conorme	nom
$min(x, y)$	$max(x, y)$	Zadeh
$max(0, x + y - 1)$	$min(1, x + y)$	Łukasiewicz
$1 - min([(1 - x)^p + (1 - y)^p]^{\frac{1}{p}}, 1)$	$min((x^p + y^p)^{\frac{1}{p}}, 1)$	Yager
xy	$x + y - xy$	probabiliste

TABLE 2.1 – Quelques t-normes et t-conormes associées.

Le raisonnement en logique floue est fondé sur un élément principal : la **relation floue** [Zadeh, 1965, Mamdani et Assilian, 1975]. On définit une relation floue comme suit :

Définition 11. *Soient* X *et* Y *deux ensembles. Une relation floue* $\mathcal{R}$ *entre* X *et* Y *est un sous-ensemble flou appartenant à* $X \times Y$

Ainsi, les implications floues ont été construites comme des relations floues entre deux ensembles X et Y. Par exemple, la température de l'air en ˚C (ensemble X) est liée aux niveaux du plan canicule (ensemble Y) : il y a une implication (que l'on pourrait qualifier de floue) entre la température et les trois niveaux de déclenchement : le niveau de veille saisonnière, le niveau de mise en garde et actions et le niveau de mobilisation maximale.

Définition 12. *Soient A et B deux sous-ensembles flous appartenant aux deux ensembles X et Y. Une implication floue se traduit par :*

$$SI\ (x\ est\ A)\ ALORS\ (y\ est\ B)$$

où x et y sont des valeurs appartenant aux deux ensembles de référence X et Y respectivement.

Une implication floue est donc une quantification du degré de vérité (ou du lien) qui lie les deux propositions élémentaires (x est A) et (y est B). Il est donc naturel qu'elle dépende des fonctions d'appartenance, f_A et f_B, des deux sous-ensembles flous A et B.

L'implication notée $(p \Rightarrow q) = (\neg p \vee q)$ en logique classique où les valeurs prennent seulement les valeurs 0 ou 1, est définie par la table de vérité *unique* du tableau 2.2.

$f_A(x)$	$f_B(y)$	$f_{\mathcal{R}}(x,y)$
0	0	1
0	1	1
1	0	0
1	1	1

TABLE 2.2 – Implication en logique classique

Par contre, en logique floue, il existe *plusieurs* définitions de l'implication dont nous présentons quelques unes des plus utilisées dans le tableau 2.3.

Nom	Définition
Mamdani	$\min(f_A(x), f_B(y))$
Łukasiewicz	$\min(1 - f_A(x) + f_B(y), 1)$
Larsen	$f_A(x)f_B(y)$
Reichenbach	$1 - f_A(x) + f_A(x)f_B(y)$
Kleene-Dienes	$\max(1 - f_A(x), f_B(y))$

TABLE 2.3 – Les principales implications floues.

La définition de Mamdani est la plus simplificatrice, puisqu'elle réduit une implication à un opérateur de type "et" [Mamdani, 1977]. Celle de Łukasiewicz est conforme à la t-norme du même nom [Łukasiewicz, 1920] et celle de Larsen correspond à la t-norme

probabiliste [Larsen, 1980]. L'implication de Reichenbach résulte de l'axiome $(p \Rightarrow q) = (\neg p \vee q)$ qui peut s'écrire également $(p \Rightarrow q) = (1 \wedge (\neg p \vee q))$ donc $(p \Rightarrow q) = (\neg p \vee p) \wedge (\neg p \vee q)$ c'est-à-dire, en factorisant, $(p \Rightarrow q) = \neg p \vee (p \wedge q))$, avec l'addition pour disjonction et le produit pour conjonction [Reichenbach, 1934].

La définition de Kleene-Dienes, quant à elle, correspond exactement à la définition de la logique classique, mais appliquée aux sous-ensembles flous [Dienes, 1949].

La conception d'un système à raisonnement flou passe le plus souvent par la création de règles d'inférence. Chaque règle est une implication floue qui peut être composée de plusieurs propositions floues liées par des opérateurs binaires (ET et OU). Si nous considérons trois sous-ensembles flous A, B et C appartenant à X, Y et Z respectivement, une règle d'inférence floue peut par exemple s'écrire sous la forme :

$$\text{SI } (x \text{ est } A) \textbf{ ET } (y \text{ est } B) \text{ ALORS } (z \text{ est } C)$$

avec $x \in X, y \in Y$ et $z \in Z$.

Le résultat de l'évaluation d'une règle floue est un sous-ensemble flou dont la règle générale s'écrit sous la forme indiquée dans la définition 13.

Définition 13. *Soient A, B et C trois sous-ensembles flous appartenant à X, Y et Z respectivement et f_A, f_B et f_C leurs fonctions d'appartenance respectives. Soit également $f_{\mathcal{R}}$ la fonction d'appartenance d'une implication floue. La fonction d'appartenance du sous-ensemble flou solution S de Z est :*

$$f_S(z) = f_{\mathcal{R}}(\top(f_A(x), f_B(y)), f_C(z))$$

où $\top$ est une t-norme et $x \in X, y \in Y, z \in Z$.

Le *modus ponens généralisé* (MPG), qui est une extension du *modus ponens* de la logique classique, permet le raisonnement à partir des entrées observées et des règles d'inférence établies [Zadeh, 1975]. Le MPG permet un raisonnement approximatif, c'est-à-dire qu'il construit à partir de la règle "SI (x est A) ET (y est B) ALORS (z est C)" et des observations "x est A'" et "y est B'" une conclusion "z est C'", où A', B' et C' sont des formes modifiées (approximatives) de A, B et C.
Lors du raisonnement flou, il résulte de l'évaluation de chaque règle d'inférence numérotée i un sous-ensemble flou solution S_i. Chaque règle ayant donné "sa" solution, il faut ensuite agréger l'ensemble de ces solutions pour obtenir le sous-ensemble flou solution final. L'agrégation est une disjonction, c'est-à-dire que, graphiquement, le sous-ensemble flou solution final est représenté par l'union de toutes les fonctions d'appartenance obtenues à chaque règle (*cf.* Figure 2.2).

Définition 14. *La fonction d'appartenance du sous-ensemble flou S des solutions du système d'inférence flou est :*

$$\forall z \in Z, f_S(z) = \bigvee(f_{S_1}(z), f_{S_2}(z), \ldots, f_{S_n}(z))$$

où $\bigvee$ est un opérateur d'agrégation floue.

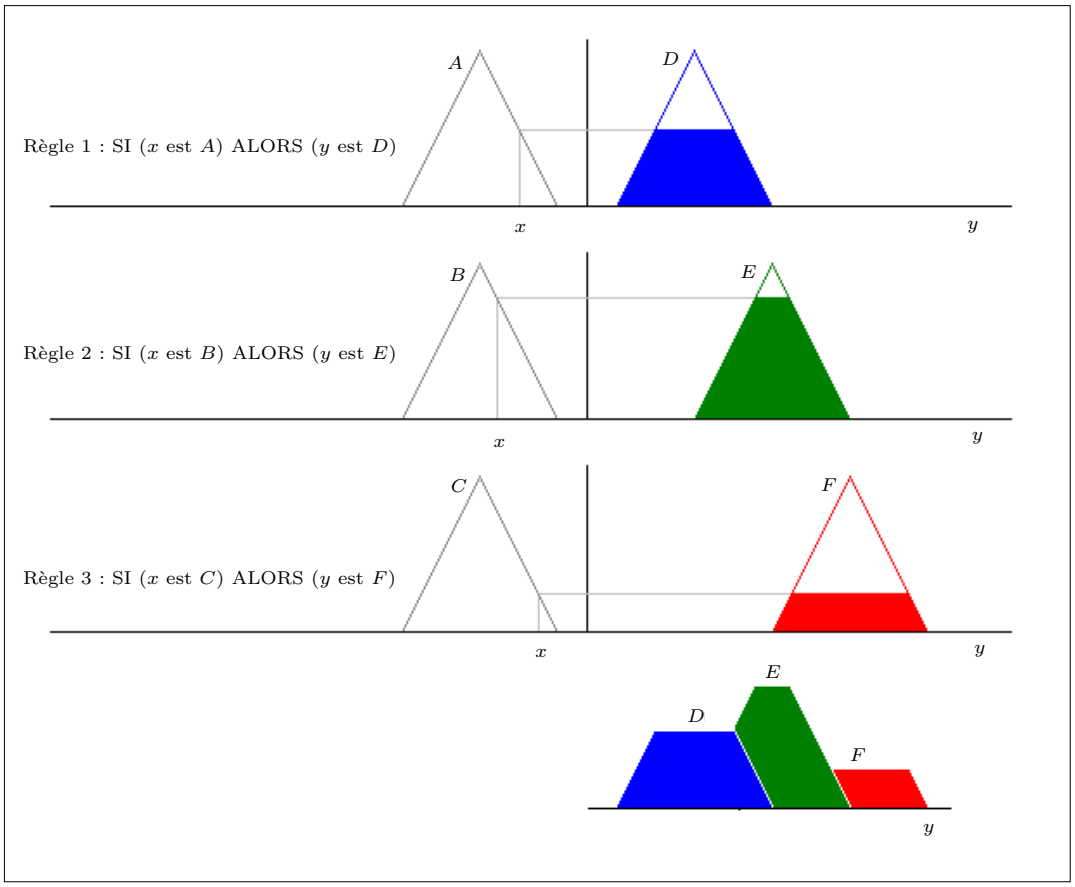


FIGURE 2.2 – Le sous-ensemble flou solution final est composé de l'union des 3 sous-ensembles flous obtenus après l'application de chaque règle *(source : Wikipedia)*.

Dans un système à contrôleur flou, les entrées sont souvent des valeurs numériques (provenant de capteurs par exemple) [Mamdani et Assilian, 1975]. Il est donc nécessaire de passer par une étape de *fuzzification* des entrées. La *fuzzification* consiste en l'attribution d'une fonction d'appartenance à une valeur réelle donnée. Elle dépend donc grandement de la modélisation floue des variables linguistiques d'entrée du système.

Aussi est-il nécessaire de *défuzzifier* le sous-ensemble flou final pour obtenir une solution numérique exploitable par le système. Cette opération est donc l'opération inverse de la *fuzzification*.

Il existe plusieurs méthodes [Van Leekwijck et Kerre, 1999] de *défuzzification* parmi lesquelles on peut citer :

– Méthode du **maximum** : la valeur *défuzzifiée* est l'abscisse du point maximal de la fonction d'appartenance du sous-ensemble flou solution. Cette méthode a l'avantage d'être facile à mettre en œuvre et ne nécessite pas beaucoup de calculs.

Cependant, la valeur maximale n'est pas toujours la mieux représentative du sous-ensemble flou *défuzzifié*.
- Méthode de la **moyenne des maxima** : c'est une variante de la méthode du maximum mais qui consiste à prendre l'abscisse de la moyenne des maxima. Elle est utile quand plusieurs points ont la valeur maximale du sous-ensemble flou solution.
- Méthode du **centre de gravité** : la valeur choisie ici est l'abscisse du centre de gravité du sous-ensemble flou solution. Cette méthode donne, dans la plupart des cas, de meilleurs résultats que les précédentes car elle prend en considération l'ensemble des variations du sous-ensemble flou solution, mais elle est beaucoup plus gourmande en temps de calcul.

Le choix de la méthode de *défuzzification* dépend donc essentiellement de la précision souhaitée et des ressources mises à disposition du système à contrôleur flou.

2.1.2 Modélisation à l'aide des 2-tuples linguistiques

Comme nous venons de le voir, l'approche de la logique floue permet un raisonnement approximatif dans un contexte qualitatif en se fondant notamment sur la représentation des données sous forme de variables linguistiques (voir la définition 2). Ceci s'avère très utile dans des cas où les données sont non quantifiables, vagues ou entachées d'imprécision et dont les descriptions linguistiques sont plus facilement compréhensibles par les humains.

Cependant, le raisonnement en logique floue implique souvent une perte de précision. En effet, nous avons vu que le sous-ensemble flou final obtenu suite à l'évaluation des règles d'inférence nécessite une *défuzzification* afin d'être exploité par un système à contrôleur flou par exemple. Cette *défuzzification* engendre une perte d'information puisque le résultat obtenu n'est qu'une approximation du sous-ensemble flou solution (une valeur représentative de l'information représentée par le sous-ensemble flou).

Aussi, le sous-ensemble flou solution n'est pas facilement exprimable par le biais des sous-ensembles flous des variables linguistiques du système. Idéalement, le résultat de l'inférence floue devrait être exprimé sous la forme (éventuellement modifiée) d'un des sous-ensembles flous de sortie ayant servi à modéliser le système flou. Dans l'exemple de la figure 2.2, cela signifierait que le sous-ensemble flou "union", résultant des sous-ensembles flous D, E et F devrait être exprimé relativement aux sous-ensembles flous de départ, soient A, B ou C. Ceci permettrait une compréhension plus facile du résultat obtenu. Or, ici ce n'est pas le cas car le sous-ensemble flou solution ne ressemble à aucun des sous-ensembles flous du système.

Pour pallier ces lacunes inhérentes au modèle de calcul en logique floue, un modèle de représentation de linguistique floue fondé sur des 2-tuples appelés *fuzzy linguistic 2-tuples* (FL2T) a été introduit par Herrera et Martínez à partir des années 2000 [Herrera et Martínez, 2000]. Nous choisissons de garder le terme anglosaxon originel *2-tuple* (au lieu de "couple" ou "paire") pour bien marquer le fait que c'est de ce modèle-ci dont il s'agit et pas d'un autre. Par ailleurs, une communauté

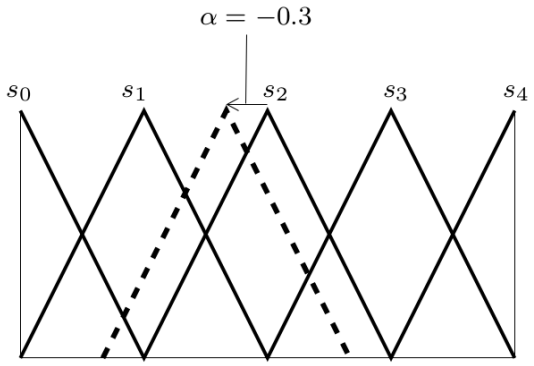


FIGURE 2.3 – Représentation du 2-tuple $(s_2, -0.3)$.

maintenant importante a adopté ce modèle, en conservant également le nom originel, voir par exemple [Wang et Hao, 2006, Dong et al., 2013, Orduna et al., 2013, Pérez-Asurmendi et Chiclana, 2013, Dursun et Karsak, 2014].

Dans leur modèle, les auteurs représentent les données floues sous la forme d'un couple (s, α) où s appartient à un ensemble de $g+1$ termes linguistiques $S = \{s_0, \ldots, s_g\}$ et α est une valeur numérique représentant la *translation symbolique* associée au terme s. Les termes linguistiques sont censés exprimer les différentes nuances de l'information dont il est question, nuances symbolisées par des mots, d'où, sans doute, l'expression "linguistic" des auteurs (par exemple, $T = \{s_0 : \textit{complètement faux}, s_1 : \textit{plutôt faux}, s_2 : \textit{ni vrai ni faux}, s_3 : \textit{plutôt vrai}, s_4 : \textit{complètement vrai}\}$) et on suppose que le choix de cet ensemble S est fait par un expert. La translation symbolique n'ayant pas de sémantique particulière pour l'instant, à part une simple notion de décalage sur l'axe, les auteurs la fixe à zéro pour tous les couples de départ. C'est après calculs que sa valeur pourra éventuellement changer. Ainsi, si α est positif, s_i est renforcé, sinon, s_i est affaibli.

Les définitions 15 à 17 sont empruntées à Herrera et Martínez, *op. cit.*

Définition 15. *Une* translation symbolique *est une valeur numérique comprise dans l'intervalle* $[-0.5, 0.5[$. *Elle représente la différence entre* β*, le résultat d'une agrégation d'information linguistique, et l'indice du terme linguistique le plus proche de ce résultat.*

Le sous-ensemble flou triangulaire en pointillé de la figure 2.3 représente la modélisation du 2-tuple $(s_i, -0.3)$. Cette modélisation est obtenue en translatant le terme s_2 à gauche à l'aide d'une translation symbolique $\alpha = -0.3$.

Ainsi, pour un ensemble de termes linguistiques $S = \{s_0, \ldots, s_g\}$ et pour une valeur $\beta \in [0, g]$ mais avec $\beta \notin \{0, \ldots, g\}$ (c'est-à-dire β n'étant pas une valeur entière) obtenue par une agrégation, il est nécessaire d'utiliser une fonction d'approximation pour exprimer le résultat dans S. Le 2-tuple qui représente β est obtenu par la fonction Δ qui renvoie un couple de $S \times [-0.5, 0.5[$.

Définition 16. *La fonction* Δ *permettant d'obtenir un 2-tuple est définie par :*

$$\Delta : [0, g] \to S \times [-0.5, 0.5[$$
$$\Delta(\beta) = \begin{cases} s_i & i = round(\beta) \\ \alpha = \beta - i & \alpha \in [-0.5, 0.5[\end{cases}$$

où *round* est une opération d'arrondi permettant de récupérer la partie entière, s_i est le terme linguistique ayant l'indice le plus proche de β et α est la valeur de la translation symbolique.

Par exemple, si, après un calcul, on obtient $\beta = 1.6$ alors le 2-tuple associé sera $\Delta(\beta) = (s_i, \alpha) = (s_2, \beta - i) = (s_2, -.4)$.

Naturellement, tout terme linguistique s_i appartenant à un ensemble S peut être converti en 2-tuple en utilisant la fonction θ :

$$\theta : S \to S \times [-0.5, 0.5[$$
$$\theta(s_i) = (s_i, 0)/s_i \in S$$

La fonction Δ^{-1} permettant de calculer la valeur numérique β correspondant à un 2-tuple est l'inverse de la fonction Δ.

Définition 17. *Soient $S = \{s_0, \ldots, s_g\}$ un ensemble de termes linguistiques et (s_i, α) un 2-tuple, alors :*

$$\Delta^{-1} : S \times [-0.5, 0.5[\to [0, g]$$
$$\Delta^{-1}(s_i, \alpha) = i + \alpha = \beta$$

En reprenant le même exemple que précédemment, on retrouve bien la valeur β : $\Delta^{-1}(s_2, -.4) = 2 - .4 = 1.6$.

Herrera et Martínez, *op. cit.*, définissent un modèle de calcul fondé sur les 2-tuples permettant de raisonner avec des données floues sans perte d'information. Ils définissent en particulier trois fonctions :

1. la comparaison de 2-tuples ;
2. la négation d'un 2-tuple ;
3. l'agrégation de 2-tuples.

La comparaison

La comparaison entre deux 2-tuples est définie de la manière suivante :

Définition 18. *Soit deux 2-tuples (s_i, α_i) et (s_j, α_j) :*

- *si $i < j$ alors (s_i, α_i) est inférieur à (s_j, α_j)*
- *si $i > j$ alors (s_i, α_i) est supérieur à (s_j, α_j)*

- *si* $i = j$
  - *si* $\alpha_i < \alpha_j$ *alors* (s_i, α_i) *est inférieur à* (s_j, α_j)
  - *si* $\alpha_i > \alpha_j$ *alors* (s_i, α_i) *est supérieur à* (s_j, α_j)
  - *si* $\alpha_i = \alpha_j$ *alors* (s_i, α_i) *et* (s_j, α_j) *sont égaux*

La négation

Les 2-tuples étant définis sur un ensemble continu, la négation d'un 2-tuple est définie comme suit :

Définition 19. *Soient* $S = \{s_0, s_1 \ldots, s_g\}$ *et* (s_i, α_i) *un 2-tuple. La négation de* (s_i, α_i) *est calculée ainsi :*

$$Neg((s_i, \alpha_i)) = \Delta(g - (\Delta^{-1}(s_i, \alpha_i)))$$

L'agrégation

En agrégeant des 2-tuples, on cherche à les combiner pour mieux exprimer le résultat d'un calcul. Par exemple, dans un sondage d'opinion, on peut souhaiter agréger les différentes réponses pour exprimer un consensus.

Parmi les différents agrégateurs existant dans la littérature, nous pouvons tout d'abord citer la moyenne arithmétique, la moyenne pondérée et la moyenne pondérée ordonnée.

La moyenne arithmétique permet de calculer le point (ou valeur) central d'un ensemble de données floues.

Définition 20. *Soit un ensemble de 2-tuples* $A = \{(s_0, \alpha_0), (s_1, \alpha_1), \ldots, (s_g, \alpha_g)\}$*. La moyenne arithmétique de l'ensemble A est calculée par M :*

$$M(A) = \Delta(\sum_{i=1}^{g} \frac{1}{g} \Delta^{-1}(s_i, \alpha_i))$$

Dans la moyenne pondérée, des poids w_i sont associés à chacun des 2-tuples de l'ensemble afin de leur donner une importance différente dans le calcul.

Définition 21. *Soient* $A = \{(s_0, \alpha_0), (s_1, \alpha_1), \ldots, (s_g, \alpha_g)\}$ *un ensemble de 2-tuples et* $W = \{w_1, w_2, \ldots, w_g\}$ *l'ensemble des poids qui lui est associé. La moyenne pondérée est calculée ainsi :*

$$MP(A) = \Delta(\frac{\sum_{i=1}^{g} \Delta^{-1}(s_i, \alpha_i).w_i}{\sum_{i=1}^{g} w_i})$$

Une généralisation de la moyenne pondérée appelée *Ordered Weighted Averaging operator* (OWA) a été introduite par Yager [Yager, 1988, Yager, 1993]. Celle-ci permet d'associer des poids, cette fois-ci non pas aux valeurs mais aux positions occupées par ces valeurs. Ainsi, puisque l'ordre des poids ne change pas, tout changement dans l'ordre

des valeurs à agréger peut changer la valeur de la moyenne résultant de l'agrégation. Par exemple, si l'on range les valeurs dans l'ordre croissant et que l'on affecte à ces valeurs les poids suivants : $1, 0, ..., 0$ alors l'OWA devient un opérateur min. Si, au contraire, on affecte les poids suivants : $0, ..., 0, 1$ alors l'OWA devient un opérateur max.

La définition 22 donne la formule de l'opérateur OWA de type moyenne, adapté aux 2-tuples.

Définition 22. *Soient* $A = \{(s_0, \alpha_0), (s_1, \alpha_1), \ldots, (s_g, \alpha_g)\}$ *un ensemble de 2-tuples et* $W = \{w_1, w_2, \ldots, w_g\}$ *l'ensemble de poids qui lui est associé avec* $w_i \in [0, 1]$ *et* $\sum w_i = 1$. *L'opérateur* F *de la moyenne OWA est calculé ainsi :*

$$F(A) = \Delta(\textstyle\sum_{i=1}^{g} w_i.\beta_i)$$

où β_i *est la* $i^{\text{ème}}$ *plus grande valeur parmi les valeurs* $\Delta^{-1}(s_i, \alpha_i)$.

Ainsi, partant de la nature du modèle de représentation des 2-tuples, plusieurs types d'agrégateurs ont été adaptés spécialement par l'équipe de Herrera et Martínez. De ce fait, s'inspirant notamment des travaux de Yager sur l'agrégation de données [Yager, 1998, Yager et Troiano, 2005], est né l'opérateur *Linguistic Ordered Weighted Averaging operator* (LOWA), le *Weighted Median Aggregation* (WMA) et bien d'autres (par exemple S-OWA (S étant une t-conorme), T-OWA (T étant une t-norme), ST-OWA). En effet, l'agrégation passe donc d'abord par une phase de transformation des 2-tuples à agréger en valeurs numériques *via* la fonction Δ (définition 16).

Ces valeurs numériques sont ensuite agrégées en utilisant tout opérateur d'agrégation numérique. Enfin, le résultat numérique obtenu suite à l'agrégation est transformé en 2-tuple en utilisant la fonction Δ^{-1}. Un exemple est donné plus loin, avec des notes anglosaxones.

Ensembles de termes linguistiques non équilibrés

La modélisation de la linguistique floue fondée sur les 2-tuples permet à un grand nombre de problèmes d'être estimés par des variables linguistiques dont les termes linguistiques sont symétriquement et uniformément distribués sur l'axe de valeurs autour d'une valeur centrale. Ainsi, quand l'information est parfaitement répartie (*i.e.* si la distance séparant les termes est toujours exactement la même), tous les s_i sont uniformément distribués sur l'axe. Cependant, il existe plusieurs cas où nous avons besoin de modéliser les données de manière non uniforme en constituant des ensembles de termes linguistiques non équilibrés (*unbalanced linguistic term sets*). Le besoin de ce genre d'ensemble se fait notamment sentir lorsqu'il nous faut plus de finesse et de précision d'un côté plutôt que de l'autre de l'univers de discours. Par exemple, pour une variable linguistique décrivant la chaleur, nous pouvons avoir besoin de plusieurs termes pour représenter le côté "chaleur" et seulement d'un ou deux termes pour le coté "froid" car ce côté ne nous intéresse pas vraiment pour le problème à traiter. Un autre exemple est proposé dans la figure 2.4, concernant la notation anglosaxone (A pour "excellent", B pour "bien", C pour "assez bien", D pour "moyen" et E pour "insuffisant") non uniformément distribuée sur l'axe des notes.

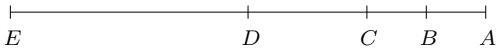


FIGURE 2.4 – Notes non uniformément distribuées sur l'axe.

Afin de gérer ce genre de cas de figure, Herrera et Martínez ont présenté en 2008 une approche permettant de traiter des ensembles de termes linguistiques non équilibrés en utilisant la représentation sous forme de 2-tuples [Herrera et al., 2008]. Cette méthode est fondée sur la notion de **hiérarchie linguistique** [Cordón et al., 2001] pour construire une modélisation de l'ensemble de termes linguistiques de départ.

Une hiérarchie linguistique est l'union de plusieurs ensembles de termes linguistiques appelés *niveaux de la hiérarchie*, chacun ayant une granularité différente des autres niveaux (c'est-à-dire que le nombre de termes est différent). Une hiérarchie linguistique n'a donc pas de sémantique intrinsèque, un niveau n'est pas meilleur qu'un autre, simplement l'échelle des termes est subdivisée de façon équidistante à chaque niveau, de sorte de pouvoir exprimer chaque terme dans "son" niveau qui lui est propre.

Les niveaux de la hiérarchie sont notés $l(t, n(t))$ où t est le numéro du niveau dans la hiérarchie linguistique et $n(t)$ son nombre de termes linguistiques. De plus, les termes linguistiques de chaque niveau ont des fonctions d'appartenance triangulaires, sont uniformément et symétriquement distribués sur l'axe, ont une valeur centrale symbolisant une valeur neutre et sont de nombre impair, *cf.* Herrera *et al.*, *op. cit.* Les niveaux d'une même hiérarchie sont ordonnés selon leur granularité, c'est-à-dire que pour deux niveaux consécutifs t et $t+1$, on a $n(t+1) > n(t)$.

Les définitions 23 à 25 sont empruntées à Herrera *et al.*, *op. cit.*

Définition 23. *Soit LH une hiérarchie linguistique. Elle est définie par :*

$$LH = \cup_t l(t, n(t))$$

Ainsi, chaque niveau $l(t, n(t)$ constitue un raffinement du niveau précédent et est représenté par son ensemble de termes linguistiques $S^{n(t)} = \{s_0^{n(t)}, \ldots, s_{n(t)-1}^{n(t)}\}$. Un niveau $t+1$ est obtenu à partir de son prédécesseur en ajoutant un nouveau terme entre chaque couple de termes successifs du niveau t. La définition 24 résume ce processus de raffinement.

Définition 24. *Soit LH une hiérarchie linguistique et t un des niveaux de cette dernière. Le niveau suivant de la hiérarchie est obtenu par récurrence ainsi :*

$$l(t, n(t)) \mapsto l(t+1, 2.n(t) - 1)$$

La figure 2.5 représente une hiérarchie linguistique à 4 niveaux de respectivement 3, 5, 9 et 17 termes.

Naturellement, les termes d'un niveau de la hiérarchie linguistique peuvent être exprimés dans un autre niveau et ce, sans perte d'information (en utilisant la translation symbolique). Ainsi, pour une hiérarchie $LH = \cup_t l(t, n(t))$ dont les termes linguistiques sont notés $S^{n(t)} = \{s_0^{n(t)}, \ldots, s_{n(t)-1}^{n(t)}\}$, la fonction $TF_{t'}^t$ (cf. définition 25) permet d'exprimer un terme de niveau t en son équivalent au niveau t'.

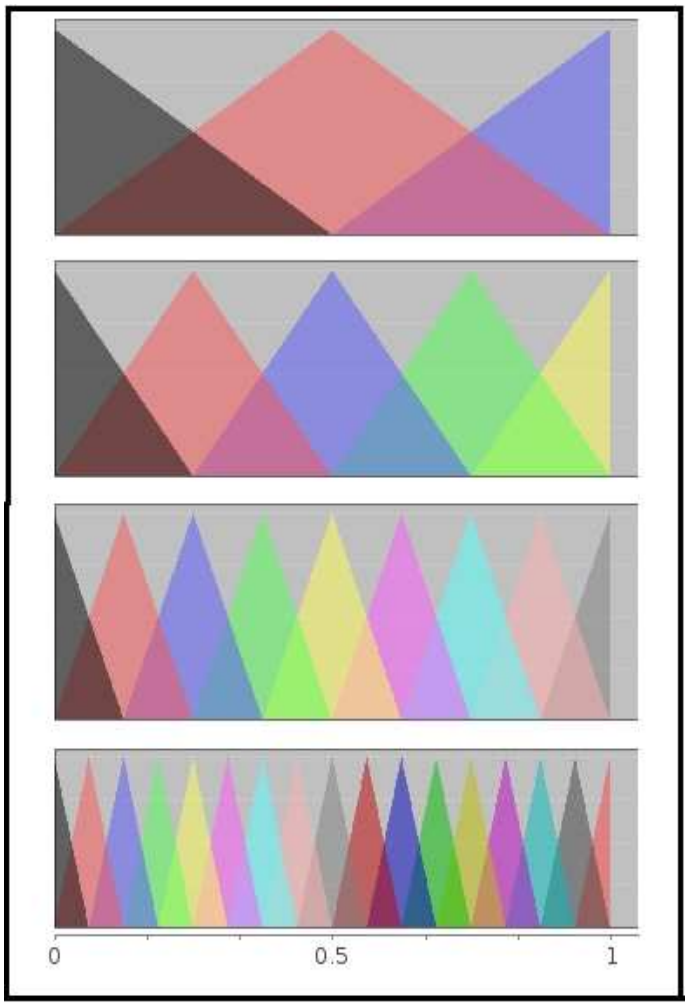


FIGURE 2.5 – Exemple d'une hiérarchie linguistique à 4 niveaux.

Définition 25. *Soit* $LH = \cup_t l(t, n(t))$ *et* $S^{n(t)} = \{s_0^{n(t)}, \ldots, s_{n(t)-1}^{n(t)}\}$. *La fonction* $TF_{t'}^t$ *est définie par :*

$$TF_{t'}^t : l(t, n(t)) \mapsto l(t', n(t'))$$

$$TF_{t'}^t(s_i^{n(t)}, \alpha^{n(t)}) = \Delta_{t'}(\frac{\Delta_t^{-1}(s_i^{n(t)}, \alpha^{n(t)}).(n(t')-1)}{n(t)-1})$$

Des travaux récents ont proposé une méthode alternative pour décrire la sémantique des 2-tuples pour des ensembles de termes non équilibrés [Bartczuk et al., 2012]. Les auteurs définissent ce qu'ils appellent un *terme linguistique étendu* : $\hat{s} = \{s_i, \gamma_i\}$ où s_i est le $i^{\text{ème}}$ terme linguistique d'un ensemble de termes donné et γ_i un *facteur de correction* se traduisant par une translation sur l'axe de valeurs.

Ainsi, l'ensemble de termes de départ S est associé à un ensemble de termes linguistiques étendus $\hat{S}$ ayant le même nombre de termes. Chaque terme linguistique $s_i \in S$ est représenté par un terme linguistique étendu $\hat{s} \in \hat{S}$ dont le facteur de correction garantit

le positionnement à la position souhaitée sur l'axe de valeurs. Le facteur de correction γ_i peut être vu comme une translation symbolique (au sens de Herrera et Martínez) pour lequel la valeur n'est plus bornée en $[-0.5, 0.5[$ mais en $[-i, g]$ où g est le nombre de termes de l'ensemble S.

Bien que cette méthode permette de réduire les calculs nécessaires afin d'attribuer une sémantique aux ensembles de termes non équilibrés, elle présente néanmoins un grand désavantage. En effet, avant l'application du facteur de correction, les termes linguistiques étendus sont symétriquement et uniformément distribués sur l'axe de valeurs. Une fois les facteurs appliqués, les fonctions d'appartenance triangulaires associées à chaque terme sont translatées sur l'axe mais elles subissent une déformation afin de couvrir l'ensemble de l'univers de discours (les triangles sont rétrécis ou élargis sur les cotés selon le cas). Cette déformation induit un changement (injustifié à nos yeux) de la sémantique de chaque terme, ce qui n'est pas le cas dans la méthode fondée sur les hiérarchies linguistiques.

D'autres travaux, récents également, ont permis d'améliorer ce modèle des hiérarchies linguistiques [Espinilla et al., 2011]. On parle d'amélioration car la définition 24 précise la contrainte de passage d'un niveau de la hiérarchie au niveau de granularité supérieur permettant ainsi d'ajouter de nouveaux niveaux à celle-ci. Or, cette contrainte induit un manque de souplesse dans la mesure où la hiérarchie présentée par la figure 2.5 ne permet pas d'ajouter un niveau à sept termes linguistiques, par exemple.

Les *Extended Linguistic Hierarchies* (ELH) (c'est-à-dire hiérarchies linguistiques étendues) répondent donc à cette problématique en levant la contrainte posée par la définition 24. En contrepartie, pour garantir la possibilité d'exprimer des termes d'un certain niveau par leur équivalent dans un autre niveau, un niveau supplémentaire est ajouté. Il a pour nombre de termes le plus petit commun multiple entre les nombres de termes de chacun des niveaux de la hiérarchie. Ce nouveau niveau est donc le dernier niveau de la hiérarchie puisqu'il a la plus grande granularité.

Ainsi, il est garanti que tous les termes de chaque niveau ont leur équivalent dans ce nouveau niveau, ce qui en fait un pivot intermédiaire pour toute transformation de termes d'un niveau à un autre dans la hiérarchie ELH.

Représentation d'un ensemble de termes

La méthode présentée par Herrera et Martínez permet d'obtenir une partition floue pour un ensemble de termes linguistiques de départ en s'appuyant sur une hiérarchie linguistique et sur la modélisation fondée sur les 2-tuples garantissant ainsi un modèle de calcul sans perte d'information.

Pour ce faire, l'ensemble de termes de départ S est partagé en trois : $S = S_L \cup S_C \cup S_R$ où S_C est le sous-ensemble constitué du terme central de S, S_L est le sous-ensemble des termes à gauche de l'élément central et S_R est le sous-ensemble des termes à sa droite.

Le processus de représentation des termes de S par les termes de la hiérarchie linguistique traite séparément les deux ensembles S_L et S_R. Celui-ci étant similaire pour les deux, nous n'allons décrire que la représentation de l'ensemble S_R.

Afin de choisir les termes de la hiérarchie pour représenter les termes de S_R, la première étape est de choisir le niveau à partir duquel seront choisis ces termes. Pour cela, la condition suivante doit être vérifiée : $\exists t \in LH, \frac{n(t)-1}{2} = \#(S_R)$ où $\#(S_R)$ est le nombre de termes de S_R.

S'il existe dans la hiérarchie un niveau t qui satisfait cette condition, alors les termes de S_R sont représentés directement par les termes linguistiques de droite de ce niveau que l'on nomme $S_R^{n(t)}$. Sinon deux niveaux successifs t et $t+1$ de la hiérarchie sont choisis tels que : $\frac{n(t)-1}{2} < \#(S_R) < \frac{n(t+1)-1}{2}$. Ces deux niveaux ont pour nombre de termes à droite respectivement $\#(S_R^{n(t)})$ et $\#(S_R^{n(t+1)})$ qui encadrent $\#(S_R)$ et sont tous deux utilisés dans la représentation des termes de S_R. Ensuite, suivant la densité choisie (*middle* ou *extreme*), k termes de l'ensemble S_R sont représentés par k termes du niveau $t+1$ en partant de la droite pour la densité *extreme* ou en partant du centre pour la densité *middle* (avec $k = \frac{n(t+1)-1}{2} - \#(S_R)$). Les $\#(S_R) - k$ termes restant sont représentés par des termes du niveau n.

Enfin, la moitié descendante (la moitié droite du triangle de la fonction d'appartenance) est représentée par la moitié descendante du terme central du niveau $t+1$ si la densité est *middle* ou du niveau t si elle est *extreme*. Comme dit précédemment, le processus est symétriquement identique pour l'ensemble S_L.

Ainsi, le choix de la densité exprime le besoin d'avoir, lors du processus de représentation, une granularité plus importante autour du terme central ou plutôt à l'extrémité de la partition floue.

Par exemple, pour l'ensemble de départ des notes anglosaxones (*cf.* Figure 2.4) $S = \{E, D, C, B, A\} = \{E\} \cup \{D\} \cup \{C, B, A\}$, une densité *extreme* et en utilisant la hiérarchie linguistique illustrée par la figure 2.5, S_R est représenté par les deux niveaux 2 et 3 ($s_6^9/s_3^5 \rightarrow$ C, $s_7^9 \rightarrow$ B, $s_8^9 \rightarrow$ A), S_C est représenté par s_1^3/s_2^5 et S_L est représenté par s_0^3.
La figure 2.6 résume le processus de la représentation de l'ensemble S et le partitionnement flou obtenu. A la dernière ligne, l'expression linguistique qui pourrait être "presque B" est représentée par le 2-tuple $(s_7^9, -.15)$.

Nous remarquons que pour les termes pivots, C et D, les moitiés ascendantes et descendantes du triangle les représentant, appartiennent à deux niveaux différents de la hiérarchie et peuvent donc être représentés par deux termes linguistiques différents. Par exemple, C peut être représenté soit par s_6^9 ou alors par s_3^5 car les deux termes représentent la même valeur sur l'axe.

Agrégation de termes non équilibrés

Supposons maintenant que l'on souhaite agréger les 6 notes suivantes : D, C, B, C, C, C que l'on note : $(s_2^5, 0)$, $(s_6^9, 0)$, $(s_7^9, 0)$, $(s_6^9, 0)$, $(s_6^9, 0)$, $(s_6^9, 0)$. Tout d'abord, il faut les exprimer dans une hiérarchie commune. On prend la hiérarchie la plus fine (s_2^5 devient donc s_4^9) et on obtient les nouvelles valeurs à agréger : $(s_4^9, 0)$, $(s_6^9, 0)$, $(s_7^9, 0)$, $(s_6^9, 0)$, $(s_6^9, 0)$, $(s_6^9, 0)$.

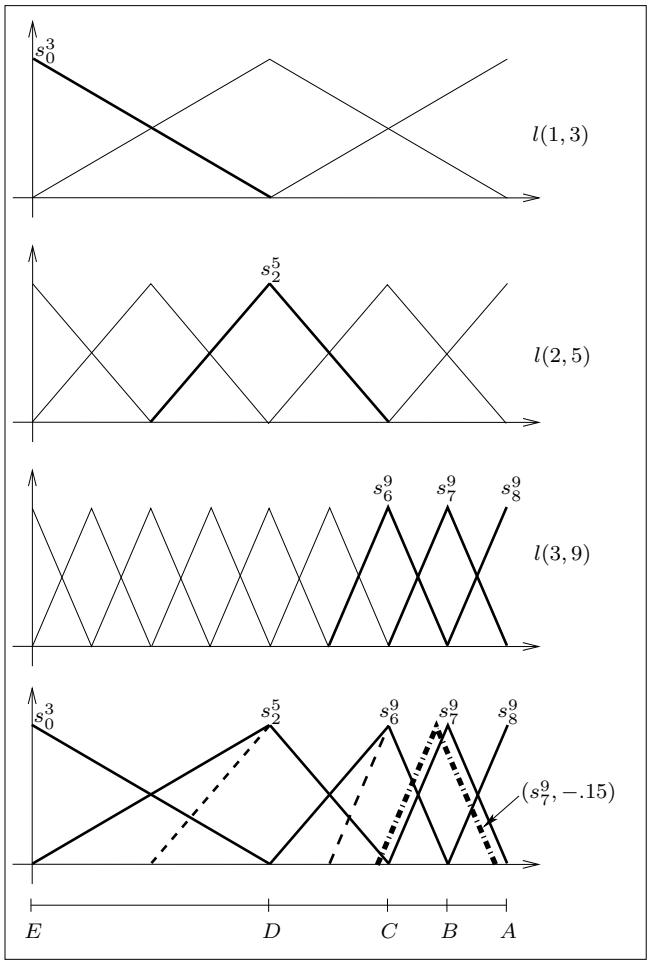


FIGURE 2.6 – Partition floue utilisant une hiérarchie à 3 niveaux.

Ensuite, on choisit l'opérateur, par exemple une moyenne arithmétique, qui utilise uniquement les i. L'agrégation est donc égale à :

$$M(\{(s_0^{n(t)}, \alpha_0), (s_1^{n(t)}, \alpha_1), ..., (s_g^{n(t)}, \alpha_g)\}) = (\Sigma_{i=0}^{g} i)/(g+1)$$

Soit, dans notre exemple $M(\{(s_4^9, 0), (s_6^9, 0), (s_7^9, 0), (s_6^9, 0), (s_6^9, 0), (s_6^9, 0)\}) = (4+6+7+6+6+6)/6 \approx 5.84 = (s_6^9, -.16)$. Le résultat se trouve donc "avant" s_6^9. Dans cette zone, la hiérarchie utilisée est à 5 étiquettes (*cf.* Figure 2.6). Ainsi, il faut renommer $(s_6^9, -.16)$ en $(s_3^5, -.08)$ (α doit être divisé par deux puisque la hiérarchie est deux fois plus grossière). La réponse est donc $(C, -.08)$ que l'on peut garder telle quelle si d'autres calculs sont à venir ou bien nommer linguistiquement : *quasiment C*.

Il faut remarquer que notre exemple utilise 3 hiérarchies différentes réparties comme telles : les valeurs comprises entre s_0^3 et s_2^5 sont dans une hiérarchie à 3 étiquettes, les valeurs comprises entre s_2^5 et s_6^9 sont dans une hiérarchie à 5 étiquettes et les valeurs supérieures sont dans une hiérarchie à 9 étiquettes, comme le montre la figure 2.6. Ainsi, pour agréger les valeurs des évaluations, nous devons tout d'abord passer dans une

TABLE 2.4 – Calcul de la valeur finale de α.

2-tuple	Valeur de α	Opération à effectuer sur α
s_0^9	positive	$\alpha/4$
s_1^9	négative	$0,125 - \alpha/4$
s_1^9	positive	$0,5 + \alpha/4$
s_2^9	négative	$0,5 + \alpha/4$
s_2^9	positive	$-0,5 + \alpha/4$
s_3^9	négative	$-0,375 + \alpha/4$
s_3^9	positive	$-0,125 - \alpha/4$
s_4^9	négative	$\alpha/4$
s_4^9	positive	$\alpha/2$
s_5^9	négative	$0,5 + \alpha/2$
s_5^9	positive	$-0,5 + \alpha/2$
s_6^9	négative	$\alpha/2$

hiérarchie commune à 9 étiquettes et exprimer le résultat dans cette hiérarchie. Il faut ensuite repasser dans la hiérarchie multiple en fonction de la valeur obtenue : pour les résultats compris entre s_0^3 et s_2^5 nous divisons α par 4, pour les résultats compris entre s_2^5 et s_6^9 nous divisons par 2. Le tableau 2.4 donne le détail de la conversion à effectuer pour obtenir la valeur finale de α.

Cet agrégateur peut facilement être remplacé par d'autres opérateurs, notamment les OWA et leurs dérivés déjà évoqués. Il suffit de modifier l'opérateur M. Les différences obtenues entre les agrégations sont bien sûr directement liées aux opérateurs eux-mêmes. Un des éléments les plus importants de ces agrégateurs sont les poids (tous entre 0 et 1 inclus), et plus précisément le vecteur de poids, noté W :

$$W = \bigcup_{i=1}^{p} \{w_i\}$$

La façon de calculer ce vecteur peut être soit le résultat d'un travail empirique, soit le résultat d'un calcul fondé, par exemple, sur une fonction de la famille *Basic Unit-interval Monotonic* (BUM) [Yager, 1988].

Une fonction BUM, notée f_{BUM} est une application de $[0, 1]$ dans $[0, 1]$ qui admet les propriétés suivantes :

- $f_{BUM}(0) = 0$
- $f_{BUM}(1) = 1$
- f_{BUM} est croissante (*i.e.* si $x > y$ alors $f_{BUM}(x) \geq f_{BUM}(y)$)

Le vecteur de poids W est calculé par f_{BUM} de la façon suivante :

$$w_i = f_{BUM}(i/p) - f_{BUM}((i-1)/p)$$

La fonction BUM choisie peut être $f_{BUM}(x) = x$ (dans ce cas, tous les poids valent $(1/p)$) ; ou $f_{BUM}(x) = x^3$ (dans ce cas, w_1 est très petit par rapport à w_p) ; ou $f_{BUM}(x) =$

$\sqrt{x}$ (dans ce cas, w_1 est le poids le plus fort). Pour analyser le choix de f_{BUM}, on mesure habituellement la *orness* ("quantité de OU") du vecteur de poids :

$$orness(W) = \frac{1}{p-1}\sum_{i=1}^{p}(p-i)w_i$$

Cette mesure *orness*, comprise entre 0 et 1, indique à quel point l'agrégateur ressemble à un OU. Quand $orness(W) = 1$, l'agrégateur sera précisément un OU. Quand $orness(W) = 0$, l'agrégateur sera précisément un ET. Par exemple, si $f_{BUM}(x) = x$, $orness(W) = 0.5$. C'est-à-dire que l'agrégateur est exactement à mi-chemin entre un ET et un OU. Si w_1 est beaucoup plus grand que les poids suivants (ils sont **ordonnés**), $orness(W)$ tend vers 1, donc l'agrégateur ressemble à un ET. Bien sûr, ces choix dépendent des utilisations qui seront faites par la suite.

2.1.3 Modélisation à l'aide des 2-tuples proportionnels

Une version alternative des 2-tuples linguistiques a été proposée par Wang et Hao sous le nom de 2-tuples proportionnels [Wang et Hao, 2006]. Cette fois-ci, les données floues sont représentées sous la forme d'un couple $(\alpha l_i, (1-\alpha)l_{i+1})$ où l_i et l_{i+1} sont deux termes d'un ensemble de termes $L = \{l_0, l_1, \ldots, l_n\}$ et α est la proportion symbolique, avec $\alpha \in [0, 1]$. L'ensemble des 2-tuples proportionnels de l'ensemble L est noté $\overline{L}$.
Les auteurs définissent trois fonctions basiques nécessaires au modèle de calcul des 2-tuples proportionnels.

La comparaison

Soient $L = \{l_0, l_1, \ldots, l_n\}$ un ensemble de termes ordonnés et $\overline{L}$ l'ensemble des 2-tuples proportionnels qui en est obtenu. Soit deux 2-tuples $(\alpha l_i, (1-\alpha)l_{i+1})$ et $(\beta l_i, (1-\beta)l_{i+1})$ appartenant à $\overline{L}$. La comparaison de ces deux 2-tuples se fait comme suit :

- si $i < j$, alors
  - $(\alpha l_i, (1-\alpha)l_{i+1})$ et $(\beta l_i, (1-\beta)l_{i+1})$ représentent la même donnée si $i = j-1$ et $\alpha = 0, \beta = 1$
  - $(\alpha l_i, (1-\alpha)l_{i+1}) < (\beta l_i, (1-\beta)l_{i+1})$ sinon
- si $i = j$, alors
  - si $\alpha = \beta$ alors $(\alpha l_i, (1-\alpha)l_{i+1})$ et $(\beta l_i, (1-\beta)l_{i+1})$ représentent la même donnée
  - si $\alpha < \beta$ alors $(\alpha l_i, (1-\alpha)l_{i+1}) < (\beta l_i, (1-\beta)l_{i+1})$
  - si $\alpha = \beta$ alors $(\alpha l_i, (1-\alpha)l_{i+1}) < (\beta l_i, (1-\beta)l_{i+1})$.

La négation

Soit $L = \{l_0, l_1, \ldots, l_n\}$ un ensemble de termes ordonnés, la négation d'un 2-tuple proportionnel est définie par : $Neg((\alpha l_i, (1-\alpha)l_{i+1})) = ((1-\alpha)l_{n-i-1}, \alpha l_{n-i})$ où $n+1$ est la cardinalité de L.

La fonction d'indice de position

La fonction d'indice de position π permet de transformer les 2-tuples proportionnels en des valeurs numériques équivalentes sur l'axe sur lequel ils sont définis. Cette fonction est fondée sur les indices de position des 2-tuples proportionnels à l'image de la fonction Δ (définition 16) des 2-tuples linguistiques.
Elle est définie par : $\pi((\alpha l_i, (1-\alpha) l_{i+1})) = i + (1-\alpha)$ où $i = 0, 1, \ldots, n-1$ et $\alpha \in [0, 1]$.

Ainsi, la fonction π permettant d'obtenir l'équivalent numérique des 2-tuples proportionnels, il est possible d'agréger ces derniers par tout opérateur d'agrégation numérique comme ceux cités en section 2.1.2.

2.1.4 Modélisation à l'aide de modificateurs symboliques

Nous avons vu dans le paragraphe 2.1.1 l'intérêt qu'ont apporté les travaux de Lotfi Zadeh dans la modélisation des données imprécises [Zadeh, 1965, Zadeh, 1975]. Ces recherches ont notamment grandement contribué à la simulation du raisonnement humain qui, par nature, est fondé sur des notions vagues et imprécises. Zadeh a, entre autres, introduit la notion de modificateurs linguistiques (*linguistic hedges*) qui permettent une modélisation et un raisonnement graduels des connaissances afin d'en assurer la plus grande et la plus souple modulation possible en permettant notamment de comparer deux quantités dont seule une est connue [Zadeh, 1972].

Un parallèle peut être fait avec les travaux de De Glas dans le cadre de la logique multivalente initiée par Łukasiewicz [Łukasiewicz, 1920] dans laquelle s'inscrit la théorie des multi-ensembles de De Glas [De Glas, 1989]. En effet, les théories de De Glas et de Zadeh sont relativement comparables, bien que différant essentiellement au sujet de la modélisation des données. Là où, pour Zadeh, les données sont modélisées par des fonctions d'appartenance prenant des valeurs comprises entre 0 et 1 et exprimant l'appartenance d'une donnée à une valeur d'un univers de discours, pour De Glas, ces données sont définies par des multi-ensembles fondés sur des ensembles finis avec une notion de multiplicité de chaque élément.

Dans un multi-ensemble, un élément peut avoir plusieurs appartenances (il peut coexister plusieurs instances d'un élément dans un multi-ensemble), contrairement aux ensembles "classiques" où chaque élément n'en a qu'une seule. De ce fait, l'appartenance y est partielle est s'écrit sous la forme : $x \in_\alpha A$ qui signifie que x appartient à A avec un degré α. On dit aussi que "x est A" est τ_α-vraie, avec τ_α un degré de vérité. Dans sa représentation, De Glas ne garde que l'axe des abscisses comprenant les degrés associés aux valeurs que peuvent prendre les données.

Plus récemment, des travaux ont repris l'idée des théories de Zadeh et De Glas aboutissant à des outils de modification des données linguistiques qu'elles soient sous forme de symboles ou sous forme de sous-ensembles flous [Truck et al., 2001b, Truck et al., 2001a]. Ces outils proposés par les auteurs ont été regroupés sous forme d'opérateurs de modification appelés *Generalized Symbolic Modifiers* (GSM) [Truck et Akdag, 2009].
Ainsi, tout sous-ensemble flou ou symbole peut être considéré (ou du moins exprimé) comme étant le modificateur d'un autre sous-ensemble flou ou symbole respectivement.

Les auteurs définissent un GSM par le biais d'un ensemble ordonné de M degrés de vérité $\mathcal{L}_M = \{\tau_0, \ldots, \tau_i, \ldots, \tau_{M-1}\}$ [14] ($\tau_i \leq \tau_j \Leftrightarrow i \leq j$) avec τ_0 correspondant à faux et τ_{M-1} à vrai.
Ils définissent quatre opérateurs $\vee$ (max), $\wedge$ (min), $\neg$ (négation ou complémentation symbolique, avec $\neg\tau_j = \tau_{M-j-1}$) et l'implication suivante de Łukasiewicz $\rightarrow_L$:

$$\tau_i \rightarrow_L \tau_j = \min(\tau_{M-1}, \tau_{M-1-(i-j)})$$

Ainsi, un GSM est considéré comme une application de $\mathcal{L}_M$ dans $\mathcal{L}_{M'}$, qui associe $\tau_{i'} \in \mathcal{L}_{M'}$ à $\tau_i \in \mathcal{L}_M$ (définition 26) où $\tau_{i'}$ correspond à un GSM noté m_ρ dont le rayon est ρ.

Définition 26.

$$\begin{array}{rccc} m_\rho \colon & \mathcal{L}_M & \rightarrow & \mathcal{L}_{M'} \\ & \tau_i & \mapsto & \tau_{i'} \end{array}$$

De ce fait, m_ρ modifie le couple (τ_i, M) en un nouveau couple $(\tau_{i'}, M')$. La modification opérée par l'opérateur de modification est plus ou moins forte selon le ρ choisi. Les GSM peuvent appartenir à trois familles : affaiblissant, renforçant et centraux suivant l'effet qu'ils ont sur la valeur à laquelle ils sont appliqués. La relation d'ordre existant entre les familles est fondée sur la notion de proportion : $\text{Prop}(\tau_i) = \dfrac{p(\tau_i)}{M-1}$ où $p(\tau_i)$ est la position dans l'échelle du degré i. Un ordre existe bien, du fait que certains GSM sont plus puissants que d'autres.

Le tableau 2.5 résume les différents opérateurs affaiblissants et renforçants définis par les auteurs.

MODE / **NATURE**	**Affaiblissant**	**Renforçant**
Erosion	$\tau_{i'} = \tau_{\max(0,i-\rho)}$, $\mathcal{L}_{M'} = \mathcal{L}_{\max(1,M-\rho)}$ — **EW(ρ)**	$\tau_{i'} = \tau_i$, $\mathcal{L}_{M'} = \mathcal{L}_{\max(i+1,M-\rho)}$ — **ER(ρ)** $\tau_{i'} = \tau_{\min(i+\rho,M-\rho-1)}$, $\mathcal{L}_{M'} = \mathcal{L}_{\max(1,M-\rho)}$ — **ER′(ρ)**
Dilatation	$\tau_{i'} = \tau_i$, $\mathcal{L}_{M'} = \mathcal{L}_{M+\rho}$ — **DW(ρ)** $\tau_{i'} = \tau_{\max(0,i-\rho)}$, $\mathcal{L}_{M'} = \mathcal{L}_{M+\rho}$ — **DW′(ρ)**	$\tau_{i'} = \tau_{i+\rho}$, $\mathcal{L}_{M'} = \mathcal{L}_{M+\rho}$ — **DR(ρ)**
Conservation	$\tau_{i'} = \tau_{\max(0,i-\rho)}$, $\mathcal{L}_{M'} = \mathcal{L}_M$ — **CW(ρ)**	$\tau_{i'} = \tau_{\min(i+\rho,M-1)}$, $\mathcal{L}_{M'} = \mathcal{L}_M$ — **CR(ρ)**

TABLE 2.5 – GSM affaiblissants et renforçants.

La particularité des GSM centraux réside dans le fait qu'il modifient la granularité de l'échelle, et ce, sans modifier la proportion. Nous donnons ici la définition de deux GSM centraux particulièrement intéressants.

14. où M est un entier strictement positif.

Définition 27. *Le modificateur central appelé* DC *est défini comme suit :*

$$\mathrm{DC}(\rho) = \begin{cases} \tau_{i'} = \tau_{i\rho} \\ \mathcal{L}_{M'} = \mathcal{L}_{M\rho-\rho+1} \end{cases}$$

Définition 28. *Le modificateur central appelé* DC$'$ *est défini comme suit :*

$$\mathrm{DC}'(\rho) = \begin{cases} \tau_{i'} = \begin{cases} \tau_{\frac{i*(M\rho-1)}{M-1}} & \textit{si } \tau_{\frac{i*(M\rho-1)}{M-1}} \in \mathcal{L}_{M'} \\ \tau_{\lfloor \frac{i*(M\rho-1)}{M-1} \rfloor} & \textit{sinon (pessimiste)} \\ \tau_{\lfloor \frac{i*(M\rho-1)}{M-1} \rfloor+1} & \textit{sinon (optimiste)} \end{cases} \\ \mathcal{L}_{M'} = \mathcal{L}_{M\rho} \end{cases}$$

En partant des relations d'ordre (partielles) qui lient deux GSM donnés, les auteurs établissent un treillis résumant les relations entre les GSM (figure 2.7).

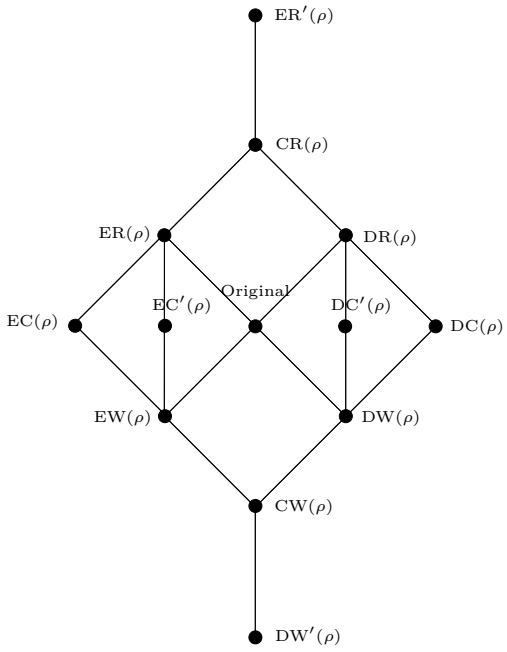


FIGURE 2.7 – Treillis des relations entre GSM.

2.1.5 Discussion

Nous avons essayé de passer en revue l'essentiel des techniques de modélisation et de traitement des données issues de concepts pouvant être vagues ou entachés d'imprécision. Évidemment, il existe d'autres disciplines traitant de cette problématique mais en se

fondant cette fois-ci sur les théories des probabilités ou des possibilités par exemple. Dans la majorité des cas, ces disciplines se concentrent essentiellement sur les techniques de traitement de ce type de données, ne donnant qu'une valeur mineure à la modélisation proprement dite de ces dernières. *A contrario*, toutes les techniques que nous avons citées dans ce chapitre distinguent deux processus : *modélisation* et *raisonnement* à partir des données.

Le traitement des données de départ étant le but premier de toutes les techniques proposées, la phase de raisonnement prend une place très importante dans toutes celles-ci. En effet, le raisonnement implique à la fois l'élaboration d'un modèle de calcul fondé sur un ensemble d'opérateurs (agrégation, négation, t-norme, t-conorme, etc.) et le choix des règles d'inférence qui régiront le système de raisonnement mis en place. En d'autre termes, le raisonnement reflète le comportement attendu du système dans lequel il est implanté.

Ceci dit, un bon raisonnement ne peut avoir lieu sans se reposer sur des données correctement modélisées car un traitement aussi fiable soit-il ne peut donner de bons résultats si les données sur lesquelles il repose ne sont pas fidèles à la réalité. Une modélisation des données de départ, bien adaptée, est donc au moins aussi importante que les traitements qui sont faits à partir de celles-ci. En particulier, la sémantique floue attachée aux données par le biais des partitionnements flous est un point central qui requiert toute notre attention.

Nous allons revenir plus en détail sur ce point dans le chapitre 4.

2.2 Interfaces et traitement de la langue naturelle

Nous avons évoqué les données linguistiques dans leur acception d'imprécision, de la théorie des sous-ensembles flous. Nous allons évoquer leur acception sémantique dans une analyse du langage naturel. Lors de l'analyse en début de chapitre, notre approche utilise les thématiques directement liées au TALN pour le paramétrage des alertes en langage naturel.

Le TALN est une discipline née de l'interaction entre l'informatique (souvent approche IA) et la linguistique (sciences du langage). Ceci lui procure une richesse en terme de domaines d'application allant de la traduction automatique jusqu'aux agents conversationnels en passant par le résumé automatique, le traitement de la parole, la fouille de textes ou encore la classification de documents.

En effet, depuis les années 50, les chercheurs en informatique se sont intéressés au traitement du langage naturel. Ces traitements se fondaient sur l'utilisation de bases de règles écrites manuellement par des experts de la langue (en tirant profit des travaux sur le langage nature) durant les années 60 et 70 avec les travaux de [Winograd, 1972, Weizenbaum, 1966].

Cependant, les progrès réalisés dans le domaine de l'apprentissage automatique (*Machine Learning*) ont permis l'avènement d'une nouvelle conception du traitement du langage naturel reposant sur les arbres de décision et les modèles statistiques afin de comprendre et désambiguïser les phrases d'un texte [Chomsky, 1986, Kuhn et Mori, 1990].

Les analyses sur le langage naturel peuvent se faire selon deux axes principaux d'analyse qui se complètent l'un l'autre : grammaticale et sémantique. En effet, l'analyse d'une phrase selon un seul de ces axes n'est souvent pas suffisante pour en cerner le sens (si nous considérons que le sens d'une phrase est le but premier de son existence). Il est donc souvent nécessaire de combiner différentes analyses afin d'atteindre ce but.

2.2.1 Analyse grammaticale de la langue

L'analyse grammaticale d'une langue joue un rôle important dans le traitement automatique de la langue dans la mesure où la grammaire dicte les règles syntaxiques décrivant les combinaisons de mots donnant des phrases considérées comme correctement formées. L'analyse grammaticale englobe plusieurs types de traitement de la langue comme l'identification de la structure du discours, le partitionnement morphologique ou encore l'analyse syntaxique (*parsing*).

La grammaire dite traditionnelle (Grammaire du Port Royal) décrit un ensemble de règles normatives de fonctionnement du langage [Arnauld et Lancelot, 1803, Foucault, 1967]. Avec la linguistique moderne (début du 20^e siècle), la grammaire devient l'étude de règles qui régissent la construction du langage. Nous trouvons de nouvelles grammaires issues pour la plupart du fait que les gens parlent une langue sans se soucier forcement des règles appliquées par les académiciens. Ces grammaires peuvent être regroupées en trois groupes : les grammaires distributionnelles, les grammaires génératives [Chomsky, 1965, Chomsky, 1975] et, plus récemment, les grammaires relationnelles.

Les grammaires distributionnelles sont issues de l'analyse distributionnelle [Bloomfield, 1935, Harris, 1951]. Ce sont des descriptions linguistiques fondées sur l'observation de la distribution des éléments dans une phrase du texte.

La grammaire générative est née sous l'impulsion de Noam Chomsky qui prône une vision où notre compréhension des phrases est relative à un ensemble de règles sur lesquelles nous nous appuyons souvent inconsciemment. Chomsky y différencie la *capacité* à construire et comprendre des discours (compétence) avec les discours produits eux-mêmes (performance). La performance est considérée comme innée, alors que la compétence est l'objet de la linguistique générative.

Les grammaires relationnelles, quant à elles, sont nées de la conviction qu'il est possible de former une grammaire pour tout système vivant qui obéit à un ensemble de règles [Perlmutter, 1983]. Ces grammaires reposent, comme leur nom l'indique, sur l'analyse relationnelle entre des individus qui communiquent (à l'aide du langage naturel). L'interaction respecte certaines règles définies par la nature de la relation entre les deux individus. Par exemple, lorsque deux personnes discutent, elles s'expriment à tour de rôle (norme conversationnelle). L'analyse relationnelle est à la fois une analyse sémantique et une analyse des interactions.

La syntaxe est la partie de la grammaire qui étudie les règles de la structure des phrases et est concernée par la distribution des mots, les accords et les fonctions dépendant des relations logiques entre les mots. La combinaison des unités lexicales donne le sens des énoncés. Les définitions possibles pour une phrase se regroupent autour de "sens"

et de "logique" : le sens pour la capacité d'exprimer une pensée et la logique qui unit un prédicat et son sujet. La pluralité de la définition de la phrase a fait émerger l'analyse des éléments qui la composent comme les mots invariables, la place des mots dits grammaticaux ou encore la signification de certaines catégories grammaticales comme les adverbes exprimant le temps ou la manière. Ceci est communément appelé : l'analyse des parties du discours.

L'analyse des parties du discours a amené à une universalité de l'analyse linguistique car, même si les parties changent selon les langues, les rôles des parties et les critères d'analyse restent les mêmes. Les critères ont été identifiés et classés selon une diversité de descriptions scientifiques qui a donné différents courants linguistiques : distributionnel, fonctionnel, transformationnel, etc.

Les principes fondamentaux de la théorie du courant du fonctionnalisme d'André Martinet, qui ont par ailleurs inspirés nos travaux, sont l'autonomie, la dépendance et la position [Martinet, 1962]. Toutes ces notions expriment la relation de l'unité syntaxique avec le reste de l'énoncé afin d'en exprimer le sens.

Le courant générativiste a contribué grandement à l'enrichissement et l'implémentation par l'ordinateur du traitement du langage naturel par sa formalisation de la syntaxe. Chomsky croyait à l'autonomie de la syntaxe car la compréhension du langage est divisé, selon lui, en deux composants : la syntaxe formelle (en entrée) et la sémantique (en sortie).

Les différentes études sur la grammaire ont contribué, entre autre, à la création des parseurs qui servent de nos jours de point d'entrée à tous les outils de traitement de la langue naturelle. La plupart des parseurs utilisent un tagueur fournissant du texte prétagué au parseur [Brill, 1992, Charniak et al., 1993, Silberztein, 1994, Brants, 2000] et se fondent sur des techniques d'apprentissage, mais d'autres approches utilisent plutôt un parseur dont la finalité en sortie est un tagueur [Pappa, 2006] permettant ainsi de travailler sur des textes non annotés.

2.2.2 Analyse sémantique de la langue

La sémantique est l'étude des sens des mots ou des phrases (composition de mots) dans leur contexte. Le concept de *sens* est un peu flou car le langage, par nature, peut être flou et imprécis du fait qu'un ensemble de mots peut avoir plusieurs sens.

La complexité de l'analyse sémantique vient du fait qu'un mot polysémique ne peut être étudié qu'à l'échelle de la séquence dans laquelle il apparaît (contexte dans la phrase). La description sémantique ne peut se poser uniquement qu'à l'échelle du mot.

Plusieurs travaux ont été menés dans les divers domaines d'application de l'analyse sémantique. En particulier, l'extraction d'information à partir de corpus de texte est une tâche très récurrente dans le traitement automatique du langage naturel.

Par exemple, des auteurs présentent une méthode permettant d'enrichir un texte donné pour le préparer au processus d'extraction d'informations [Jacquemin et al., 2002]. L'enrichissement du texte se fait en récupérant les synonymes de chacun de ses mots à partir d'un dictionnaire de la langue. Les auteurs apportent une attention particulière

au traitement de la polysémie de certains mots dans le cadre de la désambiguïsation sémantique *Word Sense Desambiguation* (WSD). Ils se fondent sur un dictionnaire ("Les Verbes Français" [Dubois et Dubois-Charlier, 1997]) ainsi que sur une ontologie de tags sémantiques pour le processus de désambiguïsation. Les auteurs construisent une base de règles sémantiques en parsant les exemples de phrases illustrant le sens et la construction syntaxique donnés dans les définitions des mots. Ils construisent des règles du type "R : Quand *conduire* est suivi de *voiture*, alors il prend le sens S".

Ensuite, en se fondant sur les catégories sémantiques de l'ontologie, ces règles sont généralisées pour donner par exemple "R : Quand *conduire* est suivi de *véhicule*, alors il prend le sens S". Les auteurs ajoutent ensuite une base de règles grammaticales construite de la même manière, sauf que cette fois-ci, les liens entre les mots sont d'ordre fonctionnel (*avion* peut être sujet de *décolle*).

Ainsi, trois niveaux de règles sont construits afin de faciliter la désambiguïsation sémantique lors de l'enrichissement du texte :

- règles des mots : information extraite des exemples du dictionnaire ;
- règles du domaine : généralisation des règles précédentes aux mots du même domaine ;
- règles syntaxico-sémantiques : information extraite des *patterns* syntaxiques.

D'autres auteurs abordent également la problématique de l'ambiguïté sémantique engendrée par la polysémie des mots [Prince et Sabah, 1992]. Leurs travaux s'insèrent en tant que module dans un *framework* existant pour le traitement du langage naturel intitulé CARAMEL. Le module proposé est composé d'un lexique sous forme d'un graphe de catégories et sémantiques conceptuelles ainsi qu'un ensemble de règles d'analyse appelées *pragmatic rules*. Ces règles sont définies sous forme de prédicats du premier ordre et représentent les connaissances lexicales que nous avons de l'usage d'un mot : c'est l'expertise linguistique.

La synonymie est une des relations sémantiques les plus problématiques à gérer dans le traitement automatique du langage. La complexité de la gestion de la synonymie vient du fait qu'il n'existe aucune métrique standard permettant de mesurer le "degré" de synonymie entre deux mots donnés.

Cependant, plus récemment, Lafourcade et Prince ont proposé de quantifier la synonymie par ce qu'ils appellent la *distance angulaire* qui leur permet de définir deux types de synonymie entre deux mots : la synonymie relative et la synonymie subjective [Lafourcade et Prince, 2001]. Cette distance angulaire est définie comme une distance sémantique entre deux vecteurs conceptuels V_i et V_j où un vecteur est une combinaison de plusieurs concepts c_i. Ainsi, la synonymie relative est définie comme une synonymie entre deux termes par rapport à un concept donné (qui peut être *via* un synonyme commun aux deux termes par exemple). La synonymie subjective permet de faire de deux termes des synonymes en adoptant un champ sémantique assez éloigné de façon à ce que leur différence devienne négligeable.

L'extraction d'information à partir de corpus est également utilisée pour la construction de dictionnaires terminologiques spécialisés [Ferrari et Prince, 2000] ou encore pour

la création d'une ontologie spécialisée [Makki et al., 2008]. Certains travaux partent d'un corpus monolingue qui sert de base d'extraction afin d'alimenter un dictionnaire multilingue [Ferrari et Prince, 2000]. L'extraction automatique de données passe par plusieurs phases dont la segmentation et l'étiquetage du corpus mais nécessite néanmoins l'intervention d'un expert humain pour résoudre certains cas de polysémie problématiques. Makki et ses co-auteurs présentent une méthode d'enrichissement automatique d'une ontologie générique par des concepts extraits d'un corpus spécialisé (c'est une spécialisation de l'ontologie) [Makki et al., 2008]. L'enrichissement de l'ontologie passe par trois étapes :

- l'annotation du corpus de départ ;
- l'extraction des verbes et des concepts qui leur sont reliés. Chaque verbe est considéré comme une relation entre deux concepts ;
- l'enrichissement proprement dit de l'ontologie par les informations extraites.

L'annotation des parties du discours (*Parts of Speech Tagging* (PST) [Green et Rubin, 1971]) est réalisé par le biais de l'outil *TreeTagger* qui annote grammaticalement le corpus ("vv" pour verbe, "nn" pour nom, etc.). Ensuite, l'ensemble des verbes qui lie deux concepts de l'ontologie de départ est extrait. Un verbe est considéré comme une relation R_{ab} entre deux concepts C_a et C_b appartenant à l'ontologie. Les auteurs définissent les règles d'extraction pour chaque type de relation permettant pour chaque verbe V d'extraire un triplet (W_1, V, W_2) où W_1 et W_2 sont deux ensembles de mots précédant et suivant le verbe V dans le corpus. L'exemple suivant illustre une règle d'extraction d'un verbe suivi de la conjonction "et" :

$$R : w_1 \dots w_i V_{ab} w_j \dots w_k CC w_l \dots w_m \longrightarrow$$
$$(w_1 \dots w_i V_{ab} w_j \dots w_k) et (w_1 \dots w_i V_{ab} w_l \dots w_m)$$
$$\text{où } CC = \text{ET (le terme)}$$

Dans la même thématique d'enrichissement, d'autres travaux ont abordé la création automatique d'ontologies par la prédiction de relations entre deux termes donnés [Tisserant et al., 2012] ou encore l'enrichissement d'une base de connaissances (sous forme de graphes) par le dialogue entre des agents cognitifs [Yousfi-Monod et Prince, 2007].

En plus de l'extraction d'information, l'interprétation sémantique représente une part importante dans le traitement du langage naturel. Dans [Prince, 1994], l'auteur propose un "lexique intelligent" afin d'interpréter sémantiquement un mot donné. Le lexique intelligent regroupe à la fois une structure descriptive des connaissances (réseau sémantique, graphe de concepts, dictionnaire, etc.) et un système d'inférence permettant de raisonner à partir de ces connaissances. Ce mécanisme de raisonnement regroupe trois types de règles : les règles pragmatiques, les contraintes sémantiques et les règles de raisonnement par défaut. Le formalisme choisi par l'auteur associe chaque mot à son *potentiel*, c'est-à-dire le regroupement des concepts qui sont reliés, des caractéristiques (*features*) décrivant chacun de ces concepts et les liens reliant chaque caractéristique à son concept. Les trois types de règles de raisonnement sont ensuite définis pour ce formalisme afin de choisir, lors de l'interprétation, le sens le plus pertinent selon le contexte.

Les travaux que nous avons passés en revue permettent de guider nos réflexions à plusieurs niveaux quant aux traitements sémantiques de la langue. Tout d'abord, nous constatons qu'une analyse efficace nécessite un certain nombre de pré-traitements et de représentations de connaissances facilitant notamment la gestion des cas particuliers de polysémie, de désambiguïsation sémantique ou encore de synonymie. Aussi, la quantification de la sémantique des mots reste un problème entier malgré les différentes approches proposées et notamment celle de Lafourcade et Prince où une quantification de la synonymie a été introduite. Les outils de la logique floue, et plus particulièrement le CW, peuvent apporter une approche alternative où les termes du langage naturel peuvent être représentés, de façon qualitative, par des variables linguistiques. Les termes du langage peuvent donc puiser leur "signification" dans une variable linguistique. Bien évidemment, cette approche ne peut être appliquée à tous les mots du langage car elle s'apparente davantage à la représentation des adjectifs et des adverbes qu'au reste des mots de la langue.

Nous allons donc maintenant discuter des travaux qui se sont intéressés conjointement aux deux disciplines que sont la théorie des sous-ensembles flous et le traitement de la langue naturelle.

2.3 Traitement "flou" du langage naturel

Dans la littérature nous retrouvons certains travaux mêlant les techniques de traitement du langage naturel à celles issues de la logique floue.

Dans [Huang et al., 2006], les auteurs utilisent les ensembles approximatifs (*fuzzy rough sets*) pour extraire des phrases-clefs d'un document afin d'en constituer le résumé. D'autres auteurs proposent un modèle générique "expert métier du bois" pour modéliser des connaissances métier dans un formalisme particulier (méthode NIAM) en s'appuyant sur un système de règles linguistiques floues [Vincent et al., 2004].

Par ailleurs, d'autres travaux s'appuient sur des règles d'inférence floue pour améliorer la reconnaissance des mots (dans le cadre d'une reconnaissance vocale) en liant chaque mot à son contexte afin de déterminer s'il a été correctement reconnu ou non [Sun et al., 2002].

D'autres auteurs utilisent les techniques de la logique floue pour la désambiguïsation sémantique. Par exemple, Bergmair et Bodenhofer utilisent des relations floues comme représentation des concepts (flous) du langage naturel afin d'enrichir de sémantique les règles de production d'une grammaire [Bergmair et Bodenhofer, 2006]. Diou *et al.*, quant à eux, considèrent le problème de la désambiguïsation comme un problème d'associations imprécises entre les mots et leur sémantique [Diou et al., 2006]. Les sous-ensembles flous sont utilisés pour fournir un degré d'association entre un sens et un mot en s'appuyant sur les définitions données par le dictionnaire en ligne WordNet. Par la suite, ils proposent un algorithme de classement des sens d'un mot dans un texte donné. Ce dernier associe un sous-ensemble flou à chaque sens du mot en question avec des degrés d'appartenance différents. Lors de la désambiguïsation, le sens sélectionné pour un mot est celui dont le degré d'appartenance est le plus élevé pour un contexte donné.

D'autres travaux se sont également intéressés à l'étude de l'articulation entre les techniques issues de la logique floue et du traitement du langage naturel [Novák, 1991, Glöckner et Knoll, 1997].

Dans [Novák, 1991], l'auteur présente un modèle mathématique appelé AML (*Alternative Mathematical model for natural Language semantics*) où il se donne pour objectif de décrire le langage naturel (et plus particulièrement une phrase ou une expression donnée) à l'aide des outils de la logique floue. Ainsi, le sens d'une phrase est déduit en composant les sens des mots, décrits par des sous-ensembles flous, en appliquant les relations floues qui lient les mots entre eux (ces relations floues étant également définies par les auteurs). Le sens global d'une phrase est donc obtenu par une inférence floue dictée par les relations définies entre les mots la composant. Cette approche, bien qu'étant originale et ambitieuse, reste néanmoins théorique dans la mesure où elle n'intègre pas les connaissances dans ce processus de déduction de sens (en particulier, la polysémie des mots rend difficile l'application et la généralisation de cette approche).

Par la suite, ces travaux ont donné lieu à une approche plus spécifique où l'objectif est de modéliser une expression linguistique (au sens large) par une ou plusieurs expressions logiques [Novák, 1992, Novák et Perfilieva, 1999]. Les auteurs décrivent les expressions linguistiques sous forme de prédicats et d'implications flous dont la sémantique est modélisée par des quantificateurs linguistiques [Mostowski, 1957]. Ils se focalisent, à cause de l'ampleur de l'objectif, sur l'évaluation (au sens logique) des expressions de type affirmatif et de l'antonymie.

Une approche assez similaire a été présentée par Glöckner et Knoll [Glöckner et Knoll, 1997]. Les auteurs y proposent d'améliorer un système de requêtes et de récupération d'information à l'aide de la gestion fine des quantificateurs flous. Ces quantificateurs flous sont l'extension des quantificateurs (encore appelés déterminants) issus de la théorie des quantificateurs généralisés *Theory of Generalized Quantifiers* (TGQ) [Barwise et Cooper, 1981]).

Dans cette théorie, un déterminant D associe à chaque sous-ensemble $(X_1, \ldots, X_n)$, appartenant à un ensemble d'éléments E, un vecteur de valeurs de vérité lui correspondant $D(X_1, \ldots, X_n) \in 2$. Un déterminant est donc une fonction (*cf.* définition 29).

Définition 29. *Un déterminant n-aire d'un ensemble d'éléments E est une fonction $D : \mathcal{P}(E)^n \to 2$ où $2 = \{0, 1\}$ est un semble de valeurs de vérité et $\mathcal{P}(E)$ est l'ensemble des sous-ensembles de E.*

Voici quelques exemples de déterminants connus :

$$\forall(X) = 1 \Leftrightarrow X = E$$
$$\exists(X) = 1 \Leftrightarrow X = \emptyset$$
$$\text{tous}(X_1, X_2) \Leftrightarrow X_1 \subseteq X_2$$
$$\text{quelque}(X_1, X_2) \Leftrightarrow X_1 \cap X_2 \neq \emptyset$$
$$\text{au\_minimum\_}n(X_1, X_2) \Leftrightarrow X_1 \cap X_2 \geq n$$

Les déterminants de base étant binaires ($2 = \{0, 1\}$) les auteurs les ont étendus à des déterminants flous dont les valeurs de vérité sont obtenues cette fois-ci *via* des fonctions

d'appartenance. Ainsi, pour chaque sous-ensemble flou X d'un ensemble E est associée une valeur d'appartenance $\mu_X(x) \in I$ où $I = [0, 1]$ et $x \in E$.

Définition 30. *Un déterminant flou n-aire d'un ensemble d'éléments E est une fonction $D : \widetilde{\mathcal{P}}(E)^n \to I = [0, 1]$.*

Les auteurs se servent des déterminants flous pour représenter certains termes et quantifier (évaluer) des expressions et requêtes en langage naturel. Ils démontrent l'utilité de cette approche dans le cas spécifique des requêtes composées où les "opérateurs" de compositions ne sont pas toujours les conjonctions *et* et *ou* mais peuvent être des quantificateurs du langage naturel (par exemple *autant que possible*).

2.4 Discussion et limites des modèles existants

2.4.1 Univers, approximation et modèles computationnels

En général, les univers sur lesquels tous les modèles évoqués dans la section 2.1 s'appuient sont continus. Mais les humains, eux, manipulent des concepts ("petit", "moyen", "grand", etc.) exprimables le plus souvent sur des intervalles (univers) discrets. D'où le besoin de "discrétiser" l'espace avec, par exemple, des sous-ensembles flous pour représenter les termes (*cf.* les partitionnements flous). En quelque sorte, on peut dire que le partitionnement flou permet de conserver un intervalle continu pour les calculs (à un "bas niveau", donc) mais en faisant apparaître (à un "haut niveau") les termes linguistiques associés au partitionnement (donc l'intervalle redevient discret).

Le CW, en tous cas, tel qu'il est exposé dans les modèles de type "2-tuples", permet donc de passer d'un univers continu à un univers discret, tout en conservant, en interne, la souplesse d'un univers continu.

Mais en "flou pur", et *via* le *modus ponens* généralisé, le résultat obtenu après application de (toutes ou certaines) règles, ou, plus généralement, après application d'agrégateurs sur ces sous-ensembles flous, n'est pas toujours facilement exprimable dans ou avec des termes linguistiques, c'est-à-dire dans l'*intervalle* (univers) *de départ*.

Ainsi, pour l'expression du résultat obtenu, nous notons plusieurs méthodes :

(i) dans le cas du "flou pur", on peut utiliser la défuzzification (*cf.* page 22) lors de laquelle le y_0 obtenu (*cf.* figure 2.8) est exprimé dans un intervalle continu ;

(ii) toujours dans le cas du "flou pur", on peut utiliser des *modificateurs flous* qui expriment la ressemblance du sous-ensemble flou résultat avec un des sous-ensembles flous solutions, tels que prévus dans les conclusions des règles (*cf.* figure 2.9). Notons cependant que la plupart des applications utilisant le "flou pur" (comme les contrôleurs flous) se préoccupent assez peu de la question de l'expression du résultat dans l'ensemble *de départ* et se satisfont, le plus souvent, de la valeur y_0 ;

(iii) dans les formalismes de type 2-tuples, on constate que :
- dans le cas des 2-tuples linguistiques de Herrera & Martínez où les données sont représentées sous forme de termes (s_i) associés à des sous-ensembles flous triangulaires (*cf.* la figure 2.10 et également la définition 17 en page 25), si la valeur

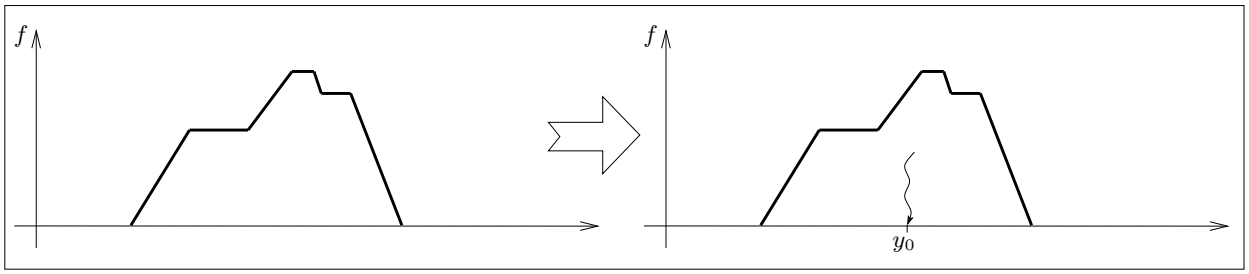


FIGURE 2.8 – Défuzzification : y_0 est une valeur exprimée dans un univers *continu*.

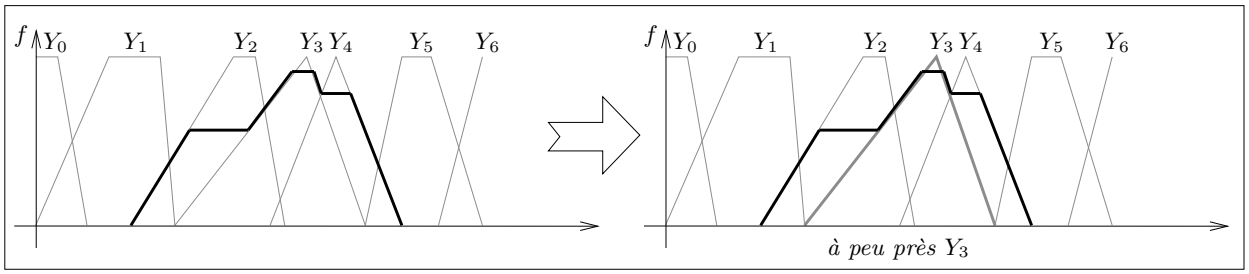


FIGURE 2.9 – La solution est exprimée dans l'univers de départ au moyen de modificateurs flous. Ici, il s'agit du modificateur *à peu près*.

à exprimer défuzzifiée (ici i, dans l'univers discret) ne coïncide pas exactement avec un des termes de départ (s_i), il faut revenir dans l'univers continu et utiliser une translation symbolique α_i qui supporte la perte d'information liée à l'approximation. Finalement, les *modificateurs* sont, dans ce modèle, les α_i ;
- dans le cas des 2-tuples proportionnels de Wang & Hao (*cf.* la section 2.1.3), le modèle est très proche de celui de Herrera & Martínez, sauf que l'approximation est portée par une proportion symbolique notée α également (*cf.* figure 2.11). Comme chez Herrera & Martínez, les *modificateurs* sont, dans ce modèle, les proportions α ;

(iv) dans le formalisme de Truck & Akdag, les auteurs considèrent seulement et exclusivement des univers discrets (ordonnés) et expriment l'approximation grâce à des modificateurs symboliques (*cf.* les GSM, page 35 ainsi que la figure 2.12 où un modificateur affaiblissant par dilatation est utilisé). Leur médiane symbolique pondérée est d'ailleurs une *fonction composée* car obtenue par composition de fonctions, lesdites fonctions étant les GSM. Ainsi, les auteurs résolvent le problème de l'approximation en changeant la granularité de leur univers par conservation, érosion ou dilatation de l'échelle.

Concernant maintenant les modèles computationnels associés aux trois derniers formalismes évoqués, on peut noter que :
- pour les 2-tuples linguistiques, le modèle sous-jacent est fondé sur le principe d'extension, c'est-à-dire qu'il utilise des fonctions "classiques", issues des mathéma-

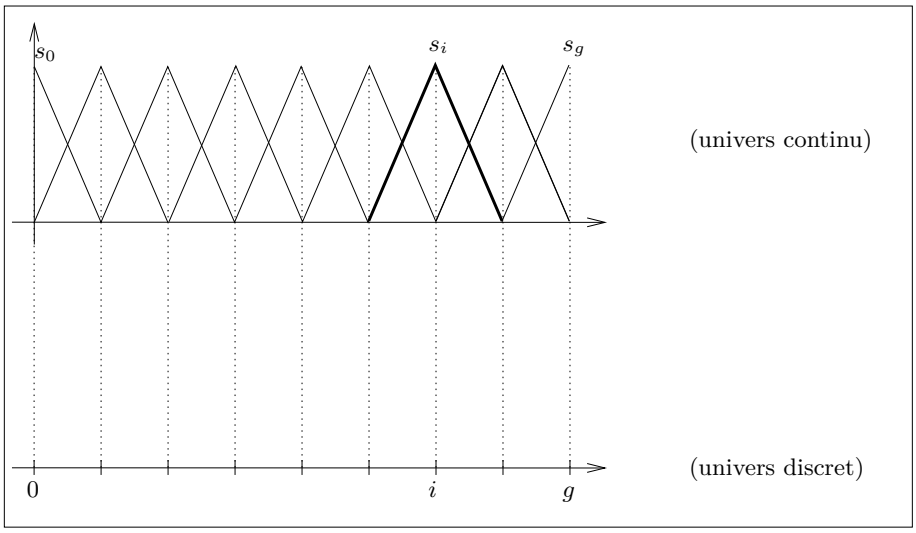


FIGURE 2.10 – Passage de l'univers *continu* (avec les sous-ensembles flous triangulaires et les termes s_i) à un univers *discret*.

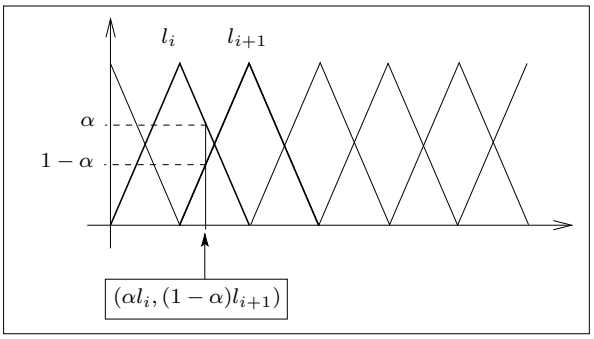


FIGURE 2.11 – Le 2-tuple correspondant à la valeur pointée est $(\alpha l_i, (1-\alpha) l_{i+1})$.

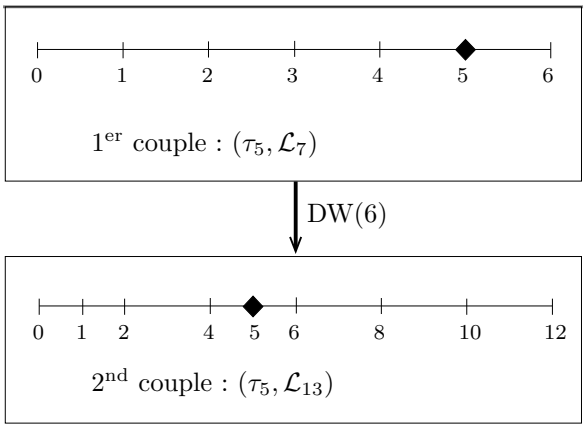


FIGURE 2.12 – Modification de l'échelle par dilatation (ici, utilisation du GSM DW(6)).

tiques (agrégateurs comme la moyenne arithmétique (éventuellement pondérée), la moyenne géométrique, etc.) qu'il étend aux 2-tuples linguistiques ;
- pour les 2-tuples proportionnels, c'est la même chose que chez Herrera & Martínez, mais les opérateurs sont étendus aux 2-tuples proportionnels ;
- pour le modèle de Truck & Akdag, les opérateurs d'addition, de soustraction, de multiplication et de division sont entièrement redéfinis en utilisant notamment le max, le min et la négation [Truck et Akdag, 2006] ainsi que l'opérateur de médiane symbolique pondérée déjà évoqué. Chacune de ces définitions utilise largement l'implication de Łukasiewicz.

2.4.2 Limites des modèles existants

Nous avons passé en revue dans ce chapitre de nombreuses techniques abordant essentiellement deux axes de recherche : la modélisation des données vagues et imprécises et le traitement automatique du langage naturel.

La théorie des sous-ensembles flous a montré qu'elle très pertinente pour manipuler les imprécisions, ce dont il est notamment question dans notre problématique, et qu'elle permettait un raisonnement approximatif grâce au concept d'inférence floue (*modus ponens* généralisé et règles d'inférence floue).

Mais une des limites que nous avons soulevées est l'impossibilité d'exprimer le sous-ensemble flou obtenu comme solution (suite à l'inférence ou suite à une agrégation de sous-ensembles flous) comme l'un des sous-ensembles flous de départ.

C'est pourquoi nous nous sommes tournés vers les 2-tuples linguistiques de Herrera et Martínez qui offrent une approche alternative à celle de Zadeh en modélisant les données sous forme de couples (s, α). Cette approche tire partie de la notion de translation symbolique afin de proposer un modèle de calcul permettant un raisonnement sans perte d'information. Elle s'appuie également sur les hiérarchies linguistiques afin de modéliser de façon cohérente les données non uniformes. De surcroît, nous avons présenté une approche qui s'inspire de celle des 2-tuples linguistiques mais qui modélise les données sous forme de couples $(\alpha l_i, (1-\alpha) l_{i+1})$ appelés 2-tuples proportionnels.

Qu'il s'agisse des 2-tuples proportionnels ou des 2-tuples linguistiques, ils n'ont pas, à ce jour, été utilisés dans des cas concrets de modélisation de données de type spatial (voir notre définition du problème dans le chapitre 1 où il est question notamment de distance à un point de passage (entrée/sortie de zone, corridor)), ni, semble-t-il, utilisés en entreprise avec des cas réels. En particulier, il semble que la construction du partitionnement de l'espace (l'univers de discours) soit rigidifiée par les ensembles de termes S_L, S_C et S_R qui imposent, de surcroît, l'existence d'un terme central et d'un nombre impair de termes (*cf.* page 30). Même les hiérarchies ELH ne résolvent pas ce problème.

Nous avons également abordé l'approche dite purement symbolique qui se donne comme objectif de ne modéliser les données que par des symboles en s'appuyant notamment sur des modificateurs symboliques comme les GSM, par exemple, que nous avons passés en revue. Bien qu'intéressante car proche, dans l'esprit des précédents travaux, cette démarche ne nous convient pas car trop restrictive en termes d'objets à manipuler. En effet, les symboles doivent être ordonnés mais placés dans des univers discrets.

On pourrait s'y ramener, mais il semble plus simple et naturel de conserver des univers continus.

Dans le paragraphe suivant, nous avons présenté un ensemble de travaux qui portent sur le TALN. Ces travaux ont été classés selon deux axes complémentaires : l'analyse grammaticale et l'analyse sémantique de la langue.

Enfin, nous avons évoqué certaines recherches qui se sont intéressées à l'articulation qu'il peut y avoir entre la modélisation de données imprécises et le traitement du langage naturel. Ces travaux emploient des techniques de modélisation issues de la logique floue afin de traiter des problèmes spécifiques du traitement automatique du langage comme la désambiguïsation sémantique, le résumé de texte, l'évaluation sémantique d'une expression ou encore la quantification de certains termes qualitatifs du langage naturel. Cependant, à notre connaissance, aucune recherche n'a été faite dans le cadre qui est le nôtre, ni dans le sens dans lequel nous souhaitons aller, à savoir une interface homme-machine utilisant le langage naturel pour définir des alertes en géolocalisation et prenant en compte les imprécisions des données manipulées.

Nous conjecturons que les techniques de TALN seront tout à fait adéquates pour compléter le traitement par la logique floue de connaissances imprécises formulées sous forme langagière. Ainsi, le cahier des charges que nous nous proposons d'adopter est le suivant. D'abord, il faut constituer un lexique des termes métier, dédié au problème de géolocalisation. Ce lexique permettra d'alimenter avec des mots réels les ensembles de termes linguistiques précédemment évoqués dans les travaux autour des approches floues. Après l'étude du lexique, il conviendra de proposer une interface homme-machine pour la programation des alertes et de présenter des tests préliminaires. La modélisation du lexique devra prendre en compte les modèles existants (modèle des 2-tuples linguistiques notamment) mais également étudier la possibilité d'étendre ces modèles selon nos besoins, vu les limites évoquées plus haut. Des algorithmes pourront être développés pour modéliser ces extensions théoriques.

Chapitre 3

Vers une explicitation des choix

Sommaire

3.1 Introduction

Comme nous l'avons introduit dans le chapitre 1 et notamment en 1.3.1, les applications en géolocalisation se déclinent souvent en programmation événementielle consistant à suivre la position d'éléments mobiles (boîtiers GPS, localisation cellulaire de téléphones portables) de manière à détecter et à réagir à des occurrences d'événements (une sortie de corridor, par exemple) qui vont déclencher des alertes devant être traitées (automatiquement ou non).

La mise au point de telles applications se heurte à une difficulté majeure qui vient du fait que la géolocalisation se fait à l'aide d'appareils spécifiques embarqués ou de téléphones mobiles, sur la base de remontées régulières de positions, puis sur leur traitement pour corréler les positions avec de potentiels événements à signaler.

Aujourd'hui, l'expression de ces corrélations et de la configuration des fréquences de remontées de positions des appareils de localisation s'effectue directement selon de nombreux paramètres techniques qui ne sont pas facilement compréhensibles par les utilisateurs finaux. D'autant plus qu'ils comportent des contraintes dans leur fixation qui sont difficiles à respecter lorsque le nombre de paramètres de géolocalisation devient élevé, ce qui est le cas des applications actuelles.

Nous nous sommes donc interrogés sur la façon dont on pouvait améliorer les interfaces existantes dans le cadre de la géolocalisation, de sorte de rendre qualitatif le traitement des événements. De surcroît, il faut pouvoir traduire les préférences et les besoins métier en alertes et configurations à bas niveau des boîtiers liés aux mobiles (côté *back office* en quelque sorte), tout en conservant (côté *front office*) un dialogue en langage naturel.

Une première analyse de cette problématique nous permet d'emblée de distinguer deux sous-problèmes complémentaires permettant d'y répondre.

- Premièrement, il faudrait réduire la complexité des interfaces graphiques actuelles en offrant une interaction simple et intuitive aux utilisateurs. En effet, une interface graphique idéale cache toute la complexité du logiciel à l'utilisateur lui offrant ainsi l'information de la manière la plus naturelle possible.
  Idéalement, les utilisateurs devraient pouvoir s'exprimer dans leurs propres mots (souvent dans un jargon du domaine) sans avoir à s'adapter systématiquement aux différents outils dont ils font usage dans une vision du génie logiciel où c'est l'interface qui s'adapte à l'utilisateur et non l'inverse [Rosson et Carroll, 2009].
  Dans ce sens, le langage naturel constitue une "interface" intuitive dans la mesure où toute personne sait utiliser sa lange maternelle sans réel effort. Par conséquent, fournir une interface de dialogue en langage naturel peut lever les barrières techniques permettant d'accéder à l'ensemble des fonctionnalités d'un logiciel y compris les plus avancées d'entre elles. D'autant plus que dans notre cas, l'expert du domaine pourrait y exprimer ses objectifs métiers et laisser le soin à l'interface de traduire ces derniers en configurations adéquates. Une interface en langage naturel semble donc une bonne alternative aux interfaces utilisateurs classiques (formulaires, applications web, composantes graphiques, etc.).
- La deuxième partie du problème concerne l'**interprétation sémantique**. En effet, le passage à un contexte qualitatif, en remplacement du contexte quantitatif actuel, nous pose la problématique quant à la sémantique à rattacher aux termes que peut employer l'utilisateur. D'autant plus que l'humain par nature exprime ses besoins au niveau du sens : ceci impose d'attacher un soin particulier à la compréhension de ce sens (et, par extension, des besoins exprimés).
  Des concepts comme *proche*, *loin*, *fort*... sont, par définition, flous. Il n'existe aucun standard quant à la limite sémantique distinguant les deux termes *proche* et *loin*. Cette problématique sera traitée au chapitre 4.

Offrir une interface fondée sur le dialogue en langage naturel pose donc un certain nombre de verrous techniques dus notamment à la complexité générée par ce genre de traitement donnant lieu à un problème NP-complet. Un des points clés reste donc la réduction de la complexité et des traitements de calcul qui en découlent.
De plus, l'analyse sémantique nécessite à la fois la mise en place d'une base de connaissance conséquente afin de cerner au mieux le sens du dialogue en cours, ainsi que le traitement des imprécisions et la subjectivité inhérentes au langage naturel et à la nature du raisonnement humain.

Dans ce qui suit, nous proposons une approche d'explicitation des choix fondée sur une interface de dialogue en langage naturel permettant aux experts du domaine d'ex-

primer leurs besoins métiers, configurer les boitiers GPS et créer facilement des alertes en géolocalisation.

3.2 Intégration de la langue naturelle

Nous souhaitons offrir aux experts la possibilité d'exprimer leurs besoins et objectifs métiers *via* une interface de dialogue en langage naturel [Melekhova et al., 2010]. Cette interface peut aboutir à terme à un assistant vocal de géolocalisation complet.

Cet assistant pourrait ainsi permettre (dans notre cadre de géolocalisation) de récupérer des informations sur l'état des mobiles d'un utilisateur, changer leur configuration, programmer des alertes (au sens Deveryware) et même offrir des fonctionnalités de haut niveau qui se traduiraient en un enchaînement de configurations et/ou création d'alertes.

Aussi, l'approche adoptée doit être facilement généralisable pour la transposer à d'autres domaines à moindre effort.

L'approche que nous proposons s'inscrit dans celle des systèmes à questions-réponses (*Question Answering Systems*) dans lesquels le programme est *leader* du dialogue, c'est-à-dire que c'est le programme qui engage l'échange et pose les questions. À chaque réponse obtenue, le programme l'analyse et décide soit de l'action à exécuter, soit de la prochaine question à poser selon un scénario donné [Abchir et al., 2012a].

La première étape de la construction de l'interface de dialogue est la construction d'un lexique de base qui va servir à analyser les phrases de l'utilisateur.

3.2.1 Corpus

Pour parvenir à la construction de ce lexique, nous constituons un corpus à partir de divers documents de la société Deveryware :

- dix plaquettes commerciales : chaque plaquette est un document constitué d'une à une dizaine de pages décrivant les produits et services de Deveryware. Chacune des plaquettes peut être considérée comme un corpus spécialisé dans la mesure où elle vise un marché particulier et utilise par conséquent le jargon métier approprié (sécurité, télé-surveillance, logistique...). Ces plaquettes sont usuellement distribuées par les services commerciaux et marketing lors de réunions avec des partenaires potentiels ou lors des salons d'expositions ;
- un manuel d'application : nous avons joint au corpus le manuel d'utilisation de l'application Web principale de Deveryware nommée *Deveryloc*. Celui-ci est composé d'un document d'une quarantaine de pages décrivant les différents aspects fonctionnels de l'application. Nous avons choisi Deveryloc car, d'une part, elle est assez représentative du cœur de métier de Deveryware, et d'autre part, elle est vouée à être accompagnée, voire remplacée (en situation de mobilité) par l'interface en langage naturel ;
- un fichier de langue : nous avons également inclus dans le corpus de base le fichier dit "fichier de langue" de Deveryloc. En effet, en génie logiciel, les textes affichés

dans les applications ainsi que leurs composantes (labels, boutons, titres des menus, aide, etc.) sont regroupés dans un seul fichier texte de type clé-valeur (dit *fichier de langue*) pour chaque langue supportée par le logiciel en question. Les valeurs sont chargées dynamiquement lors du lancement du logiciel et seul le fichier de langue correspondant à la langue choisie par l'utilisateur est utilisé.
Cette pratique a pour but de centraliser le texte affiché par un logiciel, de séparer le fond (le code) de la forme (le texte) et de simplifier la traduction des logiciels puisque, étant isolé du reste du code de ces derniers, seul le fichier de langue est envoyé au sous-traitant réalisant les traductions pour une entreprise donnée ;

– la documentation de l'*Application Programming Interface* (API) de géolocalisation : la documentation de l'API de géolocalisation fournie par Deveryware à ses clients se présente sous la forme d'un document de 129 pages. Cependant, la documentation inclut à la fois la description fonctionnelle des diverses fonctions accessibles *via* l'API ainsi qu'une description technique de ces dernières (nom des fonctions, type et nombre de paramètres, type de retour...). Nous décidons donc de retirer la documentation technique (destinée essentiellement aux développeurs) pour ne garder que la partie fonctionnelle qui, par définition, est la plus utile pour notre lexique. Ainsi, le document ajouté au corpus fait 50 pages.

3.2.2 Création du lexique

Une fois le corpus réuni, nous effectuons une étude statistique sur les occurrences des mots du corpus en excluant, bien sûr, les mots grammaticaux. Cette première étude nous permet de faire ressortir les mots les plus pertinents du jargon du domaine de la géolocalisation et qui doivent donc figurer dans le lexique de base de l'interface de dialogue. La méthode *Term Frequency-Inverse Document Frequency* (TF-IDF) qui consiste à calculer les poids des termes dans le corpus, s'avère très efficace pour cette étude.

Ensuite, nous regroupons ces mots (*tokens*), expressions et parties du discours dans un fichier XML dans le cadre d'un *Parts of Speech Tagging* (PST) où chaque entrée du lexique reçoit un ensemble de *tags* descriptifs.

Tout d'abord, chaque entrée est étiquetée par un *tag* grammatical renseignant sur la catégorie grammaticale de cette dernière (nom, verbe, adjectif, adverbe, etc.). Ces *tags* servent notamment à désambiguïser certains mots de la langue. Par exemple le mot *proche* est à la fois un nom (comme dans "un proche" ou "de proche en proche") et un adjectif (comme dans "un lien proche") [Harris, 1991]. Ce genre d'ambiguïtés est fréquent dans le langage naturel et doit être pris en considération dans l'analyse de phrases ou texte.

Cette opération (*PoS tagging*) peut être réalisée à l'aide d'un outils de *tagging* automatique tels que *TreeTagger*[15] déjà évoqué en section 2.2.2, page 42. *TreeTagger* a pour avantage de regrouper à la fois le *PoS tagging* et la lemmatisation. Il permet de présenter les termes étiquetés également sous leur forme canonique, et ce, quelle que soit la forme fléchie rencontrée (pluriel d'un mot, formes conjuguées, etc.).

15. *TreeTagger* : *cf.* `http://www.cis.uni-muenchen.de/ schmid/tools/TreeTagger/`

Par la suite, nous ajoutons des *tags* sémantiques aux entrées du lexique les plus pertinentes d'un point de vue métier. Ces *tags* servent de représentation des connaissances et permettent, entre autres, de lier les entrées rattachées aux mêmes concepts sémantiques. Cette phase de la conception du lexique métier est réalisée de manière empirique car elle nécessite la connaissance métier d'un expert du domaine.

L'exemple suivant illustre un extrait du lexique constitué. Nous y représentons une phrase qui peut être rencontrée dans l'interface de dialogue : "Je veux créer une alerte quand le camion se rapproche nettement de l'entrepôt".

Extrait du lexique métier.

```
<?xml version="1.0" encoding="UTF-8"?>
 <tokens>
        <token gram="PRON">je</token>
        <token gram="VERB">veux</token>
        <token gram="VERB">créer</token>
        <token gram="DET">une</token>
        <token gram="NOUN" sem="ALERT">alerte</token>
        <token gram="CONJ" sem="ALERT_COND">quand</token>
        <token gram="DET">le</token>
        <token gram="NOUN" sem="VEHICULE">camion</token>
        <token gram="VERB" sem="ENTREE_ZONE">se rapproche</token>
        <token gram="ADV" sem="MODIF_FUZZY">nettement</token>
        <token gram="ADP">de</token>
        <token gram="DET">l'</token>
        <token gram="NOUN" sem="POI">entrepôt</token>
 </tokens>
```

Ce petit exemple montre à la fois les *tags* grammaticaux appelés "`gram`" et ceux sémantiques appelés "`sem`". Les *tags* sémantiques peuvent représenter des éléments "génériques" (véhicules, personnes, villes, etc.) ou des éléments spécifiques au domaine de géolocalisation (types d'alertes, points d'intérêts, type de mobiles, etc.).

Une fois que nous avons créé le noyau de base de notre lexique tagué, nous cherchons à l'enrichir afin qu'il soit le plus exhaustif possible et que l'interface de dialogue prenne en charge le maximum de termes susceptibles d'être rencontrés. Nous pouvons nous permettre cela car nous nous plaçons dans le cadre d'un dialogue en langage naturel dans un domaine clos. La taille du lexique par conséquent reste relativement réduite. Néanmoins, puisque l'ensemble des synonymes constitue un champ sémantique très large, constituer des sous-ensembles contextuels (ici géolocalisation) à partir des entrées lexicales devient nécessaire afin de réduire les temps de calcul.

Afin d'étendre le noyau du lexique, nous proposons l'approche suivante : nous nous appuyons sur un dictionnaire des synonymes afin de récupérer un ensemble de synonymes pour chaque entrée du lexique. Cet ensemble de synonymes reçoit les mêmes *tags* sémantiques que ceux des termes du lexique auxquels ils sont liés.

Nous utilisons pour cela le dictionnaire électronique des synonymes (DES) [16] développé par le Centre de Recherche Inter-langues sur la Signification en COntexte (CRISCO) de l'université de Caen Basse-Normandie. Ce dictionnaire a l'avantage de présenter les synonymes d'un mot sous formes de *cliques*. Une clique est un sous-ensemble de synonymes partageant un sens particulier. Un mot appartient donc à autant de cliques qu'il a de nuances de sens.

Les premiers résultats obtenus ne sont pas tout à fait satisfaisants car, si le dictionnaire de synonymes a permis d'étendre le noyau du lexique, il y a également ajouté un certain **bruit**. En effet, parmi les termes ajoutés, certains l'ont été parce qu'ils étaient synonymes de certaines entrées de notre lexique, mais prises *au sens figuré*. C'est donc le sens figuré qui était partagé, au lieu du sens propre.

Nous adoptons donc une deuxième approche : cette fois-ci, nous récupérons la liste de synonymes de chaque entrée du lexique depuis plusieurs sources, puis nous recoupons les différentes listes de synonymes et nous ne retenons finalement que les synonymes qui reviennent dans **au moins la moitié** des listes comparées.

Voici la liste des dictionnaires de synonymes que nous avons utilisés pour l'enrichissement du lexique :

- le dictionnaire électronique des synonymes de CRISCO ;
- le dictionnaire des synonymes de Reverso [17] ;
- le dictionnaire des synonymes [18].
- le dictionnaire des synonymes de Blue Painter [19].
- le dictionnaire de l'Institut des Sciences Cognitives (ISC) [20].
- le dictionnaire des synonymes de Sensegates [21].
- le dictionnaire des synonymes du Centre National de Ressources Textuelles et Lexicale (CNRTL) [22].
- le dictionnaire des synonymes de Synonymo [23].
- le dictionnaire en ligne Larousse [24].

Cette nouvelle approche a donné de meilleurs résultats puisqu'elle a permis à la fois d'éliminer les synonymes ayant un sens figuré et de consolider le choix des synonymes à retenir en recoupant plusieurs sources de données. Par exemple, pour le terme *voiture*, les synonymes *bagnole*, *automobile* ont été retenus alors que le synonyme *train* a été rejeté.

16. DES : *cf.* `http://www.crisco.unicaen.fr/des/`
17. *cf.* `http://dictionnaire.reverso.net/francais-synonymes/`
18. *cf.* `http://www.dictionnaire-synonymes.com/`
19. *cf.* `http://www.synonymes.com/`
20. *cf.* `http://dico.isc.cnrs.fr/dico/fr/chercher`
21. *cf.* `http://www.sensagent.com/`
22. *cf.* `http://www.cnrtl.fr/synonymie/`
23. *cf.* `http://synonymo.fr/`
24. *cf.* `http://www.larousse.fr/dictionnaires/francais`

3.2.3 Création de la grammaire

Une fois que nous avons mis en place le lexique étiqueté, nous définissons une grammaire non contextuelle métier. Cette dernière aura pour objectif de définir les concepts métiers pris en charge par l'interface de dialogue.

Si nous prenons le cas particulier d'une alerte de géolocalisation, la grammaire définit les différentes composantes de celle-ci qui se traduisent par des paramètres à demander à l'utilisateur lors du dialogue. C'est une approche de **spécification d'un concept général** par des **sous-concepts** qui, eux-mêmes, peuvent être spécifiés par la suite.

Pour cela, nous utilisons une syntaxe de type EBNF (*Extended Backus-Naur Form*) pour définir chaque élément de la grammaire métier. L'exemple suivant montre une grammaire simplifiée de la définition d'une alerte de géolocalisation :

Grammaire métier définissant une alerte.

```
ALERTE          = TYPE , MOBILE , LIEU , NOTIFICATION
MOBILE          = VEHICULE | PERSONNE
TYPE            = ENTREE_ZONE | SORTIE_ZONE | CORRIDOR
LIEU            = VILLE | ADRESSE | POI | ZOI
NOTIFICATION    = DESTINATAIRE , MESSAGE
DESTINATAIRE    = NUMERO_TEL | E_ADRESSE
MESSAGE         = MSG_TEXT
```

Cette grammaire définit une alerte comme étant composée (nécessairement) d'un type d'alerte, d'un mobile, d'un lieu et d'une notification. Ces composants sont donc primordiaux à la création d'une alerte de géolocalisation. La grammaire ainsi définie sert de scénario de dialogue puisqu'un utilisateur qui souhaite créer une alerte devra renseigner l'ensemble de ces paramètres avant d'y aboutir. Nous verrons plus en détail un peu plus loin le déroulement d'un dialogue de création d'alerte.

Les éléments terminaux de chaque grammaire sont directement liés à un (ou plusieurs) *tag*(*s*) sémantique(s) du lexique métier. Ceci permet à l'interface de dialogue de faire le lien entre les mots du lexique et la sémantique qui leur est attachée au niveau métier.

Par exemple, selon l'extrait précédent de la grammaire, un `MOBILE` peut être soit un `VEHICULE` soit une `PERSONNE`. Si nous reprenons le lexique défini précédemment, nous y retrouvons l'entrée "camion" taguée par le *tag* sémantique `VEHICULE`. L'interface de dialogue va faire le lien à chaque fois qu'elle rencontre le mot "camion" durant l'analyse d'une phrase de l'utilisateur avec la composante `MOBILE` de l'alerte *via* le *tag* sémantique `VEHICULE` lié donc à la composante `VEHICULE` de la grammaire métier.

3.3 Vers une approche d'agent intelligent de dialogue

L'interface de dialogue en langage naturel que nous proposons, adopte une vision d'*agent intelligent de dialogue* [Abchir et al., 2012b]. Le but de l'agent est d'interagir avec l'utilisateur afin d'expliciter ses choix et préférences puis de réaliser les tâches qui

en découlent comme : créer une alerte sur le Geohub, configurer un boîtier mobile, afficher les informations concernant un compte utilisateur, etc.).

3.3.1 Une grammaire pour piloter l'agent

L'agent utilise pour cela le lexique et la grammaire métier qu'il charge dynamiquement à son lancement. Le lexique est stocké dans une table de hachage tandis que la grammaire est chargée sous forme de *Tree-Adjoining Grammar* (TAG), une grammaire d'arbres adjoints, en français. Bien que la représentation sous forme d'arbre d'analyse soit restrictive au niveau de la représentation des connaissances, celle-ci offre l'avantage d'être plus facile à convertir en un traitement immédiat, par exemple, sous forme de requête sur une base de données.

La figure 3.1 montre un extrait de la représentation de la grammaire sous forme d'arbre TAG.

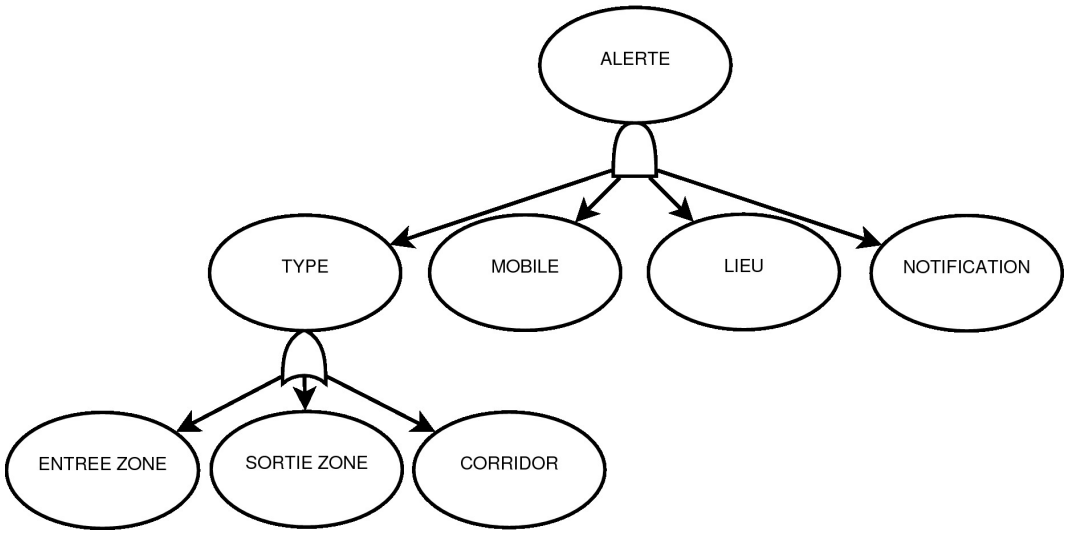


FIGURE 3.1 – Représentation en arbre TAG de la grammaire métier.

Le dialogue entre l'agent intelligent et l'utilisateur se passe de la façon suivante :

- l'agent de dialogue initie l'interaction en posant une question ouverte comme par exemple "Bonjour ! Que souhaitez-vous faire ?" ;
- il récupère ensuite la réponse de l'utilisateur qu'il transmet au parseur. Le parseur agit comme un pré-processeur qui prépare la phase d'analyse. Il étiquette chaque mot de la phrase-réponse à l'aide des *tags* sémantiques du lexique ;
- la phase suivante est celle de l'analyse de la phrase. En effet, la phrase étant taguée par des *tags* grammaticaux et sémantiques, l'analyse peut se faire selon ces deux axes. Cependant, nous choisissons de n'analyser la phrase que selon l'axe sémantique pour deux raisons :

(i) tout d'abord, il est important de veiller à réduire la complexité du traitement. En effet, notre intérêt principal étant la compréhension du sens de la phrase, nous concentrons les traitements et les efforts autour de cet axe,

(ii) ceci apporte une souplesse dans la formulation de réponses à l'utilisateur. Ainsi, une phrase agrammaticale peut être considérée comme correcte du moment que le sens est compréhensible. Nous rejoignons d'ailleurs, par cette approche, le principe de grammaire étendue de l'école de Harris [Harris, 1991] ;

– comme les nœuds de l'arbre TAG correspondent à des composantes (ou conditions) nécessaires à l'accomplissement d'une tâche donnée, l'agent de dialogue va chercher à valider tous les nœuds de l'arbre. Les nœuds fils d'un nœud de l'arbre sont liés entre eux par un opérateur ET, c'est-à-dire qu'ils sont tous nécessaires pour valider le nœud parent. Les feuilles d'un même nœud de l'arbre, quant à elles, sont liées par un opérateur OU, c'est-à-dire que seule l'une d'entre elles est suffisante. Par ailleurs, chaque fois qu'un paramètre est renseigné, le nœud y correspondant est marqué comme validé ;
– l'agent vérifie ensuite que l'ensemble des nœuds est validé (dans l'ordre dans lequel ils apparaissent dans l'arbre) :
  • si c'est effectivement le cas, alors l'ensemble des paramètres ont été récupérés et la tâche en question peut être exécutée,
  • sinon l'agent pose une question correspondant au paramètre manquant, parse et analyse la réponse puis recommence le processus de validation,
– le dialogue continue ainsi jusqu'à ce tous les paramètres soient renseignés par l'utilisateur.

3.3.2 Implémentation d'un prototype

Nous avons réalisé un prototype d'agent intelligent de dialogue en langage naturel spécialisé en géolocalisation sous forme d'une application Android. Afin de faciliter le dialogue, nous nous sommes appuyés sur la reconnaissance et la synthèse vocales du système Android afin d'assurer l'interaction avec l'utilisateur (transposition texte à voix et inversement).

L'application (appelée **DeveryDialog**) permet de réaliser plusieurs tâches telles que programmer une alerte, afficher la liste de mobiles d'un compte client, changer les fréquences de remontées de positions d'un boîtier GPS, etc.

Les figures 3.2 et 3.3 illustrent deux *scenarii* de dialogue de création d'alerte d'*entrée de zone*. Pour rappel, une alerte d'entrée de zone est déclenchée quand le mobile en question entre dans une zone donnée.

FIGURE 3.2 – Exemple de dialogue en langage naturel.

Dans la figure 3.2, nous illustrons un dialogue complet de création d'alerte. Après la phrase d'engagement de l'agent de dialogue, l'utilisateur, ici, répond par "je voudrais créer une alerte". Après annotation de la phrase, l'agent se réfère à la grammaire métier, et, puisqu'ici c'est la création d'alerte qui est demandée, c'est donc l'arbre TAG d'alerte qui sera utilisé. L'agent procède donc à la validation des éléments de l'arbre et étant donné qu'il s'agit, ici, du début du dialogue et qu'aucun paramètre n'a encore été renseigné, l'agent va essayer de demander le premier paramètre, à savoir le type d'alerte souhaitée.

L'agent pose donc une question pour inviter l'utilisateur à préciser le type d'alerte souhaitée. Il itère ensuite jusqu'à récupérer tous les paramètres et afficher un récapitulatif à l'utilisateur concernant l'alerte qui va être créée sur le Geohub en passant par l'API de géolocalisation de Deveryware.

Il est à noter que les réponses considérées comme "hors-sujet", c'est-à-dire ne répondant pas à la question, n'influent pas sur le déroulement du dialogue. En effet, après l'analyse d'une réponse non pertinente, la phase de validation pointera toujours sur le

même paramètre manquant. Ainsi, l'agent de dialogue demandera toujours de renseigner ce dernier.

Cependant, et pour des raisons évidentes de souplesse de l'interface, il existe un certain nombre de commandes vocales qui dérogent à cette règle et qui permettent d'annuler la création d'alerte, ou encore de la recommencer, par exemple.

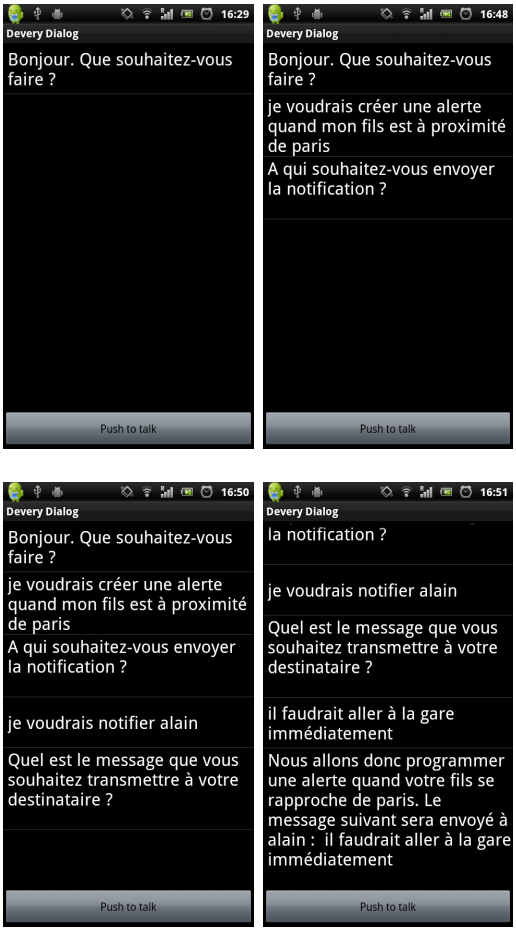


FIGURE 3.3 – Exemple de dialogue en langage naturel (long).

Dans la figure 3.3, nous illustrons le même scénario (création d'alerte) mais dans un cas de figure particulier. Cette fois-ci, l'utilisateur donne plusieurs paramètres en une seule réponse : "je voudrais créer une alerte quand mon fils est à proximité de Paris". L'utilisateur a renseigné à la fois le type d'alerte ("à proximité" étant lié par *tag* sémantique au type d'alerte "entrée de zone"), le mobile ainsi que la destination. L'agent, ayant récupéré tous ces paramètres et, après la phase de validation par l'arbre de décision, demande directement le destinataire de la notification puis le message à lui transmettre.

On constate que l'utilisation d'un arbre TAG permet une souplesse quant au déroulement du scénario de dialogue, ce qui rend ce dernier moins contraignant pour l'utilisateur.

Pour les besoins du prototype, les termes "fils" et "camion" ont été déclarés dans le lexique et tagués comme étant `MOBILE`. "Jacques" et "Alain" ont également été déclarés et tagués comme `DESTINATAIRE`. Dans sa version finale, l'agent de dialogue sera directement connecté aux profils de chaque utilisateur pour éviter la saisie manuelle de tous les mots inconnus ou les noms propres dans le lexique. Les profils regroupent, entre autres, les mobiles déclarés par l'utilisateur, son carnet d'adresse, la liste des adresses, points et zones d'intérêt qu'il a pré-enregistrés (ces derniers portent souvent des labels du type "maison", "travail", etc.).

3.4 Conclusion

Nous avons proposé une approche permettant aux experts du domaine et aux utilisateurs d'exprimer leurs besoins et préférences métiers à travers un dialogue en langage naturel. Pour ce faire, nous avons créé un corpus *ad hoc* à la géolocalisation et à Deveryware en particulier. Le lexique des termes retenus a été entièrement tagué, permettant ensuite la mise en place d'une grammaire métier. En première approche, un prototype d'agent intelligent de dialogue a ensuite été implémenté.

La deuxième partie non moins importante concerne la **modélisation sémantique** des termes du lexique, en s'appuyant sur les démarches traitant des imprécisions. On a vu dans la section 2.4 que, malgré la richesse des modèles existants, les outils actuels ne sont pas pleinement satisfaisants.

Chapitre 4

Modélisation des données fortement asymétriques

Sommaire

Nous avons présenté dans le chapitre 2.3 quelques unes des techniques phares dans le traitement des données vagues et imprécises ainsi que leur formalisme de modélisation et de calcul à partir de celles-ci.

Une approche reposant sur une modélisation à l'aide de 2-tuples peut répondre efficacement au problème de la sémantique des termes du lexique, en proposant non seulement le formalisme permettant d'obtenir une sémantique (floue) adéquate, mais également en apportant la souplesse de partitionnement nécessaire permettant de contextualiser cette dernière. Nous reviendrons sur ce point plus en détail dans le paragraphe 4.3.

Dans ce chapitre nous nous concentrons en particulier sur la question de la modélisation de données notamment quand celles-ci sont fortement déséquilibrées ou asymétriques sur leur intervalle de définition. Cette focalisation sur la phase de modélisation est motivée par l'importance que représente celle-ci dans le processus de conception d'un système à contrôleur flou fiable et fidèle au comportement souhaité par le concepteur (qui est souvent un expert d'un domaine particulier).

En l'occurrence, avec l'aide d'un expert de géolocalisation chez Deveryware, nous choisissons cinq termes (traduits en anglais) pour caractériser la distance de façon qualitative : *InTheCenter*, *VeryCloseTo*, *Near*, *Far*, *OutOfRoute* (pour "au centre de", "très proche de", "à côté de", "loin" et "hors de"). La distance, ici, représente la distance du mobile par rapport au centre du corridor. *InTheCenter* signifie que le mobile est tout à fait au centre du corridor, *VeryCloseTo* signifie qu'il est très proche du centre du corridor, *Near* signifie qu'il se rapproche des bords du corridor mais tout en restant dans ce dernier, *Far* signifie qu'il est sortie du corridor mais en reste proche et enfin *OutOfRoute* signifie qu'il s'est clairement éloigné du corridor, qu'il est hors du corridor.

Aussi, en utilisant sa propre expertise et les données des clients de Deveryware, notre expert a choisi, pour ces termes, des positions qui se trouvent être non uniformément distribuées sur l'axe. En effet, les notions de *Far* et *OutOfRoute* ne sont pas aussi semblables (deux à deux) que les notions de *InTheCenter* et *VeryCloseTo*.

4.1 Introduction

Comme nous l'avons vu dans les parties 2.1.1, 2.1.2 et 2.1.3, plusieurs techniques existent dans la littérature pour traiter des données qui peuvent être vagues, incertaines ou entachées d'imprécision. En particulier, chaque technique offre un formalisme de modélisation permettant d'attacher une sémantique (floue) à l'ensemble des termes choisis pour caractériser une notion donnée. Cette sémantique est souvent synonyme de création d'un partitionnement flou correspondant à cet ensemble de termes de départ.

Dans l'approche des sous-ensembles flous, la modélisation des données se fait par des fonctions d'appartenance. Celles-ci offrent donc une grande souplesse quant au choix de la représentation de chaque terme linguistique puisque ces fonctions peuvent prendre diverses formes. Mais comme évoqué précédemment en section 2.1.2, l'inconvénient majeur de cette approche est l'impossibilité d'exprimer la sortie d'un système à contrôleur flou par le biais des termes linguistiques utilisés pour modéliser la sortie du système. En effet, suite à l'agrégation des valeurs obtenues par l'évaluation des règles d'inférence, la fonction d'appartenance obtenue n'est pas exprimable directement par les termes linguistiques de la sortie. Ce qui oblige à effectuer une *défuzzification* qui engendre inévitablement une perte d'information.

L'approche symbolique prône, quant à elle, une vision radicalement différente en modélisant les données uniquement par des symboles. Elle utilise donc exclusivement des échelles et s'affranchit complètement de la deuxième dimension qui porte, dans la théorie des sous-ensembles flous, la valeur d'appartenance. L'appartenance est encodée directement sur l'échelle, comme on le voit avec l'expression-type "x est A" est τ_α-vraie. Ce sont les τ_α que l'échelle code.

A mi-chemin entre les deux approches, les 2-tuples linguistiques apportent une réponse aux inconvénients de l'approche des sous-ensembles flous par leur modélisation simplifiée des données sous la forme de couples (s_i, α), avec les s_i pouvant être vus comme des symboles. Effectivement, le modèle computationnel des 2-tuples linguistiques garantit de pouvoir exprimer les valeurs de sortie d'un système en fonction des termes

linguistiques décrivant cette sortie et ce, sans perte d'informations. Ce modèle de calcul permet également de réduire les calculs nécessaires lors du raisonnement flou en comparaison avec les opérations mises en œuvre par l'approche des sous-ensembles flous comme les calculs de projection de points et de centre de gravité d'un polygone par exemple.

L'approche des 2-tuples proportionnels, quant à elle, n'est qu'une formalisation alternative à celle des 2-tuples linguistiques mais qui se rapproche davantage de l'esprit des sous-ensembles flous en gardant une notion forte de fonction d'appartenance. Les auteurs de celle-ci se sont davantage focalisés sur les opérateurs de calcul plutôt que sur la modélisation elle-même des données.

Il est donc clair que les 2-tuples linguistiques se positionnent comme un modèle plutôt idéal dans notre situation, situation dans laquelle il nous faut représenter les données imprécises ou nécessitant des partitionnements flous non uniformément distribués sur l'axe, comme c'est le cas majoritairement dans notre domaine d'étude principal : la géolocalisation. Néanmoins, ces derniers présentent quelques inconvénients qui en contraignent à la fois l'utilisation mais aussi la sémantique obtenue par leur processus de partitionnement flou.

Tout d'abord, la notion de *densité* constitue une contrainte compliquée à gérer par l'utilisateur des 2-tuples linguistiques. En effet, comme le choix de la densité impacte le partitionnement flou généré, l'utilisateur doit avoir une fine connaissance des notions de granularité et des ensembles non équilibrés ou non symétriques.

De surcroît, l'algorithme de partitionnement flou utilisant les 2-tuples linguistiques (*cf.* section 2.1.2) oblige à avoir un nombre impair de termes dans l'ensemble de départ afin d'avoir un élément central S_C. Cette contrainte ne correspond pas toujours à la réalité souhaitée puisqu'un expert d'un domaine peut souhaiter un nombre pair de termes pour décrire une notion donnée.

Enfin, le partitionnement généré par cette méthode ne correspond pas toujours à ce qui est souhaité. En effet, durant le processus de partitionnement le choix des termes linguistiques, et, par extension, celui des niveaux de la hiérarchie linguistique, se fonde essentiellement sur le nombre de termes de l'ensemble de départ. Le partitionnement flou s'en trouve donc rigidifié et ne permet pas de prendre les préférences de l'utilisateur pour générer un partitionnement aussi fidèle que possible à celui souhaité.

Afin d'illustrer ce problème nous prenons comme exemple la modélisation de la mesure de l'alcoolémie selon des valeurs inspirées par la législation américaine. Nous considérons les valeurs suivantes :

- **0%** veut dire qu'il n'y a pas de présence d'alcool dans le sang,
- **0.05%** est la limite légale de quantité d'alcool dans le sang pour les conducteurs ayant moins de 21 ans,
- **0.065%** est une valeur intermédiaire (correspondant au stade de la désinhibition),
- **0.08%** est la limite légale pour les conducteurs de plus de 21 ans et enfin
- **3%** qui représente le taux d'alcoolémie à partir duquel il y a risque de mort.

Le partitionnement flou idéal que l'on souhaiterait obtenir devrait donc correspondre aux données et respecter la répartition des termes linguistiques sur l'axe (partitionnement

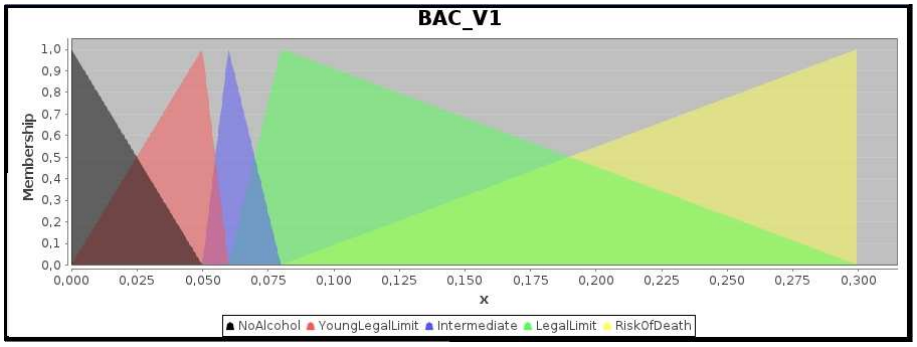


FIGURE 4.1 – Partitionnement flou idéal pour les taux d'alcoolémie aux USA.

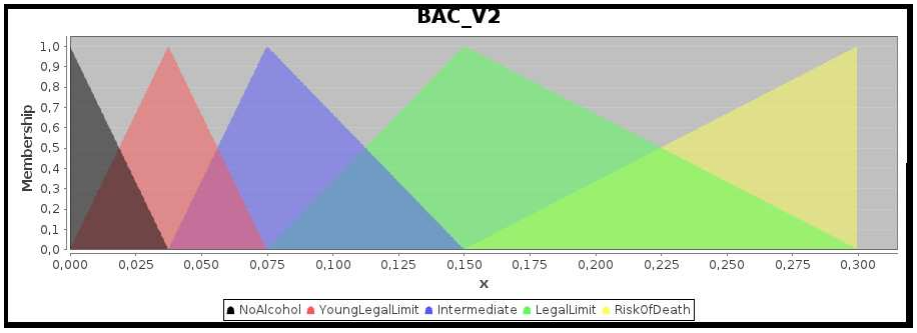


FIGURE 4.2 – Partitionnement flou pour les taux d'alcoolémie aux USA *via* les 2-tuples linguistiques.

illustré par figure 4.1 par des fonctions d'appartenance triangulaire sans sémantique particulière).

En utilisant les 2-tuples linguistiques, nous posons (les termes utilisés ont été traduits en anglais) :
$S = \{NoAlcohol, YoungLegalLimit, Intermediate, LegalLimit, RiskOfDeath\}$ ainsi que la configuration suivante : {(3,extreme),1,(1,extreme)}, ce qui nous donne : $S_L = \{NoAlcohol, YougLegalLimit, Intermediate\}$, $S_C = \{LegalLimit\}$ et $S_R = \{RiskOfDeath\}$. Le résultat obtenu par la méthode de Herrera et Martínez vue en section 2.1.2 est illustré par la figure 4.2.

Comme nous pouvons le constater en comparant les deux figures 4.1 et 4.2, le résultat obtenu n'est pas celui qui était attendu, et pour cause, l'algorithme de génération de partition floue ne prend pas en considération les valeurs souhaitées pour chaque terme. De plus, un autre problème apparait dans cet exemple, l'algorithme de partitionnement de permet pas de gérer les données qui sont fortement déséquilibrées sur l'axe, *i.e.* le cas où les distances entres les termes sont très grandes comme c'est le cas entre les deux termes *LegalLimit* et *RiskOfDeath*. En effet, la méthode permet l'utilisation des niveaux dans la hiérarchie (représentant la granularité ou la précision) mais sans réelle possibilité de choisir le bon niveau.

Suite aux précédents constats, nous envisageons une méthode de modélisation de données (fortement) asymétriques ou déséquilibrées sur leur axe permettant à la fois de tirer profit des avantages des 2-tuples linguistiques et de garantir un partitionnement flou le plus proche possible de celui souhaité.

4.2 Le modèle proposé

Comme nous l'avons exposé, les 2-tuples linguistiques présentent des inconvénients qui en compromettent la généralisation à tout problème traitant des données floues et spécialement quand celles-ci sont fortement déséquilibrées sur l'axe. Néanmoins, nous souhaitons garder le formalisme apporté par les 2-tuples linguistiques pour leur modèle de calcul optimisé et sans perte d'information ainsi que pour le fait qu'ils conservent les termes linguistiques de départ dans l'expression de la solution finale.

Nous proposons dans ce qui suit une méthode de partitionnement flou donnant de meilleurs résultats y compris et en particulier quand les données à modéliser sont fortement déséquilibrées sur leur axe.

4.2.1 La sémantique au cœur du partitionnement

Afin de proposer une méthode de partitionnement flou efficace fondée sur les 2-tuples linguistiques, nous allons revenir sur leur problématique principale qui est la non prise en compte durant le processus de partitionnement des préférences de l'utilisateur (ou l'expert) quant aux positions souhaitées des termes de départ sur l'axe. Ce qui crée naturellement un décalage entre la sémantique attribuée aux termes par l'algorithme de partitionnement et celle attendue. Idéalement, les positions des termes souhaitées par l'utilisateur devraient être l'élément principal dans le choix de la sémantique à leur attribuer afin que le partitionnement flou résultant reflète au mieux la réalité des données que l'on souhaite modéliser.

Pour cela, nous modélisons les données sous la forme d'un couple (s, v) où s est un terme et v sa position sur l'axe $(\mathit{NoAlcohol}, 0.0)$ par exemple. Nous appellerons ces couples, des **couples sémantiques** [Abchir et Truck, 2011]. Le mot *sémantique* est utilisé en référence au langage naturel, car c'est bien la sémantique qui doit guider la modélisation des termes.

Ainsi, l'ensemble de termes de départ est composé d'un ensemble de couples sémantiques $(\mathsf{s}_i, \mathsf{v}_i)$ comme le montre la définition 31.

Définition 31. *Soit $\mathcal{S}$ un ensemble ordonné de termes linguistiques asymétriques et U l'univers de discours numérique sur lequel les termes sont projetés. Chaque valeur linguistique est modélisée par un couple sémantique unique $(\mathsf{s}, \mathsf{v}) \in \mathcal{S} \times U$. Nous notons d_i la distance entre les deux termes s_i et s_{i+1} avec $d_i = \mathsf{v}_{i+1} - \mathsf{v}_i$.*

En vue d'attribuer une sémantique à l'ensemble des termes de $\mathcal{S}$ nous nous appuyons également sur les hiérarchies linguistiques (cf. définitions 23 et 24). Ces dernières sont composées de plusieurs niveaux, chacun défini par un ensemble de termes linguistiques

ordonnés et uniformément distribués sur l'axe. Nous considérons, à l'image du modèle de Herrera et Martínez, des ensembles de termes S où chaque terme est associé à une translation symbolique notée α telle que $\alpha \in [-0.5, 0.5[$. Les définitions 15 et 16 restent donc valables ici.

Définition 32. *Soit donc $S = \{s_0, \ldots, s_p\}$ un ensemble de termes linguistiques asymétriques et soit (s_i, α) un 2-tuple linguistique, au sens de Herrera et Martínez. Pour supporter l'asymétrie, S est étendu à plusieurs ensembles de termes linguistiques uniformément distribués, chacun étant noté $S^{n(t)} = \{s_0^{n(t)}, \ldots, s_{n(t)-1}^{n(t)}\}$ (obtenus grâce aux définitions 23 et 24) et défini au niveau t d'une hiérarchie linguistique LH à $n(t)$ étiquettes. Il n'existe qu'une et une seule façon de passer de $\mathcal{S}$ (définition 31) à S, selon l'algorithme 1 (voir plus loin).*

De plus, nous définissons le grain d'un niveau de la hiérarchie comme étant la distance entre deux termes $(s_i^{n(t)}, \alpha)$ de celui-ci.

Proposition 1. *Soient $l(t, n(t))$ un niveau d'une hiérarchie linguistique LH et t un niveau dans LH. Le grain g_t d'un niveau $l(t, n(t))$ est la distance entre deux de ces termes successifs. Ainsi, g_t est obtenu comme suit :*

$$g_t = g_{l(t,n(t))} = \frac{1}{n(t)-1}$$

Preuve : le grain g est défini comme la distance entre $(s_i^{n(t)}, \alpha)$ et $(s_{i+1}^{n(t)}, \alpha)$, *i.e.* entre les deux noyaux des fonctions d'appartenance qui leur sont associées (car α est égale à 0). Pour une hiérarchie normalisée (définie sur $[0, 1]$), le grain est obtenu en utilisant la fonction Δ^{-1} (cf. définition 16) où $\Delta^{-1}(s_i^{n(t)}, \alpha) = \frac{i}{n(t)-1}$ quand $\alpha = 0$. Nous obtenons ainsi $g_t = \frac{i+1}{n(t)-1} - \frac{i}{n(t)-1} = \frac{1}{n(t)-1}$.

Notre volonté étant de proposer des dialogues homme-machine et, donc, d'utiliser la langue naturelle, nous utiliserons dans ce qui suit les distances entre les couples sémantiques $(\mathsf{s}_i, \mathsf{v}_i)$ et le grain des niveaux de la hiérarchie linguistique afin de déterminer la *sémantique* associée à chaque couple. On le comprend, la notion de distance est centrale : en effet, les termes étant placés sur des échelles, les écarts en termes de valeurs se traduisent en écarts en termes de sens qu'on leur rattache. La sémantique d'un couple sera représentée par un 2-tuple que nous nommons **2-tuple sémantique**.

4.2.2 Le partitionnement flou

La modélisation des couples sémantiques s'inspire directement de celle des 2-tuples linguistiques, ce qui nous permet d'en garder tous les avantages.
Cependant, la sémantique associée à chaque couple est obtenue de manière différente et ce, bien qu'elle garde le même principe fondamental : associer à chaque couple sémantique $(\mathsf{s}_i, \mathsf{v}_i)$ un ou plusieurs termes linguistiques $(s_j^{n(t)}, \alpha_j)$ d'une hiérarchie linguistique LH (ou hiérarchie linguistique étendue ELH) sous la forme d'un 2-tuple sémantique. En effet, nous utilisons à la fois la *position* des couples sémantiques, la *distance* entre deux couples successifs ainsi que le *grain* de chaque niveau de la hiérarchie linguistique pour constituer

le partitionnement flou leur correspondant [Abchir et Truck, 2013]. L'attribution de la sémantique aux couples de l'ensemble de départ $\mathcal{S}$ se déroule comme suit :
Soit deux couples sémantiques successifs $(\mathsf{s}_i, \mathsf{v}_i)$ et $(\mathsf{s}_{i+1}, \mathsf{v}_{i+1})$. La première étape consiste à choisir le meilleur niveau de la hiérarchie linguistique avec lequel ils devraient être représentés. Idéalement, le niveau choisi devrait avoir une granularité assez élevée pour représenter de la manière la plus précise possible les deux couples en question. Une granularité insuffisante engendrerait un partitionnement flou dont la sémantique serait assez éloignée de celle souhaitée initialement.

Conditions requises.

Afin que le niveau de la hiérarchie linguistique corresponde au mieux aux couples $(\mathsf{s}_i, \mathsf{v}_i)$ et $(\mathsf{s}_{i+1}, \mathsf{v}_{i+1})$, nous nous appuyons sur le grain g des niveaux de la hiérarchie linguistique. Ainsi, le niveau retenu est celui dont le grain se rapproche le plus de la distance d_i entre les deux couples en question.
Il est important que la granularité du niveau t choisi soit suffisante, *i.e.* si nous avons un ensemble de départ $\mathcal{S}$ dont les termes sont uniformément distribués sur l'axe avec une même distance d_i, alors le niveau t doit avoir au moins autant de termes $(s_j^{n(t)}, \alpha_j)$ que de couples $(\mathsf{s}_i, \mathsf{v}_i)$. Ceci permet à tous les couples d'être représentés par (au moins) un terme linguistique. La condition suivante doit donc être vérifiée : $g_i < d_i$.

Construction.

La recherche du niveau idéal peut aboutir à deux issues :
- soit la condition précédente est vérifiée par un niveau t de la hiérarchie linguistique et, dans ce cas, il faut passer à l'étape suivante ;
- soit aucun niveau de la hiérarchie linguistique ne satisfait cette condition et dans ce cas, nous ajoutons à la hiérarchie autant de niveaux que nécessaire jusqu'à ce que la condition soit satisfaite.

Une fois le niveau idéal t identifié, il nous faut choisir quels termes linguistiques conviennent le mieux pour représenter les deux couples sémantiques $(\mathsf{s}_i, \mathsf{v}_i)$ et $(\mathsf{s}_{i+1}, \mathsf{v}_{i+1})$. Pour cela, nous nous référons à leurs positions respectives et nous choisissons le terme linguistique $(s_j^{n(t)}, 0)$ (initialement $\alpha = 0$ pour tous les termes linguistiques d'un niveau de la hiérarchie) dont la position sur l'axe est la plus proche possible de celle du couple sémantique $(\mathsf{s}_i, \mathsf{v}_i)$.
En minimisant la distance $D = |\Delta((s_j^{n(t)}, 0)) - \mathsf{v}_i|$, le terme $(s_j^{n(t)}, 0)$ a la sémantique la plus appropriée pour représenter le couple $(\mathsf{s}_i, \mathsf{v}_i)$.

Quand la distance $D \neq 0$, ceci veut dire que le terme linguistique $(s_j^{n(t)}, 0)$ ne correspond pas parfaitement au couple sémantique $(\mathsf{s}_i, \mathsf{v}_i)$. Donc représenter $(\mathsf{s}_i, \mathsf{v}_i)$ par $(s_j^{n(t)}, 0)$ directement entraînerait une perte d'information quantifiée par D. Pour remédier à cela, nous utilisons la translation symbolique α_j pour compenser le manque à gagner (ou manque de précision).
Ainsi, $\alpha_j = \pm D$ selon si $\Delta((s_j^{n(t)}, 0)) - \mathsf{v}_i$ est strictement positif ou strictement négatif,

ce qui revient à faire correspondre le noyau de la fonction d'appartenance représentée par $(s_j^{n(t)}, 0)$ avec la position v_i.

Comme nous traitons les couples sémantiques par paire $[(\mathsf{s}_i, \mathsf{v}_i), (\mathsf{s}_{i+1}, \mathsf{v}_{i+1})]$, nous ne représentons, à chaque itération, que la moitié descendante du premier notée $(\underline{\mathsf{s}_i, \mathsf{v}_i})$ et la moitié ascendante du deuxième notée $(\overline{\mathsf{s}_{i+1}, \mathsf{v}_{i+1}})$.
Donc, une fois calculé le terme idéal, nous en associons la moitié descendante à celle du premier 2-tuple : $(\underline{\mathsf{s}_i, \mathsf{v}_i}) \longleftarrow (\underline{s_j^{n(t)}, \alpha_j})$.
Naturellement, nous sélectionnons le terme linguistique suivant dans le même niveau de la hiérarchie linguistique, $(s_{j+1}^{n(t)}, 0)$, pour représenter le deuxième couple. Ici également, nous calculons la translation symbolique nécessaire à faire correspondre le noyau de la fonction d'appartenance de $(s_{j+1}^{n(t)}, 0)$ avec la position v_{i+1}, puis nous en associons la moitié ascendante dans ce cas : $(\overline{\mathsf{s}_{i+1}, \mathsf{v}_{i+1}}) \longleftarrow (\overline{s_{j+1}^{n(t)}, \alpha_{j+1}})$.

Remarque. Par commodité d'écriture, nous plaçons la barre horizontale en dessous (ou au-dessus selon la moitié considérée — descendante ou ascendante) d'un 2-tuple ou d'un couple dans son intégralité, mais il est équivalent de placer cette barre horizontale sous (ou sur) le terme s_i seulement ou bien sous (ou sur) la translation α seulement. Ainsi,
$(\underline{s_i^{n(t)}, \alpha_i}) = (\underline{s_i^{n(t)}}, \underline{\alpha_i})$ et $(\overline{s_i^{n(t)}, \alpha_i}) = (\overline{s_i^{n(t)}}, \overline{\alpha_i})$.
Aussi, les 2-tuples sémantiques peuvent aussi s'écrire d'une façon qui met plus en valeur l'association de deux demi 2-tuples comme suit : $[(\overline{s_j^{n(t)}, \alpha_j}), (\underline{s_{j+1}^{n(t)}, \alpha_{j+1}})] = (s_j^{n(t)} | s_{j+1}^{n(t)}, \alpha_j | \alpha_{j+1})$.

A la fin de cette itération du processus de partitionnement flou, nous attribuons une sémantique aux parties respectivement descendante et ascendante de la paire de couples que nous traitons. Nous passons ensuite à la paire $[(\mathsf{s}_{i+1}, \mathsf{v}_{i+1}), (\mathsf{s}_{i+2}, \mathsf{v}_{i+2})]$ et nous recommençons le processus.
A la fin de l'ensemble du processus, chaque couple $(\mathsf{s}_{i+1}, \mathsf{v}_{i+1})$ puise sa sémantique dans le 2-tuple sémantique $[(\overline{s_j^{n(t)}, \alpha_j}), (\underline{s_{j+1}^{n(t)}, \alpha_{j+1}})]$ qui lui est associé. Il est à noter que le premier et le dernier couples sémantiques de l'ensemble de départ $\mathcal{S}$ ne sont associés qu'à une moitié de 2-tuple sémantique (une moitié descendante pour le premier et une moitié ascendante pour le dernier).

L'algorithme 1 résume l'ensemble du processus de partitionnement flou pour un ensemble de couples sémantiques de départ $\mathcal{S}$. Comme le montrent les précédentes explications, nous ne posons aucune condition quant à la parité du nombre de termes dans $\mathcal{S}$.

Il est à noter que, comme la hiérarchie linguistique est normalisée par défaut
$\forall j \in [0, n(t) - 1], \Delta((s_j^{n(t)}, \alpha_j)) \in [0, 1]$, celle-ci est mise à l'échelle de l'axe des 2-tuples sémantiques, *i.e.* $\forall j \in [0, n(t) - 1], \Delta((s_j^{n(t)}, \alpha_j) \in [\mathsf{v}_{min}, \mathsf{v}_{max}])$ où v_{min} et v_{max} sont respectivement les positions minimale et maximale de l'ensemble de départ $\mathcal{S}$.

Nous notons aussi que comme la représentation des moitiés ascendante et descendante d'un même 2-tuple sémantique peut appartenir à deux niveaux différents de la hiérarchie linguistique, les 2-tuples sémantiques ont les mêmes caractéristiques que les

Algorithme 1 Algorithme de partitionnement

Require: $\langle(\mathsf{s}_0, \mathsf{v}_0), \ldots, (\mathsf{s}_{p-1}, \mathsf{v}_{p-1})\rangle$ sont p couples sémantiques de $\mathcal{S} \times U$;
et $t, t_0, \ldots, t_{p-1}$ sont les niveaux de la hiérarchie linguistique

1: mise à l'échelle de la hiérarchie sur $[\mathsf{v}_{\min}, \mathsf{v}_{\max}]$, où $\mathsf{v}_{\max}$ et $\mathsf{v}_{\min}$ sont le maximum et le minimum des valeurs v
2: nous pré-calculons η niveaux de la hiérarchie ainsi que leur grain g (avec $\eta \geq 6$)
3: **for** $k = 0$ to $p - 1$ **do**
4: $\quad d_k \leftarrow \mathsf{v}_{k+1} - \mathsf{v}_k$
5: $\quad$ **for** $t = \eta$ to 1 **do**
6: $\quad\quad$ **if** $g_{l(t,n(t))} \leq d_k$ **then**
7: $\quad\quad\quad t_k \leftarrow t$
8: $\quad\quad$ **end if**
9: $\quad$ **end for**
10: $\quad tmp = \mathsf{v}_{max}$
11: $\quad$ **for** $i = 0$ to $n(t_k) - 1$ **do**
12: $\quad\quad$ **if** $tmp > |\Delta^{-1}(s_i^{n(t_k)}, 0) - \mathsf{v}_k|$ **then**
13: $\quad\quad\quad tmp = |\Delta^{-1}(s_i^{n(t_k)}, 0) - \mathsf{v}_k|$
14: $\quad\quad\quad j \leftarrow i$
15: $\quad\quad$ **end if**
16: $\quad$ **end for**
17: $\quad \underline{s_k^{n(t_k)}} \leftarrow \underline{s_j^{n(t_k)}}$; $\overline{s_{k+1}^{n(t_k)}} \leftarrow \overline{s_{j+1}^{n(t_k)}}$
18: $\quad$ selon le niveau de la hiérarchie, $\underline{\alpha_k} = \mathsf{v}_k - \Delta^{-1}(s_j^{n(t_k)}, 0)$ ou
$$\overline{\alpha_{k+1}} = \mathsf{v}_{k+1} + \Delta^{-1}(s_{j+1}^{n(t_k)}, 0)$$
19: **end for**
20: **return** l'ensemble $\{(\underline{s_0^{n(t_0)}}, \underline{\alpha_0}), (\overline{s_1^{n(t_0)}}, \overline{\alpha_1}), (\underline{s_1^{n(t_1)}}, \underline{\alpha_1}), \ldots, (\underline{s_{p-2}^{n(t_{p-2})}}, \underline{\alpha_{p-2}}), (\overline{s_{p-1}^{n(t_{p-2})}}, \overline{\alpha_{p-1}})\}$

2-tuples linguistiques intermédiaires (ou pivots) que nous avons précédemment présentés en section 2.1.2. Ils constituent l'union de deux demi-termes linguistiques (ou encore deux demi 2-tuples) et peuvent avoir deux représentations ayant la même sémantique (les représentations dans les deux niveaux concernés de la hiérarchie linguistique).

Remarque : Pour l'algorithme de partitionnement (*cf.* algorithme 1), nous nous fondons sur les hiérarchies linguistiques de type LH et non pas ELH. En effet, bien que ces dernières permettent l'ajout de niveaux sans la contrainte de multiplication par deux de la granularité (cf. définition 24), le (faible) gain de souplesse ne justifie pas la complexité induite à l'ajout d'un niveau de forte granularité.

De plus, contrairement aux 2-tuples linguistiques, notre méthode de partitionnement flou pour 2-tuples sémantiques utilise une hiérarchie dynamique au lieu d'une hiérarchie statique. Ceci se traduit par l'ajout, en cas de besoin, de niveaux, suivant la règle 24 tout en garantissant la cohérence de la hiérarchie linguistique et, par la même occasion, le partitionnement flou obtenu.

Dans le modèle fondé sur les 2-tuples linguistiques, Herrera et Martínez garantissent plusieurs propriétés comme, par exemple, celles de Ruspini, notamment le fait qu'en

tout point x de l'axe, la somme des valeurs d'appartenance de x à chaque sous-ensemble flou défini sur l'axe doit toujours faire exactement 1 ($\sum_i \mu_{s_i^{n(t_i)}}(x) = 1$) [Ruspini, 1969]. Comme notre but est d'obtenir un partitionnement aussi fidèle que possible aux choix de l'utilisateur, ces dernières conditions peuvent ne pas être remplies à cause de la translation symbolique que nous appliquons aux termes linguistiques $(s_i^{n(t)}, 0)$.

Nous assumons ce choix car d'une part, nous exploitons pleinement les translations symboliques, et d'autre part, notre but premier est de garantir une couverture minimale de l'univers de discours (*cf.* proposition 2).

Proposition 2. *Les termes linguistiques $(s_i^{n(t)}, \alpha_i)$ choisis pour représenter les 2-tuples sémantiques ont des fonctions d'appartenance triangulaires et offrent une couverture minimale de l'univers de discours U.*

Preuve : La propriété de couverture minimale est garantie par le fait que la distance entre chacune des paires $[(\underline{s_k^{n(t)}, \alpha_k}), (\overline{s_{k+1}^{n(t)}, \alpha_{k+1}})]$ est strictement supérieure au double du grain du niveau t. En effet, nous avons utilisé d_k pour choisir le niveau idéal t pour cette paire. Et comme la valeur t est inversement proportionnelle au grain $g_{l(t,n(t))}$ alors nous avons :

$$g_{l(t,n(t))} \leq d_k < g_{l(t-1,n(t-1))} \tag{4.1}$$

Durant le processus de partitionnement flou, nous utilisons la translation symbolique sur les termes linguistiques choisis et la distance entre $(\underline{s_k^{n(t)}, \alpha_k})$ et $(\overline{s_{k+1}^{n(t)}, \alpha_{k+1}})$ est égale à d_k. Sachant que le grain d'un niveau est deux fois supérieur au grain du niveau suivant (dû à la nature des hiérarchies linguistiques LH) et vu l'équation 4.1, nous obtenons :

$$d_k < 2g_{l(t,n(t))} \tag{4.2}$$

ce qui veut dire que pour toutes les valeurs de l'univers U, le partitionnement obtenu a une valeur d'appartenance minimale ε strictement supérieure à 0. En prenant $\mu_{s_i^{n(t)}}$ la fonction d'appartenance associée à un terme linguistique $(s_i^{n(t)}, \alpha_i)$, la propriété de couverture minimale est notée :

$$\forall u \in U, \qquad \mu_{s_0^{n(t_0)}}(u) \vee \cdots \vee \mu_{s_i^{n(t_i)}}(u) \vee \cdots \vee \mu_{s_{p-1}^{n(t_{p-1})}}(u) \geq \varepsilon > 0. \tag{4.3}$$

4.2.3 Modèle de calcul des 2-tuples sémantiques

Le modèle de représentation des données par les 2-tuples sémantiques étant inspiré de celui des 2-tuples linguistiques, nous nous fondons également sur leur modèle de calcul afin d'assurer une inter-compatibilité entre les deux modèles.

Les fonctions Δ, Δ^{-1}, $\mathcal{LH}$ et $\mathcal{LH}^{-1}$ que nous utilisons pour les calculs des 2-tuples sémantiques sont directement dérivées de celles définies par Herrera et Martínez.
Pour rappel, les fonctions Δ et Δ^{-1} permettent d'obtenir le terme linguistique $(s_j^{n(t)}, \alpha_j)$ d'une hiérarchie linguistique LH correspondant à une valeur β sur l'axe de valeurs et

inversement, tandis que $\mathcal{LH}$ et $\mathcal{LH}^{-1}$ (voir [Herrera et al., 2008]) permettent d'obtenir le 2-tuple (linguistique (s_i, α_i) ou sémantique dans notre cas) correspondant à un terme linguistique d'une hiérarchie linguistique et inversement.

Dans ce qui suit, nous allons voir en détail des opérateurs essentiels à tout modèle de calcul flou : la question de l'agrégation, au sens général du terme et l'addition,en particulier.

Agrégation, exemple avec la moyenne, et cas général

Afin d'agréger des couples sémantiques $(\mathsf{s}_i, \mathsf{v}_i)$, nous rappelons que, suite au processus de partitionnement flou, chaque couple est associé à un 2-tuple sémantique (ou plutôt à deux demi 2-tuples sémantiques). Ainsi, agréger des couples sémantiques revient à agréger les 2-tuples sémantiques qui les représentent puis représenter le résultat dans l'ensemble de départ des couples sémantiques $\mathcal{S}$.

Nous pouvons résumer le processus d'agrégation (par calcul de moyenne) des couples sémantiques ainsi :

1. calculer la valeur numérique du 2-tuple sémantique associé à chaque couple sémantique à l'aide de la fonction Δ^{-1} ;
2. appliquer l'opérateur d'agrégation aux valeurs numériques des 2-tuples sémantiques. Nous nommons β le résultat de cette opération ;
3. appliquer la fonction Δ à β afin d'obtenir le terme linguistique $(s_j^{n(t)}, \alpha_j)$ de la hiérarchie linguistique lui correspondant ;
4. transformer $(s_j^{n(t)}, \alpha_j)$ à l'aide de la fonction $\mathcal{LH}^{-1}$ afin d'obtenir un 2-tuple sémantique (s_k, α_k) exprimé dans l'espace de départ.

Le résultat (s_k, α_k) ainsi obtenu est un 2-tuple sémantique projeté dans l'ensemble de départ et ce sans perte d'information (grâce à la translation symbolique).

Pour illustrer le processus d'agrégation, nous représentons les taux d'alcoolémie précédemment introduits par les couples sémantiques suivants :
$\mathcal{S} = \{(\mathit{NoAlcohol}, 0.0)(\mathit{YoungLegalLimit}, 0.05)(\mathit{Intermediate}, 0.06)(\mathit{LegalLimit}, 0.08)$
$(\mathit{RiskOfDeath}, 0.3)\}$.

Nous allons appliquer une agrégation de type moyenne arithmétique aux deux couples $(\mathit{YoungLegalLimit}, 0.05)$ et $(\mathit{LegalLimit}, 0.08)$.

En utilisant notre algorithme de partitionnement, $(\mathit{YoungLegalLimit}, 0.05)$ est associé au 2-tuple sémantique $[\overline{(s_5^{33}, 0.003)}, \underline{(s_1^9, 0.125)}]$ alors que $(\mathit{LegalLimit}, 0.08)$ est associé à $[\overline{(s_1^3, -0.07)}, \underline{(s_4^{17}, 0.005)}]$.

Tout d'abord nous appliquons la fonction Δ^{-1} aux deux 2-tuples sémantiques $[\overline{(s_5^{33}, 0.003)}, \underline{(s_1^9, 0.125)}]$ et $[\overline{(s_1^3, -0.07)}, \underline{(s_4^{17}, 0.005)}]$. Nous obtenons respectivement 0.05 et 0.08.

Nous appliquons ensuite la moyenne arithmétique aux deux valeurs obtenues. Comme ces dernières sont définies sur une échelle absolue (la hiérarchie linguistique est mise à

l'échelle au début du processus du partitionnement), aucune transformation n'est nécessaire. Nous obtenons comme résultat de l'agrégation : $\beta = 0.065$.

En troisième étape, nous cherchons à exprimer β par son terme linguistique correspondant dans la hiérarchie LH. Le niveau choisi pour la représentation est le niveau ayant le grain le plus fin parmi ceux utilisés pour représenter les deux 2-tuples sémantiques. Dans notre cas, il s'agit de $l(5, 33)$. Nous appliquons ensuite la fonction Δ à β pour obtenir le résultat suivant : $\Delta(\beta) = (s_7^{33}, -0.001)$.

Finalement, nous appliquons la fonction $\mathcal{LH}^{-1}$ afin d'obtenir un couple sémantique exprimé dans notre ensemble de départ $\mathcal{S}$. Le résultat final obtenu est : $\mathcal{LH}^{-1}((s_7^{33}, -0.001)) = (\textit{Intermediate}, 0.005)$.
La forme (*Intermediate*, 0.005) est une commodité d'écriture qui permet une meilleure compréhension du résultat obtenu. La solution finale est donc interprétée de la façon suivante : le résultat de l'agrégation est le terme *Intermediate*, qui est issu du couple sémantique $(\textit{Intermediate}, 0.06)$, à qui nous appliquons une translation symbolique de 0.005. Ceci revient à appliquer la translation symbolique au 2-tuple sémantique auquel est associé le couple sémantique $(\textit{Intermediate}, 0.06)$.

Ce type d'agrégation peut être utile pour des statistiques incluant le calcul de moyennes de taux d'alcoolémie dans divers pays par exemple car les législations diffèrent et, par conséquent, les seuils des taux légaux également. Il peut bien sûr être utile également dans des cas, plus généraux, de décision collective, où il est nécessaire de satisfaire le maximum d'exigences.

Remarque : Grâce à l'échelle absolue que nous avons adoptée pour notre méthode de partitionnement, **tout autre type d'opérateur** peut être adapté et utilisé pour les 2-tuples sémantiques car celui-ci s'applique directement aux positions v.

Addition, cas particulier

Nous procédons, pour l'opération d'addition, comme pour les autres opérations, grâce à l'échelle absolue sur laquelle sont définis les 2-tuples sémantiques et la hiérarchie linguistique. Nous appliquons ainsi l'addition aux valeurs numériques des 2-tuples sémantiques obtenues par le biais de la fonction $\Delta - 1$, puis nous appliquons successivement les fonctions Δ et $\mathcal{LH}^{-1}$ pour obtenir le résultat final. Cette opération d'addition est notée $\oplus$.

Par exemple, considérons l'addition des deux 2-tuples sémantiques $[(\overline{s_5^{33}, 0.003}), (\underline{s_1^9, 0.125})]$ et $[(\overline{s_1^3, -0.07}), (\underline{s_4^{17}, 0.005})]$ associés respectivement aux deux couples sémantiques $(\overline{\textit{YoungLegalLimit}, 0.05})$ et $(\underline{\textit{LegalLimit}, 0.08})$. Nous notons cette addition $[(\overline{s_5^{33}, 0.003}), (\underline{s_1^9, 0.125})] \oplus [(\overline{s_1^3, -0.07}), (\underline{s_4^{17}, 0.005})]$ dont le déroulement est le suivant :

- nous calculons les valeurs numériques correspondant aux deux 2-tuples sémantiques. Nous obtenons respectivement 0.05 et 0.08 ;
- nous additionnons les valeurs 0.05 et 0.08 pour obtenir $\beta = 0.13$;
- nous appliquons Δ pour obtenir la représentation de β dans LH : $\Delta(0.13) = (s_{14}^{33}, -0.001)$;

– finalement, nous utilisons $\mathcal{LH}^{-1}$ pour obtenir le résultat final : $\mathcal{LH}^{-1}((s_{14}^{33}, -0.001)) = (\textit{LegalLimit}, 0.05)$.

Nous obtenons un 2-tuple sémantique exprimé dans l'ensemble de départ : (*LegalLimit*, 0.05). Nous avons adopté ici également la même commodité d'écriture, comme dans l'exemple précédent.

Propriétés de l'addition pour 2-tuples sémantiques

L'opérateur d'addition $\oplus$ peut être considéré comme une extension (selon le principe d'extension de Lotfi Zadeh) de l'addition classique aux 2-tuples sémantiques. Il est donc **associatif**, **commutatif**, possède un **élément neutre** $e = (s_0, 0.0)$ où s_0 est le terme correspondant à la position 0 (c'est-à-dire que $\Delta^{-1}((s_0, 0.0)) = 0$) et un **élément symétrique** (s'_i, α'_i) tel que $(s'_i, \alpha'_i) \oplus (s_i, \alpha_i) = e$.

4.3 Sémantique floue

Nous avons présenté dans le chapitre 3 une approche de traitement du langage naturel pour la création d'agents intelligents de dialogue dans un domaine clos. Cette approche s'appuie notamment sur l'établissement d'un lexique tagué ainsi que d'une grammaire métier faisant office de scénario de dialogue. L'analyse des phrases de l'utilisateur permet la compréhension et la récupération des demandes et préférences de celui-ci aboutissant ainsi à diverses actions définies par le concepteur de l'agent.

Néanmoins, il reste un point critique que nous n'avons pas abordé. Lors de ces dialogues en langage naturel, les utilisateurs emploient leurs propres mots pour décrire leurs besoins métiers. Et comme nous l'avons déjà relevé, le langage naturel comporte plusieurs termes ambigus, vagues et, par nature, imprécis. Ces termes sont le plus souvent employés pour exprimer une certaine subjectivité et leur sens (ou sémantique) est déduit du contexte dans lequel ils sont utilisés.

Comme les techniques issues de la logique floue se donnent pour objectif de modéliser et traiter les données imprécises et vagues, nous nous posons les questions suivantes : que peut, d'un point de vue plus général, apporter la logique floue au traitement automatique du langage naturel ? Et en particulier comment les 2-tuples sémantiques peuvent-ils aider le traitement de la langue ?

4.3.1 De la logique floue vers la linguistique

Pour répondre à ces questions, prenons comme exemple une phrase qui pourrait être énoncée par un utilisateur : "Je veux créer une alerte quand le camion est nettement proche de Paris".

L'utilisateur ici nous renseigne sur sa préférence quant à la distance à laquelle il souhaiterait que l'alerte se déclenche. Mais raisonnant naturellement de façon qualitative plutôt que quantitative, il a employé le terme "proche" pour caractériser la distance. Il

convient donc d'interpréter ce terme de la manière la plus juste possible afin que les objectifs métiers de l'utilisateur soient atteints.

Nous proposons une approche d'interprétation sémantique fondée sur la modélisation sous forme de 2-tuples sémantiques. L'idée est que les termes à sémantique floue soient identifiés lors du dialogue et transmis au Geohub sous forme de 2-tuples pour que celui-ci puisse les utiliser lors de traitements qu'il opère, comme la création d'alertes pour notre exemple.

Interprétation de la sémantique

Pour cela, nous nous appuyons sur un expert de géolocalisation qui est responsable de la conception des alertes. Cette fois-ci, il doit non seulement donner ses préférences quant à la position des termes flous sur l'axe des valeurs, mais il choisit également les *termes eux-mêmes* qu'il souhaite utiliser pour les différents paramètres auxquels il a recours en géolocalisation. Ces termes sont utilisés pour générer un partitionnement flou *via* notre approche fondée sur les 2-tuples et sont automatiquement ajoutés au lexique métier puis tagués par des *tags* sémantiques spéciaux correspondants à la notion à laquelle ils sont rattachés.

Par exemple, l'expert peut choisir de caractériser la distance par les termes suivants : *adjacent, proche, avoisinant, moyen, distant, éloigné, loin* et *lointain*. Après avoir donné leur position sur l'axe des valeurs, le partitionnement flou représentant leur sémantique est automatiquement généré.

Ensuite ces termes sont ajoutés au lexique afin d'être pris en compte par l'agent de dialogue en langage naturel lors de l'interaction avec l'utilisateur. Ces termes sont tagués de la manière suivante :

Termes flous du lexique métier.

```
<?xml version="1.0" encoding="UTF-8"?>
 <tokens>
        <!-- DISTANCE -->
        <token gram="ADJ" sem="DISTANCE">adjacent</token>
        <token gram="ADJ" sem="DISTANCE">proche</token>
        <token gram="ADJ" sem="DISTANCE">avoisinant</token>
        <token gram="ADJ" sem="DISTANCE">moyen</token>
        <token gram="ADJ" sem="DISTANCE">distant</token>
        <token gram="ADJ" sem="DISTANCE">éloigné</token>
        <token gram="ADJ" sem="DISTANCE">loin</token>
        <token gram="ADJ" sem="DISTANCE">lointain</token>
 </tokens>
```

Ainsi, grâce au *tag* sémantique métier qui leur est associé, ces termes seront reconnus, annotés et interprétés lors du dialogue.

Lors du traitement des données qu'il reçoit de l'agent de dialogue, le Geohub fait le lien entre les préférences exprimées par l'utilisateur lors du dialogue et la modélisation

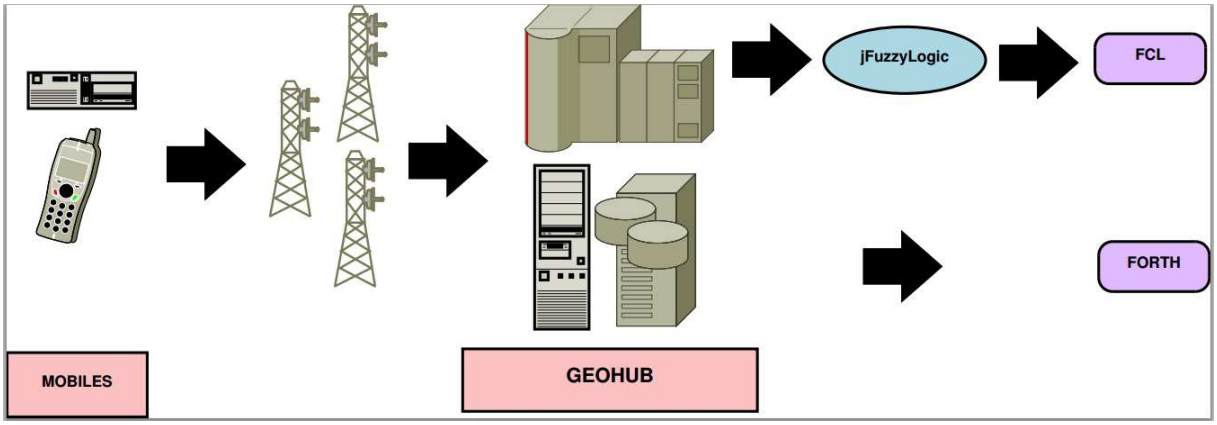


FIGURE 4.3 – Schéma général de l'interprétation sémantique des termes flous.

floue conçue préalablement par l'expert du domaine. Il est ainsi possible de créer ce que nous appelons des *alertes floues* directement, depuis l'interface de dialogue en langage naturel.

La figure 4.3 représente le schéma du processus de l'interprétation des termes et données flous du coté du Geohub. Quand ces termes flous sont employés lors du dialogue, ils sont transmis sous forme de variables 2-tuples sémantiques au Geohub qui s'appuie sur les fichiers Fuzzy Control Language (FCL) définis par l'expert de géolocalisation pour interpréter sémantiquement ces termes. L'interprétation des fichiers FCL se fait le biais de la librairie *jFuzzyLogic* étendue que nous avons précédemment présentée.

Encore une fois, afin que le lexique soit le plus exhaustif possible et qu'il recouvre un maximum de termes susceptibles d'être rencontrés lors du dialogue, nous utilisons notre approche d'extension du lexique par le biais de la relation de synonymie.

En effet, nous ne pouvons pas nous permettre de changer les termes que l'expert a choisi pour définir chaque notion qui se rapproche à son domaine (pour la géolocalisation, ces notions sont par exemple la distance, le temps, le niveau de batterie, la précision de localisation, etc.). Cependant, nous souhaitons pouvoir faire un rapprochement entre ces derniers et les termes que peut employer l'utilisateur quand ils sont proches, d'un point de vue sémantique.

Par exemple, l'utilisateur peut dire : "Je veux créer une alerte quand le camion est nettement contigu à Paris". Le terme "contigu" ne figurant pas dans la liste des termes définis par l'expert (et par extension dans la liste de mots du lexique), celui-ci n'a *pas de sémantique associée.*

Nous proposons donc d'étendre le lexique en y ajoutant les synonymes les plus pertinents pour chaque terme défini par l'expert. Afin de garantir une cohérence d'ensemble, chacun de ces nouveaux termes introduits dans le lexique est tagué et lié au terme originel duquel il est synonyme. Ainsi, le nouveau terme est traité (à savoir modélisé et interprété) exactement de la même manière que le terme auquel il est lié. Ceci nous permet d'étendre le lexique, d'assouplir les contraintes du dialogue (l'utilisateur n'est plus

obligé de n'utiliser que les termes définis) et de garantir une cohérence en respectant les préférences de l'expert.

Pour étendre le lexique des termes flous, nous procédons en utilisant la même approche que celle que nous avons présentée en section 3.2 :

- récupération de listes de synonymes pour un terme donné depuis plusieurs dictionnaires de synonymes (neuf dictionnaires dans notre cas),
- comptage du nombre de fois où revient chaque synonyme dans les listes récupérées,
- conservation seulement des synonymes qui sont présents dans au moins la moitié des listes de synonymes et qui ne soient pas déjà présents dans le lexique.

Par exemple, nous prenons le terme flou "adjacent". Nous récupérons les listes de synonymes lui correspondant depuis nos neuf sources et nous comptabilisons le score de chacun de ses synonymes. Le résultat que nous obtenons est le suivant (chaque terme est suivi du nombre de listes auxquelles il appartient) : proche(8), voisin(7), avoisinant(7), contigu(7), limitrophe(7), attenant(6), prochain(5), jouxtant(5), riverain(5), côte à côte(4), mitoyen(4), adossé(3), collé(3), juxtaposé(3), tangent(3), près(1) et rapproché(1). Ensuite, nous ajoutons au lexique tous les termes ayant un score supérieur ou égal à cinq sur neuf (correspondant au nombre de synonymes présents dans au moins la moitié des listes), comme étant liés au terme "adjacent", à l'exception des termes "proche" et "avoisinant" qui y figurent déjà (définis par l'expert).

Nous considérons, à travers ce filtrage par score, qu'un score faible reflète un **éloignement sémantique** ou encore un sens (trop) particulier du mot en question.

Ainsi, la phrase "Je veux créer une alerte quand le camion est nettement contigu à Paris" sera traitée par l'agent de dialogue de la même manière que la phrase "Je veux créer une alerte quand le camion est nettement adjacent à Paris" [25].

Importance du contexte

Lors de l'interprétation sémantique des termes flous du côté du Geohub, celui-ci s'appuie sur le(s) fichier(s) FCL défini(s) par l'expert. La sémantique de ces termes est donnée par le partitionnement flou auquel ils sont liés. Il convient de dire que, étant par nature flou et imprécis, le sens de ces termes (ou leur sémantique) change selon le **contexte** dans lequel ils sont employés. Par exemple, le terme "proche" peut aussi bien référer à plusieurs kilomètres qu'à une dizaine de mètres. Le contexte regroupe plusieurs paramètres qui varient selon le mobile localisé, le type de l'alerte ou encore le type de localisation : GPS, Wifi ou Cell-Id (en effet, la précision donnée par ces trois types de localisation varie d'une dizaine de mètres pour le GPS à plus d'un kilomètre pour le Cell-Id).

Nous proposons donc un système de contextualisation des termes flous et de leur sémantique en rendant dynamique le choix du FCL à utiliser durant le suivi d'un mobile donné. Ce système prend en entrée toutes les données relatives au mobile concerné et

25. Il est clair que l'objet de ce travail n'est pas de juger du style ou de la qualité (littéraire ou autre) des énoncés, mais plutôt de se focaliser sur leur sens.

choisit en sortie un des fichiers FCL définis par l'expert. Il en découle que, pour une même notion donnée, **plusieurs partitionnements** doivent donc être prévus afin de répondre au mieux aux éventuels contextes dans lesquels peut évoluer le mobile.

Prenons par exemple la définition de la distance que nous avons vue dans l'exemple précédent. Le sens du terme "proche" dépend grandement du contexte du mobile : si le mobile est un véhicule, ce terme n'aura pas la même sémantique que si ce dernier est une personne (marchant à pied). Il ne sera pas interprété de la même manière dans les deux phrases suivantes : "... quand le camion est proche de Paris" et "... quand ma fille est proche de la maison". De plus, même en considérant un véhicule en particulier (un camion par exemple), ce terme change de sémantique selon le type de trajet effectué (trajet de longue distance entre deux villes/pays, par exemple, ou bien trajet pour effectuer plusieurs livraisons au sein d'une même ville).

Ainsi, si nous prenons l'exemple de la distance, le choix de la sémantique à lui rattacher (et donc le choix du fichier FCL à utiliser) dépend du type de mobile, du trajet total qu'il doit effectuer, du type de localisation et du type d'alerte à créer. En effet, le type d'alerte peut ajouter de nouveaux critères à prendre en considération comme la largeur d'un corridor ou la vitesse du mobile.

Le système de contextualisation que nous adoptons est donc un système d'aide à la décision multicritère MCDM permettant de choisir un fichier FCL **dynamiquement** parmi ceux définis par l'expert du domaine. Ainsi, un même terme rencontré durant l'analyse sémantique du dialogue en langage naturel sera interprété différemment selon le contexte.

Pour illustrer le choix contextuel du partitionnement flou, nous prenons un exemple simplifié où l'expert définit trois modèles pour la distance à l'aide des 2-tuples sémantiques. Ces trois définitions de la distance se font comme suit :

- un modèle pour les véhicules réalisant un trajet longue distance,
- un modèle pour les véhicules réalisant un trajet courte distance et
- un modèle pour les personnes.

La figure 4.4 illustre le partitionnement flou généré pour chacun des modèles dans cet exemple de distance.

Évidemment, toutes les combinaisons de critères sont couvertes par le système de contextualisation afin de convenir à tous les contextes éventuels.

4.3.2 De la linguistique vers la logique floue

Dans certains cas particuliers, il arrive que l'expert ne soit pas en mesure de placer précisément les termes qu'il souhaite utiliser sur l'axe des valeurs. Dans ce cas, les termes sont, par défaut, placés uniformément et symétriquement sur l'axe. Cependant, nous souhaitons tirer profit des relations sémantiques entre les termes eux-mêmes afin de mieux les positionner sur l'axe.

Les ensembles de termes notés S qu'utilisent les experts pour définir une notion donnée sont composés de deux sous-ensembles S_1 et S_2. Les termes d'un même sous-

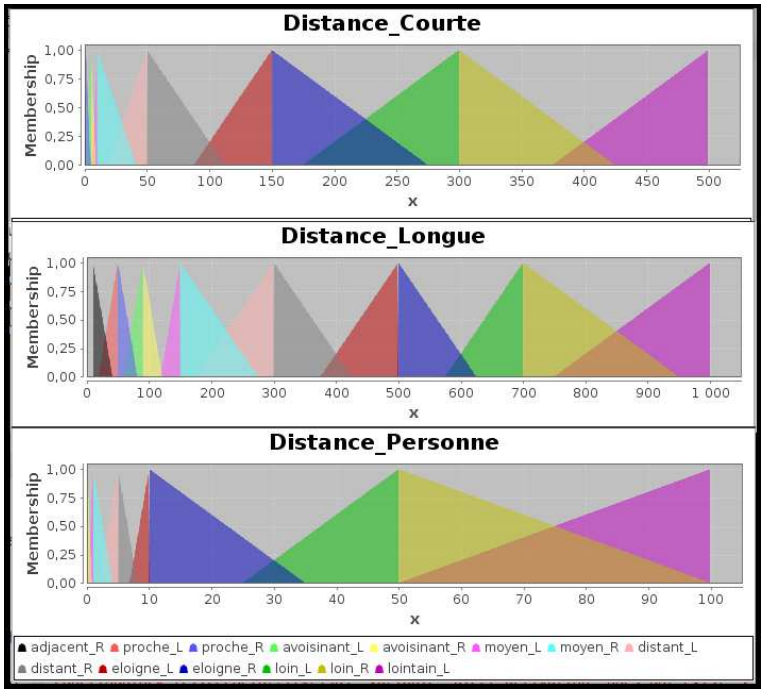


FIGURE 4.4 – Partitionnement flou de la distance selon trois contextes différents.

ensemble S_k sont synonymes entre eux alors que les termes des deux sous-ensembles S_1 et S_2 sont antonymes les uns par rapport aux autres.

Nous allons utiliser la relation de synonymie des termes entre eux (dans chaque sous-ensemble séparément) afin de déduire leurs positions sur l'axe des valeurs.

Avant de détailler l'approche que nous proposons, nous définissons la force du lien sémantique par ce que nous appelons le *taux de ressemblance* (définition 33) entre deux termes $s_i, s_j \in S_k$ où $k = \{1, 2\}$.

Définition 33. *Soient S_k avec $k = \{1, 2\}$ un sous-ensemble de synonymes d'un ensemble de termes S et $(s_i, s_j) \in S_k$ deux termes.*
Le taux de ressemblance r_{ij} entre le terme s_i et le terme s_j est le pourcentage de synonymes qu'il a en commun avec celui-ci. Il est calculé par la fonction de ressemblance R :

$$R : S_k \times S_k \mapsto [0, 1]$$
$$R(s_i, s_j) = r_{ij} = \frac{\nu_{ij}}{\nu_i}$$

où ν_{ij} est le nombre de synonymes communs entre s_i et s_j, et ν_i est le nombre total de synonymes de s_i.

Nous nous intéressons maintenant aux propriétés auxquelles obéit le taux de ressemblance.

Propriétés

- Positivité : le taux de ressemblance entre deux termes s_i et s_j est toujours positif ou nul : $r_{ij} \geq 0$.
  Preuve. ν_{ij} comme ν_i ne peuvent être négatifs, donc il en va de même pour $\frac{\nu_{ij}}{\nu_i}$.
- Réflexivité : la fonction de ressemblance R est réflexive.
  Preuve. $\forall s_i \in S_k : R(s_i, s_i) = r_{ii} = \frac{\nu_{ii}}{\nu_i} = \frac{\nu_i}{\nu_i} = 1$.
- Non commutativité : la fonction de ressemblance R n'est pas commutative.
  Preuve. Etant donné que le taux de ressemblance dépend du nombre de synonymes de chaque terme ν_i alors il est possible que r_{ij} et r_{ji} soient différents. On peut donc trouver au moins un contre exemple à la commutativité de R.
- Existence : la fonction de ressemblance R est toujours définie car le nombre de synonymes d'un terme s_i est strictement supérieur à zéro : $\forall s_i \in S_k : \nu_i > 0$. En effet, on considère qu'il y a toujours au moins un synonyme (sinon, les définitions ne s'appliquent plus car elles sont sans objet).
- Valeur nulle : si le taux de ressemblance entre deux termes s_i et s_j vaut zéro, alors soit les deux termes sont antonymes, soit ils n'ont aucun rapport en terme de sémantique.
- Valeur maximale : le taux de ressemblance ne peut excéder la valeur 1.
  Preuve. Si $R(s_i, s_j) > 1$ alors $\nu_{ij} > \nu_i$. Or, par construction $\nu_{ij} \leq \nu_i$ car il ne peut y avoir plus de synonymes communs entre 2 termes que de synonymes dans un des termes. Donc $\forall s_i \in S_k, \forall s_j \in S_k, R(s_i, s_j) \leq 1$.

Afin de calculer le taux de ressemblance entre un terme s_i et un autre terme s_j, nous nous appuyons sur le dictionnaire de synonymes du CRISCO. Nous avons choisi ce dictionnaire car, contrairement aux autres dictionnaires que nous avons présentés plus haut, il fournit une liste exhaustive [26] de synonymes pour chaque terme.

Nous récupérons la liste de synonymes L_i du terme s_i ainsi que celle, notée L_j, du terme s_j. Nous comptabilisons ensuite le nombre de synonymes en commun des deux listes ν_{ij}. Enfin, nous divisons ce nombre par le nombre total de synonymes ν_i de la liste L_i (c'est-à-dire que $\nu_i = Card(L_i)$) afin d'obtenir la proportion de synonymes en commun correspondant aux taux de ressemblance.

Grâce à ce taux de ressemblance, nous proposons une tentative de quantification de la synonymie entre deux termes donnés. Nous nous servons ensuite de cette sorte de métrique — qui est, bien évidemment subjective, puisqu'aucune métrique standard n'existe — afin de repositionner les termes sur l'axe des valeurs. Plus le taux de ressemblance entre deux termes est important et plus ils sont proches. Inversement, plus le taux est faible et plus l'écart des positions entre les deux termes est important.

Pour y parvenir, nous appliquons la procédure suivante :

- soit $S = \{s_1, s_2, \ldots, s_i, \ldots, s_n\}$ un sous-ensemble de n termes synonymes entre eux ;

26. L'exhaustivité est certes impossible à atteindre dans ce cadre, mais force est de constater après de nombreux tests que la quantité de termes fournie par le dictionnaire du CRISCO est supérieure à celle des autres dictionnaires, en qualité et en quantité.

- pour chaque terme s_i du sous-ensemble, nous récupérons la liste de ses m_i synonymes, notée $L_i = \{\sigma_1, \sigma_2, \ldots, \sigma_{m_i}\}$;
- chaque taux de ressemblance entre les termes de S pris deux à deux, est ensuite calculé : $R(s_i, s_j) = r_{ij} = \dfrac{\nu_{ij}}{Card(L_i)}$;
- les résultats obtenus sont regroupés dans une *matrice de ressemblance* notée M_R (voir définition 34).

Définition 34. *Soit $S = \{s_1, s_2, \ldots, s_n\}$ un sous-ensemble de n termes synonymes entre eux. La matrice de ressemblance M_R du sous-ensemble S est définie par :*

$$M_R(S) = \begin{pmatrix} r_{11} & \ldots & r_{1n} \\ r_{21} & \ldots & r_{2n} \\ . & \ldots & . \\ r_{n1} & \ldots & r_{nn} \end{pmatrix}$$

Problème du positionnement des termes entre eux

La matrice de ressemblance nous permet d'établir un positionnement relatif entre les termes, c'est-à-dire que les taux de ressemblance se traduisent par des **distances** entre les termes, les uns par rapport aux autres. Ceci dit, cette matrice, seule, est insuffisante. En effet, le taux de ressemblance nous renseigne sur la *distance* entre deux termes mais en aucun cas sur la *position* des termes les uns par rapport aux autres.

Prenons par exemple l'ensemble de termes suivants : $S = \{$loin, lointain, éloigné, distant$\}$. La matrice de ressemblance que nous obtenons est la suivante :

$$M_R(S) = \begin{pmatrix} 1 & 0.1 & 0.2 & 0.1 \\ 0.1 & 1 & 0.4 & 0.2 \\ 0.2 & 0.4 & 1 & 0.3 \\ 0.1 & 0.2 & 0.3 & 1 \end{pmatrix}$$

Ici, par exemple, plusieurs possibilités sont valables selon le terme que nous choisissons en premier car les autres termes se positionneront par rapport à lui.

Ceci nous amène à la deuxième problématique qui est le respect de l'*ordre des termes*. En effet, l'ordre étant défini par l'expert du domaine, il est nécessaire de le conserver tout au long du processus.

Problème de l'ordre des termes

Pour cela, nous nous inspirons de la mécanique élémentaire pour proposer notre approche. Plus particulièrement, nous faisons une analogie entre le placement des termes sur l'axe et les systèmes masse-ressort constitués de plusieurs masses reliées entre elles par des ressorts à l'horizontale comme le montre la figure 4.5 [27]. Les positions $x_1(t)$ et $x_2(t)$ des masses au temps t subissent des déplacements sous l'action des forces $F_1(t)$,

27. source : `http://www.iecn.u-nancy.fr/~sokolows/support/node45.html`

$F_2(t)$ et $F_3(t)$ qui leur sont appliquées et qui sont dues aux forces de rappel des trois ressorts.

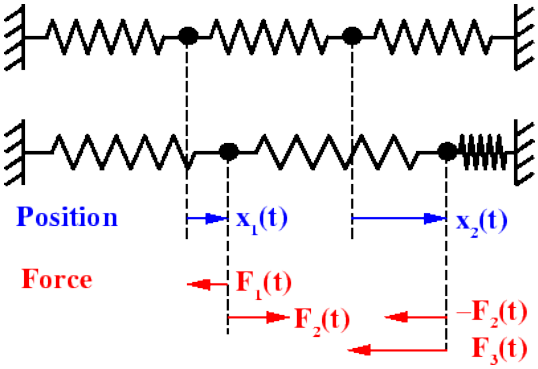


FIGURE 4.5 – Système masse-ressort.

Par analogie, les **termes** seraient donc les **masses**, les **relations sémantiques** entre les termes seraient les **ressorts** et la **force** de ces relations serait la **raideur** des ressorts. Ainsi, plus les termes sont sémantiquement proches (ce qui correspond à une grande raideur) et plus la distance entre eux est petite, et inversement.

Nous posons quelques contraintes afin de garantir une cohérence entre le placement des termes sur l'axe et les valeurs auxquelles ils sont associés :

- l'ordre des termes de départ est conservé ;
- le premier et le dernier terme du sous-ensemble ne changent pas de position. Ceci permet de garantir les valeurs minimale et maximale de l'univers de discours. Si nous reprenons notre analogie avec les systèmes masse-ressort, ces termes peuvent être considérés comme des objets fixes auxquels sont généralement attachés les ressorts — typiquement des murs ;
- le calcul de taux de ressemblance ne se fait qu'*entre deux termes successifs*. L'ordre étant conservé, il n'y a plus d'utilité à calculer *tous* les taux de ressemblance.

Exemple complet

Pour illustrer le calcul des taux de ressemblance, nous prenons l'exemple des trois termes successifs suivants : $S = \{\text{adjacent, proche, avoisinant}\}$.

Le tableau 4.1 met en évidence la liste de synonymes communs à au moins deux termes successifs parmi les trois.

D'après le tableau, nous comptabilisons huit synonymes en commun entre les termes "adjacent" et "proche" et neuf synonymes en commun entre "proche" et "avoisinant".

Il est à noter qu'il existe un cas particulier que nous avons signalé par une astérisque. Ce cas se produit quand un des termes étudiés se retrouve dans la liste de synonymes d'un autre terme et inversement. Ainsi notons-nous le nombre de synonymes entre "adjacent"

adjacent	**proche**	**avoisinant**
-	adjacent*	adjacent
avoisinant	avoisinant*	-
attenant	attenant	attenant
-	circonvoisin	circonvoisin
contigu	contigu	contigu
-	environnant	environnant
joignant	joignant	-
limitrophe	limitrophe	limitrophe
prochain	prochain	prochain
proche*	-	proche*
riverain	riverain	riverain
voisin	voisin	voisin

TABLE 4.1 – Tableau comparatif des listes de synonymes.

et "proche" comme étant 8** (ici, il y a deux astérisques) et le nombre de synonymes entre "proche" et "avoisinant" comme étant 9**.

Cette relation particulière nécessite une attention particulière, car nous considérons que le fait qu'un terme figure dans la liste de synonymes d'un autre terme renforce davantage la relation sémantique qui existe entre les deux. En effet, si deux termes ont plusieurs synonymes en commun mais qu'ils ne figurent pas dans la liste de synonymes l'un de l'autre, leur relation sémantique est moins forte que si c'était le cas.

Nous considérons donc l'astérisque comme un "point bonus" que nous ajoutons au score obtenu par les deux termes l'ayant acquis. Par exemple, le 8** des deux termes "adjacent" et "proche" équivaut en fait à un 10 (8 + 2 points bonus).

Maintenant que nous avons calculé le taux de ressemblance entre les trois termes "adjacent", "proche" et "avoisinant", nous allons nous en servir pour les repositionner sur l'axe des valeurs.

Comme nous l'avons signalé plus haut, quand l'expert ne donne pas les positions des termes, ces derniers sont, par défaut, positionnés de façon uniforme sur l'axe des valeurs. Ils reçoivent donc une position donnée *par défaut*.

Pour nos trois termes, seule la position de "proche" va être modifiée car le premier terme et le dernier terme d'un sous-ensemble restent fixes (*cf.* les contraintes évoquées plus haut).

Ainsi, on peut désormais considérer les **couples sémantiques** suivants, représentant les trois termes : $(adjacent, \mathsf{v}_1)$, $(proche, \mathsf{v}_2)$ et $(avoisinant, \mathsf{v}_3)$. La distance sémantique entre "adjacent" et "proche" étant de 10 et celle entre "proche" et "avoisinant" étant 11 (9 + 2 points bonus), la distance sémantique totale est de 21.
La distance totale sur l'axe des valeurs est, par construction, $\mathsf{v}_3 - \mathsf{v}_1$. Par une simple règle

de trois, nous déduisons que la nouvelle position de "proche" est : $\mathsf{v_2} = \frac{8^{**} \times (\mathsf{v_3} - \mathsf{v_1})}{8^{**} + 9^{**}} = \frac{10 \times (\mathsf{v_3} - \mathsf{v_1})}{21}$.

Généralisation du calcul des positions sur l'axe

La définition 35 généralise le calcul des nouvelles positions sur l'axe des valeurs à tous les termes.

Définition 35. *Soit un sous-ensemble de k termes synonymes entre eux $S = \{s_1, s_2, \ldots, s_k\}$. Le calcul de la nouvelle position des termes s_i avec $i \neq \{1, k\}$ est donné par la formule ci-dessous et permet d'obtenir des couples sémantiques $(\mathsf{s}_i, \mathsf{v}_i)$:*

$$\mathsf{v}_i = \frac{r_{(i-1)i} \times (\mathsf{v}_k - \mathsf{v}_1)}{\sum_{j=1}^{k-1} r_{j(j+1)}}$$

Pour résumer cette approche, on peut dire que la sémantique des termes flous est obtenue de deux manières différentes :

- soit l'expert définit les termes et leurs positions sur l'axe des valeurs par lesquelles il souhaite caractériser une notion donnée et, dans ce cas, la sémantique est modélisée par des 2-tuples sémantiques et le processus de partitionnement que nous avons présenté *via* l'algorithme 1 ;
- soit l'expert se contente de donner les termes qu'il souhaite et les valeurs minimale et maximale de l'univers de discours. Les termes sont, dans un premier temps, positionnés de façon uniforme sur l'axe des valeurs puis leurs positions finales sont obtenues en se fondant sur la force sémantique qui les lie, caractérisée par les taux de ressemblance (*cf.* définition 33).

4.3.3 Les modificateurs sémantiques

Nous nous intéressons maintenant à une autre catégorie de termes flous qui, cette fois-ci, ne portent pas de sens en tant que tels mais **modifient** celui des termes auxquels ils se rapportent. Reprenons la phrase exemple que nous avons citée précédemment : "je veux créer une alerte quand le camion se rapproche nettement de l'entrepôt".

L'adverbe "nettement" intensifie le sens du verbe "se rapproche" dans la phrase. Il est donc nécessaire de prendre en compte ce genre de termes dans l'interprétation sémantique du sens global de la phrase. Ça l'est d'autant plus que, dans notre cas spécifique de déclenchement d'alerte, ces termes impactent le paramétrage à réaliser, et donc indirectement l'adéquation avec les objectifs métiers des experts ou des utilisateurs.

Nous appelons ces termes dans ce qui suit : des *modificateurs sémantiques*.

De la même manière que pour les termes qui constituent notre lexique métier, nous incluons une liste non exhaustive de modificateurs sémantiques au lexique que nous avons préalablement créé. Cette liste est ensuite complétée de la même façon que pour les termes flous en s'appuyant sur la synonymie (*cf.* section 4.3.2).

Le *listing* suivant illustre la manière avec laquelle sont déclarés les modificateurs sémantiques dans le lexique métier :

Modificateurs sémantiques du lexique métier.

```
<?xml version="1.0" encoding="UTF-8"?>
 <tokens>
        <!-- MODIFICATEURS SEMANTIQUES -->
        <token gram="ADV" sem="MODIF_SEM">infiniment</token>
        <token gram="ADV" sem="MODIF_SEM">absolument</token>
        <token gram="ADV" sem="MODIF_SEM">extrêmement</token>
        <token gram="ADV" sem="MODIF_SEM">carrément</token>
        <token gram="ADV" sem="MODIF_SEM">énormément</token>
        <token gram="ADV" sem="MODIF_SEM">fortement</token>
        <token gram="ADV" sem="MODIF_SEM">vachement</token>
        <token gram="ADV" sem="MODIF_SEM">vraiment</token>
        <token gram="ADV" sem="MODIF_SEM">clairement</token>
        <token gram="ADV" sem="MODIF_SEM">nettement</token>
        <token gram="ADV" sem="MODIF_SEM">très</token>
 </tokens>
```

Il convient également de rattacher une sémantique à ces modificateurs sémantiques (!). Pour cela, nous nous inspirons de l'approche des quantificateurs flous afin de modéliser la sémantique de ces modificateurs.

Pour rappel, les quantificateurs flous ont été introduits par Barwise et Cooper (voir notamment [Barwise et Cooper, 1981]) et Zadeh (voir notamment [Zadeh, 1983]) puis redéfinis par Clöckner et Knoll (voir notamment [Glöckner et Knoll, 1997]). Zadeh propose de définir un quantificateur flou comme étant un nombre flou lié à un sous-ensemble flou. La sémantique de ces quantificateurs est déduite à partir d'une fonction d'appartenance. Clöckner et Knoll quant à eux, suivent la vision de Barwise et Cooper et proposent de définir ces quantificateurs comme étant des opérateurs logiques. De cette manière, une expression telle que "quelques personnes âgées" est traduite par une proposition du type : $\exists x \in A$ où x est une variable et A l'ensemble des personnes âgées.

Nous proposons de définir les modificateurs sémantiques par des quantificateurs flous fondés sur la sémantique de nos 2-tuples sémantiques. En effet, puisque ces modificateurs s'appliquent essentiellement aux termes flous que nous avons définis par des 2-tuples sémantiques ("proche", "loin", "fort", "faible", etc.), nous proposons de les définir par des valeurs numériques correspondant à une translation symbolique.

Ainsi, la sémantique d'une expression comme "vraiment proche" est obtenue en appliquant la translation symbolique correspondant à "vraiment", à la sémantique du 2-tuple sémantique qui modélise le terme "proche".

La figure 4.6 illustre la sémantique obtenue pour l'expression "vraiment proche". Cette dernière est obtenue en décalant la fonction d'appartenance triangulaire représentant le terme "proche" (en rouge et correspondant à "CloseTo") sur l'axe des valeurs. Le résultat est représenté par le triangle en pointillés.

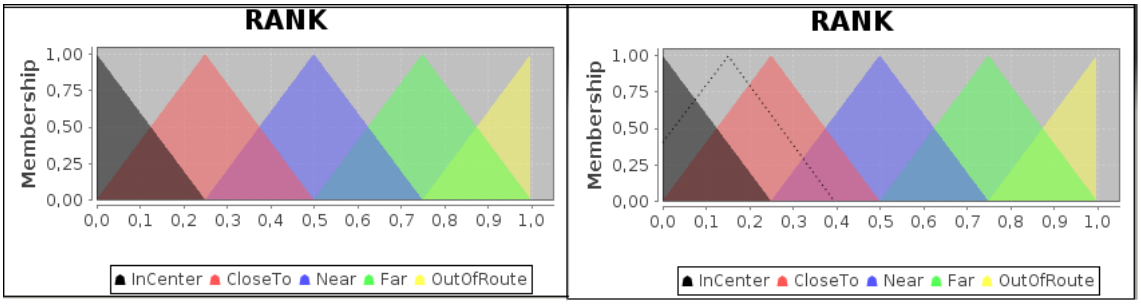


FIGURE 4.6 – Représentation de la sémantique d'un modificateur sémantique.

Nous constituons ainsi un ensemble ordonné de modificateurs sémantiques dont la sémantique (une translation symbolique) est une valeur allant de 1 à 99% de la distance entre le terme auquel nous appliquons le modificateur, et le terme suivant. Nous choisissons de répartir les modificateurs sémantiques de façon uniforme sur la plage de valeurs afin de couvrir l'ensemble de l'univers des modifications possibles.
Par exemple, si nous constituons un ensemble de quatre modificateurs, le premier aura un taux de modification de 20%, le deuxième 40%, le troisième 60% et le dernier 80%. Bien évidemment, nous excluons les valeurs 0% et 100% qui correspondent respectivement à *aucun changement* pour la première et qui obligerait à un changement de terme pour la deuxième.

Le modèle de calcul des 2-tuples et la nature de la translation symbolique font que, lorsque cette dernière dépasse les 50%, il est nécessaire de transformer la sémantique obtenue (car nous rappelons que la translation symbolique est bornée et ne peut excéder la moitié de la distance entre deux termes successifs).

Par exemple, si nous attachons la valeur de 90% au terme "extrêmement" et 10% au terme "presque", la sémantique obtenue pour l'expression "extrêmement proche" sera transformée en celle correspondant à l'expression "presque au centre" (voir le partitionnement de la figure 4.6 avec "au centre" correspondant à "InCenter"). Le choix de la sémantique à associer à chaque quantificateur est effectué par l'expert du domaine de façon empirique.

Il est à noter que la translation symbolique (puisqu'elle est définie comme $\alpha \in [-0.5, 0.5[$, *cf.* définition 31) peut être positive ou négative selon la position sur l'axe du terme flou auquel elle est appliquée. Par exemple, le modificateur "vraiment" génère une translation négative sur le terme "proche" (comme le montre la figure 4.6) car il renforce la notion de rapprochement, alors qu'il induit une translation positive sur le terme "loin" car il renforce la notion d'éloignement ("loin" correspond au terme "Far").

4.4 Discussion

Dans l'état de l'art, la section 2.1 a mis en évidence plusieurs formalismes pour exprimer le CW. Ces modèles sont, pour nous, en quelque sorte, différentes visions du CW.

Nous voudrions pointer ici quelques différences et similitudes pour situer ces modèles ainsi que le nôtre et pour proposer une sorte de comparaison.

4.4.1 Univers, approximation et modèles computationnels : comparaison avec les 2-tuples sémantiques

Nous revenons sur la discussion (*cf.* 2.4.1) concernant une comparaison entre les 2-tuples (linguistiques, proportionnels, et maintenant sémantiques) ainsi que les modèles flou et symbolique.

- Pour l'expression du résultat obtenu, dans le cas de nos 2-tuples sémantiques, l'approximation est portée par les deux translations symboliques $\underline{\alpha_k}$ et $\overline{\alpha_{k+1}}$, et ces deux translations font également office de modificateurs (nous reviendrons sur ce point plus en détail, en section 4.3.3 page 85) ;
- concernant maintenant les modèles computationnels associés à notre formalisme, on peut noter que pour nos 2-tuples sémantiques, c'est la même chose que chez Herrera & Martínez, mais les opérateurs sont étendus aux 2-tuples sémantiques.

4.4.2 Liens entre nos 2-tuples sémantiques et les GSM

Suite à ces réflexions, il nous semble qu'un parallèle peut être établi entre les GSM de Truck & Akdag et nos 2-tuples sémantiques.

En effet, on a vu dans la section 2.1.4 ce qu'étaient les GSM, à savoir des outils qui modifient plus ou moins finement des couples valeur/échelle et qui changent d'échelle. L'échelle est soit dilatée ou érodée, mais elle peut être également conservée. Ces changements d'échelle ne sont pas sans rappeler les changements de hiérarchie évoqués dans la proposition 1, page 68. En effet, un parallèle peut être dressé entre les modificateurs symboliques généralisés et les différentes étapes de l'algorithme permettant de créer la partition des données fortement hétérogène (*cf.* algorithme 1, page 71).

Là où les 2-tuples linguistiques font une différence notoire entre les différentes hiérarchies (c'est encore plus vrai lorsque l'on regarde les travaux sur les ELH, *cf.* page 30), nos 2-tuples sémantiques, tout comme les GSM ne font pas de différence, du moment que l'approximation est suffisante pour exprimer la donnée.

En particulier, avec les GSM, écrire une valeur sur une échelle fine ou sur une échelle plus grossière est considéré comme strictement équivalent. Seul le ratio $\text{Prop}(\tau_i, \mathcal{L}_M) = \dfrac{p(\tau_i)}{M-1}$ (avec $p(\tau_i) = i$) est intéressant. Par exemple, le couple $(\tau_i, \mathcal{L}_M) = (\tau_1, \mathcal{L}_3)$ est strictement équivalent au couple $(\tau_2, \mathcal{L}_5)$ ou au couple $(\tau_4, \mathcal{L}_9)$ puisque $\text{Prop}(\tau_1, \mathcal{L}_3) = \dfrac{1}{2} = \text{Prop}(\tau_2, \mathcal{L}_5) = \dfrac{2}{4} = \text{Prop}(\tau_4, \mathcal{L}_9) = \dfrac{4}{8}$. Par contre, ce n'est pas véritablement équivalent avec les 2-tuples sémantiques car on cherche toujours à exprimer les valeurs dans des échelles les plus grossières possible pour des questions de temps de calcul.

Nous allons montrer la construction d'un "partitionnement" à l'aide des GSM, comme lors du partitionnement à l'aide de nos 2-tuples, en prenant ensuite un exemple.

Construction d'un partitionnement à l'aide des GSM

Un certain nombre de valeurs doivent être considérées pour réaliser le partitionnement. Ces différentes valeurs ne sont pas connues à l'avance et on ne connaît pas non plus leur effectif. Ce qui veut dire que le partitionnement proposé est un partionnement **temporel**, calculé **à la volée**, et que les valeurs en entrée ne sont **pas nécessairement ordonnées**.

L'algorithme à appliquer est le suivant (*cf.* algorithme 2).

Tout d'abord, il faut que les valeurs à partitionner, notées $A, B, C...$, soient des entiers positifs ou nuls (en effet, le i de τ_i est un entier positif au sens large) notés val_A, val_B, val_C, etc. On calcule donc un coefficient multiplicateur γ pour transformer les valeurs en entrée en valeurs entières.

Chaque valeur est transformée en couple $(\tau_i, \mathcal{L}_M)$ selon la règle suivante : le M choisi est égal à $\max(2, val+1)$ car $M \geq 2$ par construction, et τ_i est affecté à τ_{val}. La procédure étant itérative, on note M_1 ce premier M.

Si la valeur suivante à partitionner n'a pas sa place dans l'échelle (c'est-à-dire, si $val_B \geq M_1$), alors il faut choisir un nouveau M (appelé M_2, donc). M_2 prend la valeur $val_B + 1$. On peut maintenant définir le couple (B, val_B) comme étant équivalent à $(\tau_{val_B}, \mathcal{L}_{M_2})$.

Par ailleurs, comme M a changé (M_1 est devenu M_2), il faut recalculer les précédents couples (ici, il faut recalculer la correspondance de (A, val_A) dans la notation de Truck & Akdag). (A, val_A) qui equivalait donc à $(\tau_{val_A}, \mathcal{L}_{M_1})$ est transformé en $(\tau_{val_A}, \mathcal{L}_{M_2})$ **par le biais du GSM** $\mathrm{DW}(M_2 - M_1)$(*cf.* tableau 2.5 page 36).

On continue ensuite pour chaque valeur, en veillant à chaque fois à ce que le coefficient multiplicateur γ permette l'obtention de valeurs entières.

Si ce n'est pas le cas, il faut alors modifier γ (appelé γ_{old}) en un nouveau γ qui rende entières toutes les valeurs considérées. Ainsi, $\gamma = \gamma_{old} * c$.

Il faut ensuite recalculer toutes les valeurs précédentes **en utilisant un GSM** $\mathrm{DC}(c)$ (*cf.* définition 27 page 37). De nouveaux couples $(\tau_{val}, \mathcal{L}_{M_2})$ sont ainsi obtenus.

Ce processus ne s'arrête que lorsqu'on a traité toutes les valeurs en entrée.

Algorithme 2 Algorithme de partitionnement avec les GSM

Require: $A, B, C, ...$ sont les valeurs à partitionner et elles sont notées par des couples $(A_0, val_{A_0}), (A_1, val_{A_1}), ..., (A_i, val_{A_i}), ...$ dans l'algorithme, par commodité
$\gamma \leftarrow 1$; $M \leftarrow 2$; $i \leftarrow j \leftarrow 0$

while il y a des valeurs A_i à partitionner **do**
  if $(val_{A_i} * \gamma) \notin \mathbb{N}$ **then**
    $\gamma_{old} \leftarrow \gamma$
    $\gamma \leftarrow \gamma_{old} * c$ tel que $(val_{A_i} * \gamma) \in \mathbb{N}$
    while il existe des A_j précédemment partitionnés **do**
      recalculer les val_{A_j} (c'est-à-dire des $(\tau_\zeta, \mathcal{L}_M)$) en appliquant DC($c$) : $(\tau_{\zeta_{old}}, \mathcal{L}_{M_{old}}) \xrightarrow{\text{DC}(c)} (\tau_\zeta, \mathcal{L}_M)$
      $j \leftarrow j + 1$
    end while
    $M_{old} \leftarrow M$
    $j \leftarrow 0$
  end if
  $val_{A_i} \leftarrow val_{A_i} * \gamma$
  if $val_{A_i} \geq M$ **then**
    $M_{old} \leftarrow M$
    $M \leftarrow val_{A_i} + 1$
    while il existe des A_j précédemment partitionnés **do**
      recalculer les val_{A_j} (c'est-à-dire des $(\tau_\zeta, \mathcal{L}_M)$) en appliquant DW($M - M_{old}$) : $(\tau_{\zeta_{old}}, \mathcal{L}_{M_{old}}) \xrightarrow{\text{DW}(M - M_{old})} (\tau_\zeta, \mathcal{L}_M)$
      $j \leftarrow j + 1$
    end while
  end if
  calculer val_{A_i} : son couple associé est $(\tau_{val_{A_i}}, \mathcal{L}_M)$
  $j \leftarrow 0$
end while
return l'ensemble des $(\tau_\zeta, \mathcal{L}_M)$ associés à chaque (A_i, val_{A_i})

Exemple

Prenons maintenant l'exemple des différents taux d'alcoolémie dans le sang (*cf.* page 65).

- La première valeur à utiliser pour la construction de la partition est : $(A, 0)$: elle représente, en pourcentage, l'absence d'alcool dans le sang. Notons dès maintenant qu'elle peut s'écrire, en notation GSM, $(\tau_i, \mathcal{L}_j) = (\tau_0, \mathcal{L}_2)$ ou $(\tau_i, \mathcal{L}_j) = (\tau_0, \mathcal{L}_3)$, ou $(\tau_i, \mathcal{L}_j) = (\tau_0, \mathcal{L}_4)$, etc. ;
- la deuxième valeur à utiliser est : $(B, 0.05)$: elle représente, en pourcentage, la limite légale tolérée d'alcool dans le sang pour un conducteur de moins de 21 ans ;
- la troisième valeur à utiliser est : $(C, 0.06)$: elle représente, en pourcentage, le début du stade de désinhibition (la valeur a été arrondie de 0.065 à 0.06 pour que

l'exemple ne soit pas trop long, mais l'algorithme supporte n'importe quelle valeur, bien entendu, du moment qu'elle est positive) ;
- la quatrième valeur à utiliser est : $(D, 0.08)$: elle représente, en pourcentage, la limite légale tolérée d'alcool dans le sang pour un conducteur d'au moins 21 ans ;
- la cinquième valeur à utiliser est : $(E, 0.3)$: elle représente, en pourcentage, la limite à partir de laquelle il y a risque de mort.

Hypothèse 1. Supposons que ces valeurs arrivent dans l'ordre suivant : B, C, A, E, D. En déroulant l'algorithme 2, on obtient :

(i) pour $(B, 0.05)$
- $(\gamma * 0.05) \notin \mathbb{N}$ puisque $\gamma = 1$. Par suite, $\gamma_{old} = 1$, $\gamma = c = 100$ et $val_B = 0.05 * 100 = 5$
- $val_B \geq 2$ donc $M_{old} = 2$ et $M = 5 + 1 = 6$. L'échelle $\mathcal{L}_M$ devient ainsi $\mathcal{L}_6$
- $(B, 0.05)$ s'écrit donc $(\tau_{val_B}, \mathcal{L}_6)$ c'est-à-dire $(\tau_5, \mathcal{L}_6)$

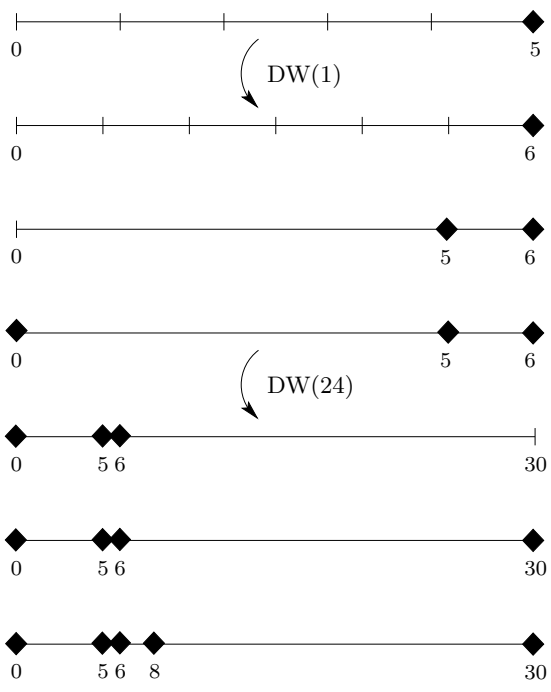
(ii) pour $(C, 0.06)$
- $(\gamma * 0.06) \in \mathbb{N}$ puisque $\gamma = 100$
- $val_C = 0.06 * 100 = 6$
- $val_C \geq 6$ donc $M_{old} = 6$ et $M = 6 + 1 = 7$. L'échelle $\mathcal{L}_M$ devient ainsi $\mathcal{L}_7$
- il faut recalculer B en appliquant $\mathrm{DW}(7-6) = \mathrm{DW}(1) : (\tau_5, \mathcal{L}_6) \overset{\mathrm{DW}(1)}{\longrightarrow} (\tau_5, \mathcal{L}_7)$
- par suite, $(C, 0.6)$ s'écrit donc $(\tau_6, \mathcal{L}_7)$

(iii) pour $(A, 0)$
- $(\gamma * 0) \in \mathbb{N}$ puisque $\gamma = 100$
- $val_A = 0 * 100 = 0$
- $val_A < 7$ donc pas de recalcul
- $(A, 0)$ s'écrit donc $(\tau_0, \mathcal{L}_7)$

(iv) pour $(E, 0.3)$
- $(\gamma * 0.3) \in \mathbb{N}$ puisque $\gamma = 100$
- $val_E = 0.3 * 100 = 30$
- $val_E \geq 7$ donc $M_{old} = 7$ et $M = 30 + 1 = 31$. L'échelle $\mathcal{L}_M$ devient ainsi $\mathcal{L}_{31}$
- il faut recalculer B en appliquant $\mathrm{DW}(31-7) = \mathrm{DW}(24) : (\tau_5, \mathcal{L}_7) \overset{\mathrm{DW}(24)}{\longrightarrow} (\tau_5, \mathcal{L}_{31})$
- il faut recalculer C en appliquant $\mathrm{DW}(24) : (\tau_6, \mathcal{L}_7) \overset{\mathrm{DW}(24)}{\longrightarrow} (\tau_6, \mathcal{L}_{31})$
- il faut recalculer A en appliquant $\mathrm{DW}(24) : (\tau_0, \mathcal{L}_7) \overset{\mathrm{DW}(24)}{\longrightarrow} (\tau_0, \mathcal{L}_{31})$
- par suite, $(E, 0.3)$ s'écrit donc $(\tau_{30}, \mathcal{L}_{31})$

(v) pour $(D, 0.08)$
- $(\gamma * 0.08) \in \mathbb{N}$ puisque $\gamma = 100$
- $val_D = 0.08 * 100 = 8$
- $val_D < 31$ donc pas de recalcul
- $(D, 0.08)$ s'écrit donc $(\tau_8, \mathcal{L}_{31})$

La figure 4.7 résume le déroulement de la construction de ces 5 valeurs avec les GSM.



étape	$(A, 0)$	$(B, 0.05)$	$(C, 0.06)$	$(D, 0.08)$	$(E, 0.3)$
(i)		$(\tau_5, \mathcal{L}_6)$			
(ii)		$(\tau_5, \mathcal{L}_7)$			
		$(\tau_5, \mathcal{L}_7)$	$(\tau_6, \mathcal{L}_7)$		
(iii)	$(\tau_0, \mathcal{L}_7)$	$(\tau_5, \mathcal{L}_7)$	$(\tau_6, \mathcal{L}_7)$		
(iv)	$(\tau_0, \mathcal{L}_{31})$	$(\tau_6, \mathcal{L}_{31})$	$(\tau_5, \mathcal{L}_{31})$		
	$(\tau_0, \mathcal{L}_{31})$	$(\tau_6, \mathcal{L}_{31})$	$(\tau_5, \mathcal{L}_{31})$		$(\tau_{30}, \mathcal{L}_{31})$
(v)	$(\tau_0, \mathcal{L}_{31})$	$(\tau_6, \mathcal{L}_{31})$	$(\tau_5, \mathcal{L}_{31})$	$(\tau_8, \mathcal{L}_{31})$	$(\tau_{30}, \mathcal{L}_{31})$

FIGURE 4.7 – Création du partitionnement temporel avec les GSM, hypothèse 1.

Hypothèse 2. Supposons que ces valeurs arrivent dans l'ordre suivant : E, B, A, C, D. En déroulant l'algorithme 2, on obtient :

(i) pour $(E, 0.3)$
- $(\gamma * 0.3) \notin \mathbb{N}$ puisque $\gamma = 1$. Par suite, $\gamma_{old} = 1$, $\gamma = c = 10$ et $val_E = 0.3 * 10 = 3$
- $val_E \geq 2$ donc $M_{old} = 2$ et $M = 3 + 1 = 4$. L'échelle $\mathcal{L}_M$ devient ainsi $\mathcal{L}_4$
- $(E, 0.3)$ s'écrit donc $(\tau_{val_E}, \mathcal{L}_4)$ c'est-à-dire $(\tau_3, \mathcal{L}_4)$

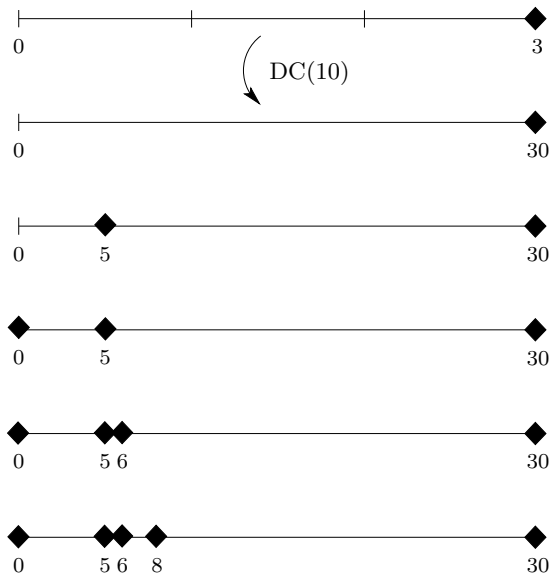
(ii) pour $(B, 0.05)$
- $(\gamma * 0.05) \notin \mathbb{N}$ puisque $\gamma = 10$. Par suite, $\gamma_{old} = 10$, $c = 10$, $\gamma = 100$
- il faut recalculer la valeur précédente E en appliquant $\mathrm{DC}(c) = \mathrm{DC}(10)$: $(\tau_3, \mathcal{L}_4) \overset{\mathrm{DC}(10)}{\longrightarrow} (\tau_{30}, \mathcal{L}_{31})$
- $M_{old} = 31$
- $val_B = 0.05 * 100 = 5$
- $val_B < 31$ donc pas de recalcul
- $(B, 0.05)$ s'écrit donc $(\tau_5, \mathcal{L}_{31})$

(iii) pour $(A, 0)$
- $(\gamma * 0) \in \mathbb{N}$ puisque $\gamma = 100$
- $val_A = 0 * 100 = 0$
- $val_A < 31$ donc pas de recalcul
- $(A, 0)$ s'écrit donc $(\tau_0, \mathcal{L}_{31})$

(iv) pour $(C, 0.06)$
- $(\gamma * 0.06) \in \mathbb{N}$ puisque $\gamma = 100$
- $val_C = 0.06 * 100 = 6$
- $val_C < 31$ donc pas de recalcul
- $(C, 0.06)$ s'écrit donc $(\tau_6, \mathcal{L}_{31})$

(v) pour $(D, 0.08)$
- $(\gamma * 0.08) \in \mathbb{N}$ puisque $\gamma = 100$
- $val_D = 0.08 * 100 = 8$
- $val_D < 31$ donc pas de recalcul
- $(D, 0.08)$ s'écrit donc $(\tau_8, \mathcal{L}_{31})$

La figure 4.8 résume le déroulement de la construction de ces 5 valeurs avec les GSM.



étape	$(A, 0)$	$(B, 0.05)$	$(C, 0.06)$	$(D, 0.08)$	$(E, 0.3)$
(i)					$(\tau_3, \mathcal{L}_4)$
(ii)					$(\tau_{30}, \mathcal{L}_{31})$
		$(\tau_5, \mathcal{L}_{31})$			$(\tau_{30}, \mathcal{L}_{31})$
(iii)	$(\tau_0, \mathcal{L}_{31})$	$(\tau_5, \mathcal{L}_{31})$			$(\tau_{30}, \mathcal{L}_{31})$
(iv)	$(\tau_0, \mathcal{L}_{31})$	$(\tau_6, \mathcal{L}_{31})$	$(\tau_5, \mathcal{L}_{31})$		$(\tau_{30}, \mathcal{L}_{31})$
(v)	$(\tau_0, \mathcal{L}_{31})$	$(\tau_6, \mathcal{L}_{31})$	$(\tau_5, \mathcal{L}_{31})$	$(\tau_8, \mathcal{L}_{31})$	$(\tau_{30}, \mathcal{L}_{31})$

FIGURE 4.8 – Création du partitionnement temporel avec les GSM, hypothèse 2.

Nota Bene.

On constate que, quel que soit l'ordre d'arrivée des valeurs, le résultat du partitionnement est bien le même. Cet algorithme permet donc de partitionner **temporellement** et de façon **unique**, des valeurs, en présentant une échelle qui *ressemble* à l'univers obtenu avec les 2-tuples sémantiques. Bien sûr, la différence s'arrête là où commencent les sous-ensembles flous associés aux 2-tuples.

Le parallèle entre le modèle de nos 2-tuples sémantiques et celui des GSM est le suivant. Les $(s_j^{n(t)}, \alpha_j)$, sans distinction de partie montante ou descendante du 2-tuple, sont liés aux $(\tau_i, \mathcal{L}_M)$ ainsi :

– (s_j, α_j) est l'équivalent de τ_i et
– $n(t)$ est l'équivalent de $\mathcal{L}_M$.

Ce résultat est intéressant si l'on souhaite utiliser différents types de modèles (par contrainte, par besoin ou par envie) : il est ainsi possible de les unifier ou de passer de l'un à l'autre plus facilement.

4.5 Conclusion

Dans ce chapitre, nous avons présenté un modèle *ad hoc* de partitionnement flou fondé sur la modélisation sous forme de 2-tuples sémantiques et utilisant les hiérarchies linguistiques. Il nous a permis d'obtenir des partitionnements flous qui reflètent mieux les préférences des experts du domaine (ou des utilisateurs en général) et qui donnent des résultats plus précis en terme de raisonnement flou.

Il est clair qu'un modèle beaucoup plus simplifié pouvait être mis en place pour la sémantique des mots. En effet, nous aurions pu nous contenter d'attribuer des poids aux différents mots en guise de sémantique. Ces poids auraient ainsi servi à donner à la fois la relation d'ordre entre les mots, et la *force sémantique* de chacun d'eux.

Cependant, attribuer un poids à chaque mot, en sachant que ces poids sont en général compris entre une valeur minimale et une valeur maximale (typiquement entre [0,1]), reviendrait à placer ces mots sur une échelle bornée comme ce qui est fait par le biais des 2-tuples linguistiques et sémantiques. De plus, en s'appuyant *uniquement* sur des poids, nous discrétiserions l'univers de discours ce qui nous ramènerait à un modèle de calcul **avec perte d'informations**. Nous serions, par conséquent, obligés d'arrondir systématiquement le résultat de tout calcul au poids le plus proche. D'autant plus qu'essayer de pallier ce problème de la perte d'information reviendrait à essayer de reconstruire le modèle des 2-tuples et notamment la notion de translation symbolique.

Il reste maintenant à tester notre modèle dans le prototype d'agent intelligent de dialogue en langage naturel, déjà évoqué.

Chapitre 5

Mise en œuvre et résultats

Sommaire

5.1 Implémentation et tests

Afin d'illustrer l'algorithme de partitionnement flou des 2-tuples sémantiques, nous allons détailler ce dernier sur un exemple concret et évoquer l'implémentation à proprement parler des 2-tuples sémantiques, ainsi que la manière de les utiliser.

5.1.1 Implémentation des 2-tuples sémantiques

L'ensemble des applications et services de Deveryware est réalisé et déployé dans un environnement autour des technologies Java et des logiciels libres. Nous avons donc choisi la librairie Java *jFuzzyLogic* [Cingolani et Alcala-Fdez, 2012] pour implémenter les 2-tuples sémantiques, facilitant ainsi leur insertion dans l'environnement technique chez Deveryware [Abchir, 2011].

jFuzzyLogic est une librairie Java implémentant les spécifications du FCL ("Fuzzy Control Langage IEC 61131 part 7", soit en français : "Langage de contrôle flou"), ce qui permet de concevoir facilement des contrôleurs flous sous forme de fichier FCL dans tout programme implémenté en Java.

FCL est un langage développé spécialement pour la conception de contrôleurs flous. Il permet de décrire les entrées, la sortie et les règles du système d'inférence désiré.

Les variables des entrées et celle de la sortie sont définies dans des blocs nommés respectivement FUZZIFY et DEFUZZIFY dont l'élément principal est un TERM. Chaque bloc TERM définit un sous-ensemble flou de la variable en question et décrit sa fonction d'appartenance.

Pour la sortie il est possible de définir, en plus des sous-ensembles flous, la méthode de *défuzzification* et sa valeur par défaut.
L'exemple suivant décrit en FCL la modélisation de l'alcoolémie notée BAC (pour *Blood Alcohol Content*) que nous avons vue en section 4.1 et illustrée par la figure 4.1. `trian` est un mot-clef permettant de définir des fonctions d'appartenance triangulaires. Il est bien évidemment possible de définir d'autres types de fonctions d'appartenance (trapézoïdales, gaussiennes, singletons, etc.).

Modélisation de l'alcoolémie

```
FUZZIFY BAC
    TERM NoAlcohol := trian 0.0 0.0 0.05 ;
    TERM YoungLegalLimit := trian 0.0 0.05 0.06 ;
    TERM Intermediate := trian 0.05 0.06 0.08 ;
    TERM LegalLimit := trian 0.06 0.08 0.3 ;
    TERM RiskOfDeath := trian 0.08 0.3 0.3 ;
END_FUZZIFY

DEFUZZIFY Taux
    TERM Legal := trian 0 0 1;
    TERM Illegal := trian 0 1 1;
    METHOD : COG;
    DEFAULT := 0;
END_DEFUZZIFY
```

Les règles d'inférence sont définies en FCL dans un bloc RULEBLOCK. Dans l'exemple suivant, nous définissons deux règles simples permettant de définir quand l'alcoolémie est un à niveau illégal pour la conduite.

Règles d'inférence pour l'alcoolémie

```
RULEBLOCK Rules
    AND : MIN;
    ACT : MIN;
    ACCU : MAX;
    RULE 1 : IF BAC IS LegalLimit THEN Taux IS Illegal;
    RULE 2 : IF BAC IS RiskOfDeath THEN Taux IS Illegal;
END_RULEBLOCK
```

Extension de FCL

Dans sa version originelle, le FCL (et par extension *jFuzzyLogic*) ne permet de créer que des variables numériques reposant sur des fonctions d'appartenance selon la vision de la logique floue précédemment introduite en section 2.1.1.

Nous proposons donc une extension du FCL afin de prendre en charge les 2-tuples linguistiques et sémantiques. Pour ce faire, nous introduisons deux nouveaux mots-clefs dans la définition de TERM : *ling* et *pairs* correspondant respectivement aux 2-tuples linguistiques et aux couples sémantiques dont la sémantique est décrite par des 2-tuples sémantiques.
Ainsi, nous avons étendu la grammaire FCL dans *jFuzzyLogic* qui est fondée sur une grammaire ANTLR [28] puis nous avons implémenté les différents algorithmes de partitionnement flou. L'alcoolémie peut ainsi être définie en FCL sous forme de 2-tuples comme le montre l'exemple suivant dans lequel la variable BAC_Ling est exprimée par des 2-tuples linguistiques et la variable BAC_Sem est exprimée par des couples sémantiques.

Modélisation de l'alcoolémie

```
FUZZIFY BAC_Ling
    TERM S := ling NoAlcohol YoungLegalLimit Intermediate | LegalLimit |
                   RiskOfDeath, extreme extreme ;
END_FUZZIFY

FUZZIFY BAC_Sem
    TERM S := pairs (NoAlcohol, 0.0) (YoungLegalLimit, 0.05)
                    (Intermediate, 0.06) (LegalLimit, 0.08)
                    (RiskOfDeath, 0.3);
END_FUZZIFY
```

5.1.2 Exemples et comparatifs

Pour dérouler notre algorithme de partitionnement, nous allons modéliser la notion de distance qui est un élément important rencontré dans toutes les applications de géolocalisation.

La notion de distance étant fortement subjective, une modélisation floue en permet une meilleure gestion ainsi que l'introduction de nuances dans les raisonnements qui en découlent.

Pour cela, nous allons proposer une modélisation de la distance dans le cadre d'une alerte de franchissement de corridor au sens Deveryware. Pour rappel (*cf.* section 1.3.1), dans ce type d'alerte, une route est prédéfinie pour un mobile donné (camion, voiture, personne...) autour de laquelle est défini un corridor d'une largeur choisie par l'utilisateur.

Le but de l'alerte est de notifier l'utilisateur quand le mobile sort du corridor (et par extension, de la route) qui a été défini.

Ces alertes sont très souvent utilisées par les compagnies de logistique pour prévenir les vols ou détournements des camions de transport de denrées ou de marchandises de

28. ANTLR : ANother Tool for Language Recognition, voir `http://www.antlr.org`

grande valeur, ou encore dans le cadre des transports de fonds. La modélisation floue permet également de prévenir certains cas de fausses sorties de corridor en "floutant" les bords de celui-ci, ce qui permet de ne déclencher une alerte que quand le mobile a clairement franchi le corridor.

En reprenant l'exemple de la définition de distances (présenté page 63) par un expert Deveryware ainsi que les données statistiques des clients de Deveryware, on choisit les positions des cinq termes, ce qui permet de créer les couples sémantiques de l'ensemble de départ comme suit :
$S = (InTheCenter, 0.0)(VeryCloseTo, 50.0)(Near, 60.0)(Far, 80.0)(OutOfRoute, 300.0)$.

Nous allons détailler le processus d'association de 2-tuples sémantiques pour les deux premiers couples sémantiques et spécifiquement pour la partie descendante du premier et la partie ascendante du deuxième. Pour cela, nous utilisons la hiérarchie linguistique à cinq niveaux que nous résumons dans le tableau 3.1.

niveau	nombre de termes	grain
1	3	150.0
2	5	75.0
3	9	37.5
4	17	18.75
5	33	9.375

TABLE 5.1 – Hiérarchie linguistique pour la distance.

La première étape du processus de partitionnement consiste à trouver le niveau de la hiérarchie linguistique ayant le grain le plus proche de la distance entre les deux couples et ayant une granularité suffisante ($g_t < d$). Ici la distance entre $(InTheCenter, 0.0)$ et $(VeryCloseTo, 50.0)$ est $d = 50.0$. Le niveau de la hiérarchie qui correspond à ces conditions est le niveau $l(3, 9)$.

Une fois le niveau de la hiérarchie choisi, nous sélectionnons parmi ses termes linguistiques le plus adéquat pour représenter le couple sémantique $(InTheCenter, 0.0)$. Bien évidemment, comme la position v égale 0.0, le premier terme du niveau correspond parfaitement au couple et nous choisissons la moitié descendante de ce dernier afin de construire le 2-tuple sémantique $(\underline{s_0^9}, 0)$ et ce, sans appliquer de translation symbolique (α reste donc à 0). Ce couple sémantique étant le premier de l'ensemble de départ, il ne reçoit qu'un demi 2-tuple sémantique.

Nous sélectionnons ensuite le terme suivant du même niveau $(s_1^9, 0)$ pour représenter le couple $(VeryCloseTo, 50.0)$. Mais comme ici le noyau de la fonction d'appartenance associée au terme linguistique $(s_1^9, 0)$ ne correspond pas directement à la position v du couple, nous appliquons une translation symbolique $\alpha = 12.5$ au terme linguistique.
En effet, le terme linguistique $(s_0^9, 0)$ correspond à la position $\mathsf{v} = 37.5$ car $\Delta^{-1}((s_0^9, 0)) = 37.5$. La translation symbolique α à appliquer au terme linguistique est la différence entre sa position absolue sur l'axe et celle du 2-tuple sémantique. Ainsi $\alpha = \Delta^{-1}((VeryCloseTo, 50.0)) - \Delta^{-1}((s_0^9, 0)) = 50.0 - 37.5 = 12.5$.

Nous construisons ainsi la première moitié (partie ascendante) du 2-tuple sémantique associé au couple $(VeryCloseTo, 50.0)$. La moitié descendante du 2-tuple est construite à la prochaine itération du processus de partitionnement.

La figure 5.1 illustre le choix du niveau et des termes linguistiques. Les barres verticales parallèles en rouge représentent la distance $d = 50.0$. Il est clair que le niveau ayant le grain le plus proche de d est le troisième niveau de la hiérarchie indiqué par la flèche verte à droite de l'image. Les deux demi-triangles en gras représentent respectivement la partie descendante du terme linguistique $(s_0^9, 0)$ et la partie ascendante du terme linguistique $(s_1^9, 12.5)$ (obtenu après translation du terme $(s_1^9, 0)$). Ils sont donc choisis pour représenter respectivement la partie descendante du 2-tuple sémantique associé au couple $(InTheCenter, 0.0)$ et la partie ascendante du 2-tuple associé au couple $(VeryCloseTo, 50.0)$.

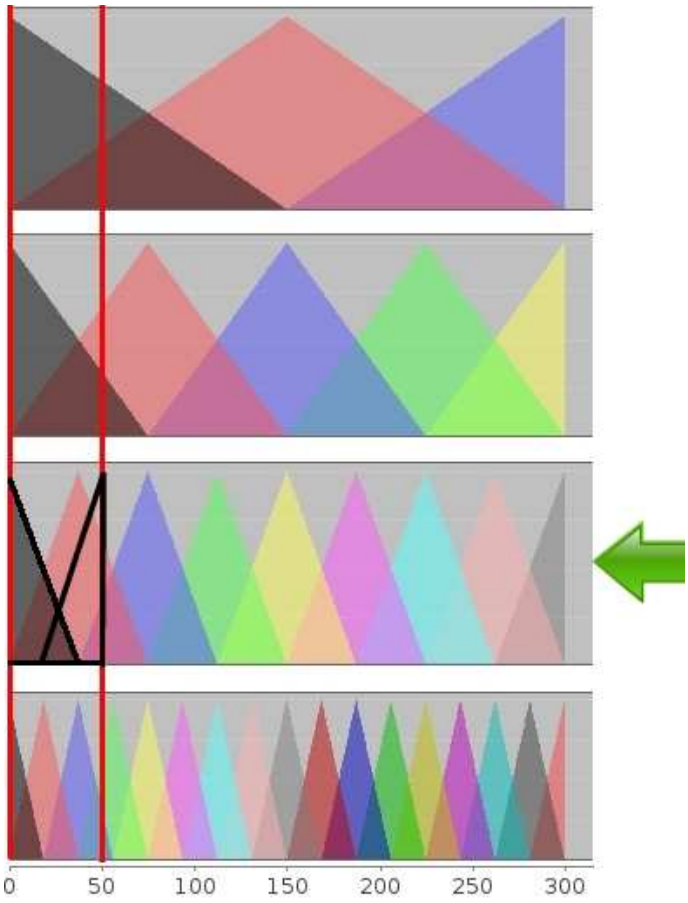


FIGURE 5.1 – Choix de la sémantique pour le 2-tuple sémantique représentant le terme *InTheCenter* dans l'exemple de la distance.

Nous passons ensuite à la paire de couples sémantiques suivante $(VeryCloseTo, 50.0)$ et $(Near, 60.0)$ et nous recommençons le processus de partitionnement. Le tableau 5.2 résume l'ensemble des termes linguistiques choisis pour la représentation sémantique des couples sémantiques de départ. Pour une meilleure compréhension, nous ajoutons le

suffixe "_R" (pour *right*) pour les moitiés descendantes des 2-tuples et le suffixe "_L" (pour *left*) pour leurs moitiés ascendantes.
Ainsi, le couple sémantique $(Near, 60.0)$, par exemple, est associé au 2-tuple sémantique $[(\underline{s_6^{33}, 3.75}), (\overline{s_3^{17}, 3.75})]$.

2-tuple	niveau	terme linguistique
InTheCenter_R	$l(3,9)$	$(s_0^9, 0)$
VeryCloseTo_L	$l(3,9)$	$(s_1^9, 12.5)$
VeryCloseTo_R	$l(5,33)$	$(s_5^{33}, 3.125)$
Near_L	$l(5,33)$	$(s_6^{33}, 3.75)$
Near_R	$l(4,17)$	$(s_3^{17}, 3.75)$
Far_L	$l(4,17)$	$(s_4^{17}, 5.0)$
Far_R	$l(1,3)$	$(s_1^3, -70.0)$
OutOfRoute_R	$l(1,3)$	$(s_2^3, 0)$

TABLE 5.2 – L'ensemble des 2-tuples sémantiques pour l'espace de la distance.

Afin de comparer les partitionnements flous obtenus par les différentes méthodes, nous modélisons à l'aide de la librairie *jFuzzyLogic* étendue le même exemple en utilisant des 2-tuples linguistiques et un modèle idéal de référence. La figure 5.2 illustre le partitionnement idéal (Distance_V1) qui correspondrait au mieux aux choix de l'utilisateur (ou de l'expert), le partitionnement obtenu par la méthode de Herrera et Martínez (Distance_V2) et celui obtenu par notre méthode de partitionnement (Distance_V3).

Nous remarquons assez clairement que le partitionnement obtenu par notre méthode se rapproche plus de celui idéal et reflète donc mieux la réalité. Ceci est dû au fait que les positions souhaitées sont complètement prises en compte lors du processus de partitionnement lui-même.

Comparaison de contrôleurs flous

Nous allons également comparer les deux méthodes de modélisation sur un exemple de raisonnement simple. Un des objectifs visés par le projet SALTY est de concevoir un système de régulation des fréquences de remontées de positions des boîtiers GPS mobiles afin de réduire à la fois les coûts des suivis de véhicules et de réduire la charge de calcul sur le Geohub. En effet, avoir moins de remontées signifie avoir moins de traitements.

L'idée de ce régulateur est de moduler la fréquence à laquelle les boîtiers envoient leurs positions au Geohub selon leurs objectifs métiers (par exemple, rapprochement d'un point d'intérêt, d'un point d'arrivée, des bords d'un corridor, etc.).

Nous reprenons l'idée du régulateur de fréquence sous forme d'un contrôleur flou que nous simplifions à un système à une seule entrée (la distance à l'objectif) et une sortie (la fréquence de remontée que doit adopter le boîtier).

Dans ce test, nous donnons en entrée différentes valeurs de distance de l'univers de discours ($U = [0, 300]$) et nous récupérons la valeur de fréquence donnée par le système

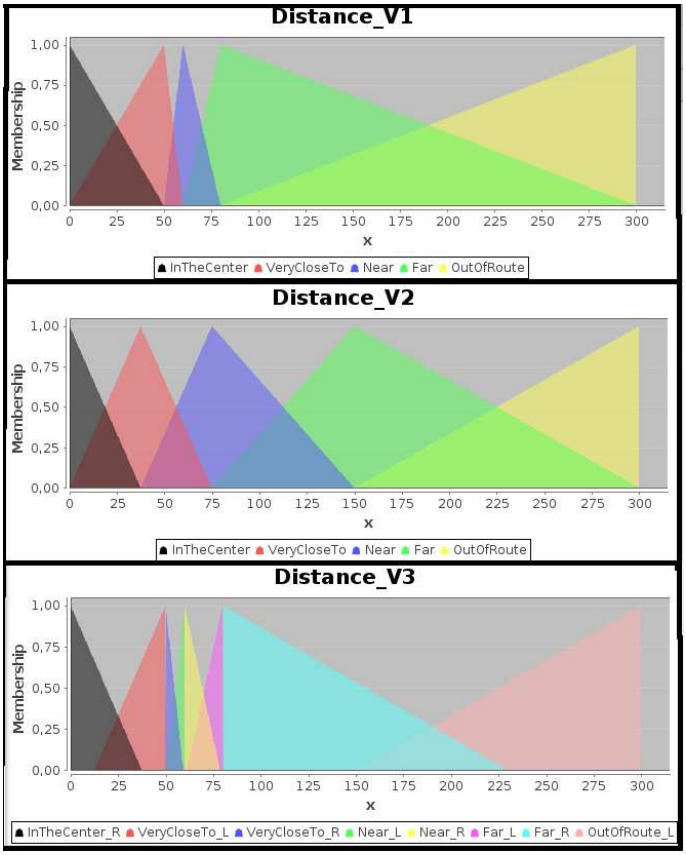


FIGURE 5.2 – Comparaison de partitionnements flous.

d'inférence. Le script FCL suivant décrit la modélisation que nous avons faite de la distance, des fréquences de remontées de positions et des règles d'inférence du contrôleur flou. Ici également, nous avons défini trois modélisations de la distance : Distance_V1 (le modèle idéal de référence), Distance_V2 (modélisation à l'aide de 2-tuples linguistiques) et Distance_V3 (modélisation à l'aide de 2-tuples sémantiques).

Modélisation du régulateur de fréquence

```
FUZZIFY Distance_V1
        TERM InTheCenter := trian 0.0 0.0 50.0 ;
        TERM VeryCloseTo := trian 0.0 50.0 60.0 ;
        TERM Near := trian 50.0 60.0 80.0 ;
        TERM Far := trian 60.0 80.0 300.0 ;
        TERM OutOfRoute := trian 80.0 300.0 300.0 ;
END_FUZZIFY

// Mise à l'échelle de [0, 1] à [0, 300]
FUZZIFY Distance_V2
        TERM S := ling InTheCenter VeryCloseTo Near | Far |
                      OutOfRoute, extreme extreme ;
```

```
END_FUZZIFY

FUZZIFY Distance_V3
        TERM S := pairs (InTheCenter, 0.0) (VeryCloseTo, 50.0)
                        (Near, 60.0) (Far, 80.0) (OutOfRoute, 300.0) ;
END_FUZZIFY

DEFUZZIFY Frequency
        TERM Minimum := trian 2.0 2.0 5.0;
        TERM Weak := trian 2.0 5.0 12.0;
        TERM Medium := trian 5.0 12.0 25.0;
        TERM VeryHigh := trian 12.0 25.0 60.0;
        TERM Maximum := trian 25.0 60.0 60.0;
        METHOD : COG;
        DEFAULT := 0;
END_DEFUZZIFY

RULEBLOCK Regles
        RULE 1 : IF Distance IS InTheCenter THEN Frequency IS Minimum;
        RULE 2 : IF Distance IS VeryCloseTo THEN Frequency IS Weak;
        RULE 3 : IF Distance IS Near THEN Frequency IS Medium;
        RULE 4 : IF Distance IS Far THEN Frequency IS VeryHigh;
        RULE 5 : IF Distance IS OutOfRoute THEN Frequency IS Maximum;
END_RULEBLOCK
```

Les résultats obtenus sont résumés par le graphique de la figure 5.3. Il exprime la fréquence (c'est-à-dire le nombre de positions reçues par heure) en fonction de la distance exprimée en mètres.
La courbe en noir représente l'ensemble solution produit par le partitionnement idéal, celle en vert l'ensemble solution produit par les 2-tuples linguistiques et celle en rouge l'ensemble solution produit par nos 2-tuples sémantiques.

Comme nous pouvons le constater, les résultats obtenus par nos 2-tuples sémantiques se rapprochent grandement des résultats du modèle idéal sauf sur la plage de valeurs $[230, 300]$ où le modèle des 2-tuples linguistiques obtient de meilleurs résultats. Ceci est dû au fait que les valeurs de distance sur cet intervalle ne sont couvertes que par un seul sous-ensemble flou dans la modélisation Distance_V3 alors qu'elles sont couvertes par deux sous-ensembles flous dans Distance_V1 et Distance_V2.

Néanmoins, notre modèle reste globalement meilleur, dans le sens où la précision est beaucoup plus importante quand le mobile se rapproche de ses objectifs que lorsqu'il en est encore loin.

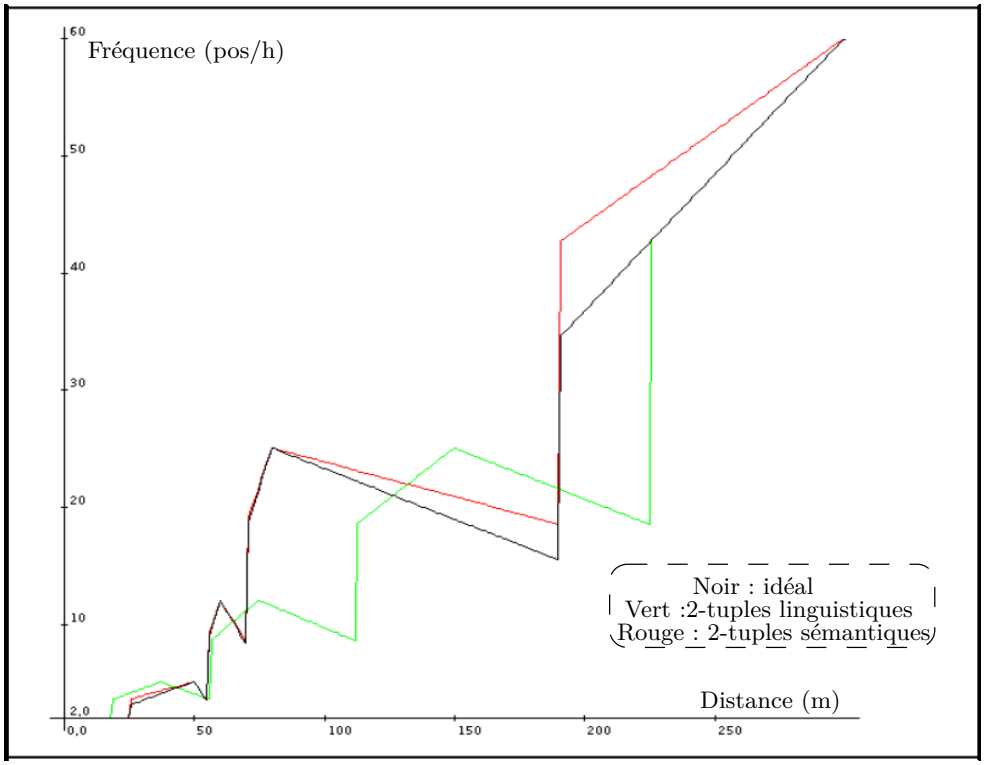


FIGURE 5.3 – Comparaison de résultats du régulateur de fréquence.

Comparaison d'alertes "floues"

L'intégration de nos travaux chez Deveryware nous a menés à créer un type d'alertes bien spécifique : des alertes ayant comme moteur de déclenchement un contrôleur flou qu'on nommera par abus de langage *alertes floues*. Nous allons donc étudier *via* un exemple d'alerte floue l'impact qu'a la modélisation des données sur la justesse de déclenchement des alertes qui en dépendent.

L'alerte en question est une alerte floue de sortie de corridor. Le déclenchement de cette alerte (sortie binaire) dépend de trois éléments : la distance du mobile par rapport aux bords du corridor, le niveau de batterie des boîtiers et un temps passé dans la zone de tolérance. La zone de tolérance est une zone autour des limites du corridor dans laquelle on admet que le mobile reste pendant un certain temps (2 minutes au maximum dans notre cas).

La distance est modélisée à l'aide de cinq termes (*centre, adjacent, proche, loin, lointain*) sur l'univers de discours $U_1 = [0, 1200]$ exprimé en mètres, le niveau de batterie est défini sur $U_2 = [0, 100]$ exprimé en pourcentage par sept termes (*minimum, faible, bas, moyen, haut, fort, maximum*) et le temps de tolérance par trois termes (*minimum, médium, maximum*) sur $U_3 = [0, 120]$ exprimé en secondes. La sortie étant binaire, elle est modélisée par deux termes (*No_Alerte, Alerte*) sur $U_4 = [0, 1]$.

Par le biais des règles d'inférence, nous donnons au contrôleur flou le comportement suivant :

- quand le mobile franchit nettement le corridor, l'alerte est déclenchée ;
- quand le mobile franchit à peine le corridor (correspond à *loin*), que le temps passé en zone de tolérance est *maximum* et que le niveau de la batterie est à *moyen* ou à une valeur supérieure, alors l'alerte est déclenchée ;
- quand le mobile franchit à peine le corridor (correspond à loin), que le temps passé en zone de tolérance est *maximum* mais que le niveau de batterie est à *faible* ou à une valeur inférieure alors l'alerte n'est pas déclenchée (nous préférons d'être sûrs qu'il ait nettement franchi le corridor pour la déclencher afin de préserver la batterie).

Pour notre test, nous fixons le temps de tolérance à 120 secondes (à *maximum*, donc) puis nous faisons varier la distance entre 0 et 1000 mètres avec un pas de 100 et le niveau de batterie de 20 à 100% avec un pas de 20 à chaque itération. Nous exécutons l'inférence avec toutes les combinaisons de valeurs puis nous relevons dans le tableau 5.3 toutes les combinaisons où les résultats obtenus par les trois configurations floues (idéale, 2-tuples linguistiques et 2-tuples sémantiques) ne sont pas identiques.

Distance	Batterie	2-tuples ling.	Idéal	2-tuples sém.
600	80	No_Alerte	Alerte	Alerte
600	100	No_Alerte	Alerte	Alerte
700	80	No_Alerte	Alerte	Alerte
700	100	No_Alerte	Alerte	Alerte
800	80	No_Alerte	Alerte	Alerte
1000	100	Alerte	No_Alerte	No_Alerte

TABLE 5.3 – Comparaison des alertes floues sur l'exemple de sortie de corridor.

Nous constatons, ici également, l'importance d'un partitionnement flou aussi proche que possible de la réalité. Ainsi, nous obtenons, par notre méthode, des résultats de déclenchement identiques à ceux obtenus par le modèle de référence (colonne "Idéal", c'est-à-dire : résultats attendus), alors qu'avec la modélisation des 2-tuples linguistiques, nous obtenons des cas d'alertes non déclenchées là où elles devraient l'être et, cas encore plus problématique (du point de vue métier), nous avons un cas de déclenchement d'alerte alors qu'elle ne devrait pas l'être.

La figure 5.4 montre la visualisation — grâce à l'API[29]) Google Maps — d'un suivi d'un camion dans le cadre d'une alerte floue de sortie de corridor. Dans cet exemple, la notification n'est déclenchée que lorsque le camion est nettement sorti du corridor, conformément au comportement décrit plus haut.

29. API : une interface de programmation est une librairie ou service web offrant des fonctionnalités et services à d'autres programmes. Ces derniers sont souvent accessibles à distance *via* les réseaux informatiques.

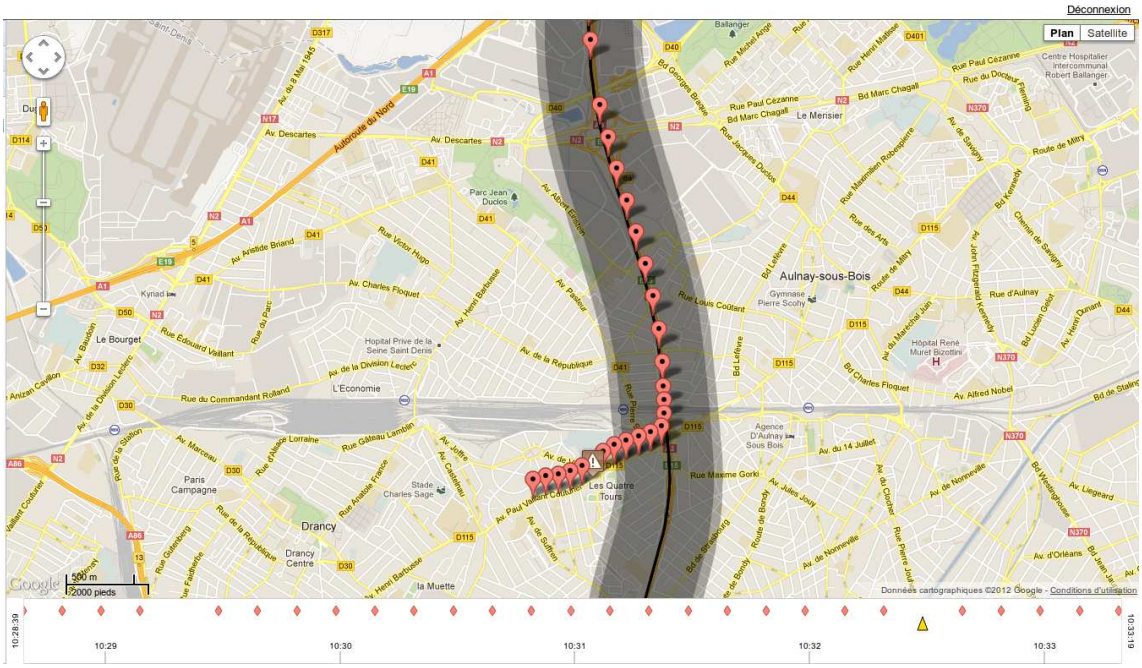


FIGURE 5.4 – Visualisation d'une alerte floue de sortie de corridor.

5.2 Complexité et intégration des algorithmes

5.2.1 Complexité des algorithmes

Nous étudions maintenant la complexité de notre algorithme de partitionnement. On note par c_i une instruction élémentaire où i est l'indice de la ligne correspondant à l'instruction. En regardant l'algorithme 1 page 71, on constate qu'il est composé de trois boucles : la boucle globale parcourant les p couples sémantiques de départ, une boucle imbriquée dans la première parcourant les η niveaux de la hiérarchie et une troisième boucle également imbriquée dans la première parcourant les $n(t_k) - 1$ termes du niveau de la hiérarchie sélectionné. Nous pouvons donc exprimer la complexité C de notre algorithme par :

$$\begin{aligned} C(p) &= p * [c_4 + c_{10} + c_{17} + c_{18} + c_7 * \textstyle\sum_{t=\eta}^{1} + (c_{14} + c_{15}) * \sum_{i=0}^{n(t_k)-1}] \\ &= p * [c_4 + c_{10} + c_{17} + c_{18} + c_7 * \frac{\eta(\eta+1)}{2} + (c_{14} + c_{15}) * \frac{n(t_k)(n(t_k)+1)}{2}] \\ &= p * [4 + \frac{\eta(\eta+1)}{2} + n(t_k)(n(t_k)+1)] \end{aligned}$$

Les deux boucles de recherches s'apparentent à des recherches d'un élément dans un ensemble ordonné fini, elle peuvent être considérées comme des algorithmes sous-linéaires dont la complexité est $\mathcal{O}(\log \eta)$ pour la première et $\mathcal{O}(\log n(t_k))$ pour la deuxième.

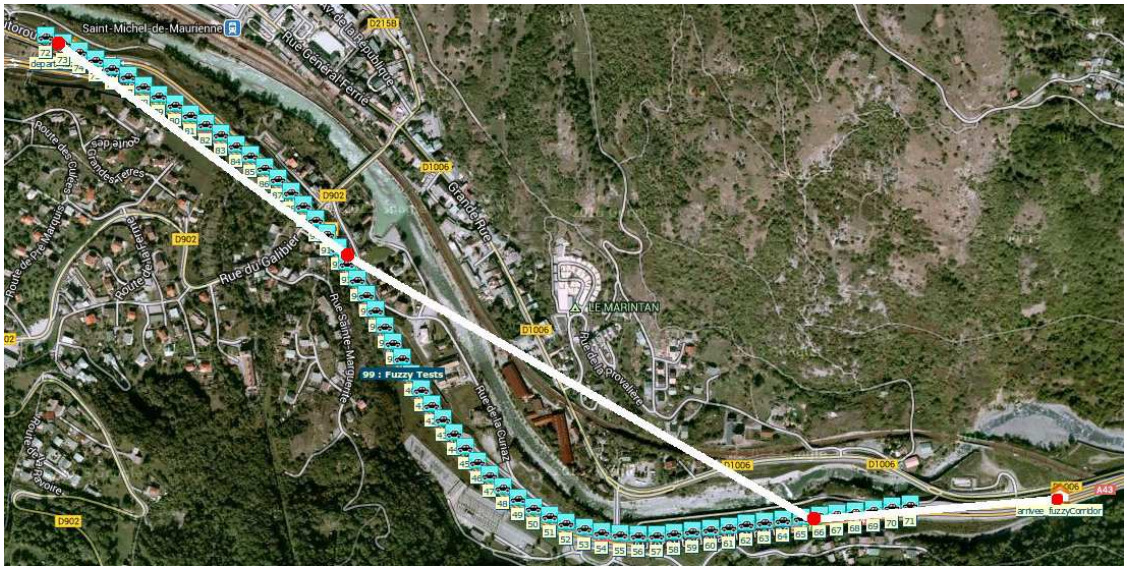


FIGURE 5.5 – Cas d'un parcours avec fausse sortie de corridor.

Comme les instructions c_4, c_{10}, c_{17} et c_{18} et les deux boucles sont exécutées p fois, nous pouvons considérer que l'algorithme est linéaire et de complexité de type $\mathcal{O}(x)$ où x est une valeur dépendant de p, η et $n(t_k)$.

5.2.2 Intégration des travaux chez Deveryware

Nos travaux ont été intégrés au sein de la société Deveryware à plusieurs niveaux. Tout d'abord, des alertes floues ont été incluses à l'application Deveryloc actuelle. Après une phase de tests puis de validation, une version de Deveryloc intégrant les alertes floues a été mise en production afin d'être proposée aux clients de Deveryware.

En particulier, nous avons réalisé des tests de déclenchement d'alertes dans le cadre d'un suivi de corridor. Il s'agissait de tests importants car les clients de Deveryware rencontraient régulièrement un problème concernant de *faux déclenchements* d'alertes. Ces déclenchements non désirés étaient dus à la nature de la construction d'un corridor lors de la création de l'alerte. En effet, lorsqu'un utilisateur souhaite définir une alerte sur sortie de corridor suivant un parcours donné, le corridor créé est composé de plusieurs segments liant deux points successifs de ce parcours. Cependant, si le parcours contient un long virage, il arrive que les deux points du parcours se situent au début et la fin de ce virage. Le segment tracé entre ces deux points se situe donc inévitablement *en dehors* du parcours indiqué. Ainsi, quand le mobile en question suit le trajet défini et parcourt le virage, le système constate qu'il est (légèrement) en dehors du corridor et déclenche l'alerte. Ce cas d'usage est illustré par la figure 5.5. Nous y expliquons trois choses : (i) l'historique des positions du mobile, (ii) le corridor construit lors de la création de l'alerte (ligne blanche) et (iii) les points (rouges) ayant servi à construire le corridor.

Pour notre test, nous avons créé, en même temps et pour le même mobile, deux alertes de sortie de corridor : une alerte "classique" et une fondée sur une inférence floue. Nous avons fait parcourir ensuite à ce mobile un trajet contenant un virage. Nous avons constaté que seule l'alerte "classique" se déclenche dans ce cas. Nous avons renouvelé le test sur plusieurs parcours différents (notamment 3 ou 4 cas problématiques, dans le Sud-Est de la France et en Ile-de-France) contenant des virages et nous avons également constaté le même résultat.

Les alertes floues ont permis de répondre aux exigences métiers des clients en permettant notamment à Deveryware de régler plus finement le degré de vérité requis en sortie du système d'inférence pour le déclenchement d'alertes. Ceci est très utile pour répondre uniformément aux exigences disparates, en terme de précision de déclenchement d'alertes, entre les différents clients de la société. Par exemple, un transporteur de denrées alimentaires et un convoyeur de fonds n'ont pas les mêmes besoins/exigences en terme de précision.

A terme, les alertes floues pourraient être complètement personnalisées par les clients eux-mêmes en choisissant d'abord quelles entrées utiliser pour une alerte donnée (distance, vitesse, niveau de batterie, données du véhicule remontée par le bus CAN...). Ils choisiraient ensuite les termes qu'ils souhaitent utiliser pour chaque entrée ainsi que pour la sortie et les placeraient sur une échelle de valeurs. Enfin, il décriraient les règles d'inférence régissant le déclenchement de l'alerte en question. Ceci reviendrait à leur offrir un **assistant de génération automatique de fichiers FCL**.

Aussi, nos travaux sur l'agent intelligent de dialogue ont été testés en situation réelle lors de la démonstration finale du projet ANR LEA [30]. Ce projet visait à produire une balise GPS intégrant, en plus de l'antenne GPS, un système utilisant à la fois des capteurs inertiels (gyroscope et accéléromètres) ainsi qu'une triangulation de signaux radio FM spécifiques. Ces systèmes permettent de suppléer la puce GPS en cas d'absence ou de brouillage du signal GPS. Dans le cadre de la démonstration du projet LEA, Deveryware a inclus une démonstration de la commande vocale de la balise LEA *via* l'agent intelligent de dialogue. En effet, la démonstration du projet consistait en un parcours prédéterminé à bord d'une voiture équipée de la balise LEA et d'un brouilleur de signal GPS qui est déclenché sur une portion donnée du parcours. Durant le parcours, le directeur de projets de Deveryware a utilisé l'agent intelligent depuis un *smartphone* Android pour consulter et changer les paramètres de la balise en temps réel (changer la fréquence de remontées de positions, activer/désactiver la localisation radio...). La démonstration s'est déroulée avec succès.

5.3 Conclusion

Nous avons proposé une approche permettant aux experts du domaine et aux utilisateurs d'exprimer leurs besoins et préférences métiers à travers un dialogue en langage naturel. Cette approche s'appuie sur l'établissement d'un lexique métier tagué (PST)

30. LEA : Localisation en Environnement Adverse, *cf.* le site `http://www.systematic-paris-region.org/fr/projets/lea`

qui rassemble trois catégories de termes : des termes métiers, des termes flous et des modificateurs sémantiques.

Afin d'accomplir le dialogue en langage naturel, nous avons adopté une approche similaire à celle des agents intelligents, aboutissant ainsi à un *agent intelligent de dialogue*. Ce dernier s'appuie, en plus du lexique métier, sur une (ou plusieurs) grammaire(s) métier(s) servant de scénario de dialogue avec l'utilisateur. Cette grammaire est décrite suivant une syntaxe EBNF et modélisée, lors de l'exécution, sous forme d'arbre TAG.

Nous nous sommes appuyés sur les 2-tuples sémantiques pour modéliser et interpréter les différents termes du lexique. Nous avons également proposé une méthode de partitionnement flou en utilisant les *relations de synonymie* entre les termes afin de mieux les positionner sur l'axe des valeurs.

La figure 5.6 illustre le schéma de l'interprétation du dialogue en langage naturel. Les phrases de l'utilisateur sont tagués suivant le lexique métier, elles passent ensuite par une phase de validation par la grammaire métier ; vient ensuite la phase d'analyse sémantique, de la génération des 2-tuples sémantiques correspondants et de la déduction des paramètres et actions à effectuer.

Ces travaux ont pu être intégrés au sein de la société Deveryware et ont permis de mettre en place des alertes de géolocalisation fondées sur une inférence floue. Ces dernières peuvent facilement être créées et paramétrées à l'aide de notre assistant de géolocalisation *via* un dialogue en langage naturel. Ces travaux ont été mis en production récemment et ont pu être testés lors de la démonstration d'un projet ANR nommé LEA.

L'approche que nous avons présentée, bien qu'étant appliquée au domaine de la géolocalisation pour les besoins de nos travaux, reste généralisable et applicable à n'importe quel domaine ou corps de métier.

En effet, les étapes de construction et d'extension du lexique, de la construction de la grammaire et des *scenarii* de dialogue ainsi que l'interprétation sémantique peuvent être adaptés à d'autres domaines donnant lieu à des agents intelligents spécialisés dans les domaines ciblés.

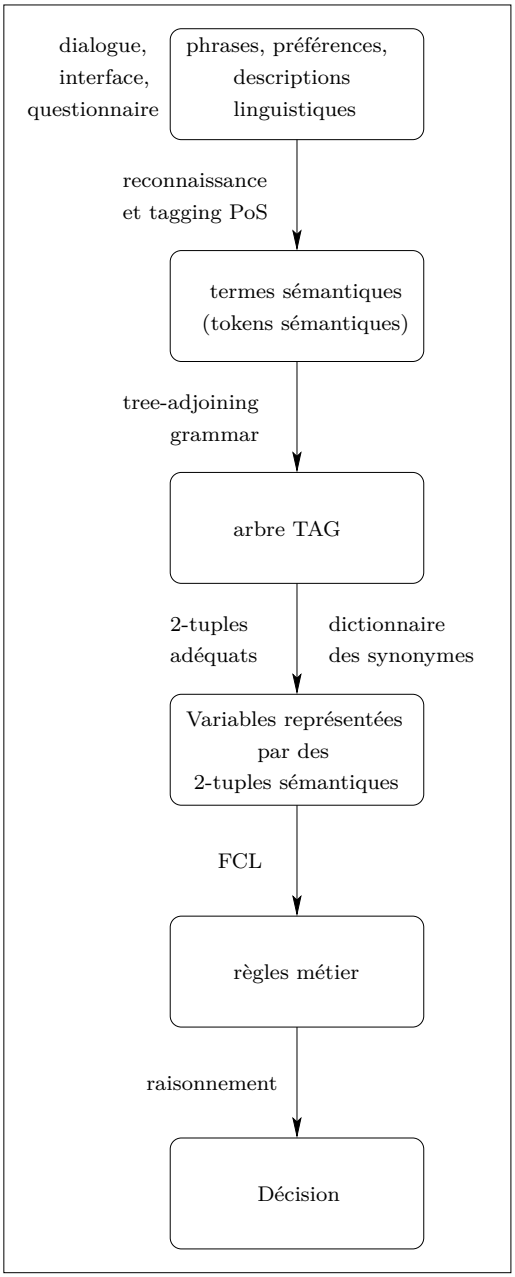


FIGURE 5.6 – Schéma général de l'interprétation du dialogue en langage naturel.

Conclusion et perspectives

1 Conclusion

Cette thèse a eu pour point de départ des besoins industriels visant à proposer des solutions à deux problèmes selon deux axes de recherche s'emboîtant l'un dans l'autre.

La question posée était : "dans un cadre de géolocalisation, comment faire en sorte que les besoins métiers des clients *utilisateurs* des services soient traduits correctement en événements et en contraintes de géolocalisation ? Et en particulier, comment passer d'un échange avec le client, échange dans lequel le client exprime ses préférences et ses choix, à une configuration adéquate des boîtiers ainsi qu'à la mise en place des alertes appropriées ?"

D'abord, pour éviter à l'utilisateur final de travailler dans le contexte quantitatif des paramètres techniques de l'application et du domaine, la recherche visait à le remettre dans le contexte qualitatif de ses objectifs métiers en lui proposant d'exprimer ses besoins sous la forme de paramètres linguistiques provenant du langage naturel.

La recherche a visé à montrer comment on peut améliorer une interface utilisateur existante pour passer d'un traitement quantitatif à un traitement qualitatif des occurrences d'événements et des contraintes de géolocalisation. L'objectif était de montrer que les techniques linguistiques et les outils de la logique floue peuvent permettre de passer automatiquement d'une expression des besoins de l'application en paramètres de configuration de la géolocalisation, de même que de savoir calculer de nouveaux paramètres quantitatifs lors de modifications qualitatives de la situation du mobile.

Dans ce cadre, la stratégie qui a été adoptée est une stratégie d'explicitation fondée sur le dialogue en langue naturelle avec l'utilisateur, qui indique, de la manière la plus proche de ses préoccupations métier, les besoins et objectifs de son application.

Afin d'analyser les réponses de l'utilisateur, nous avons proposé une méthode fondée sur l'utilisation :

- d'un lexique métier automatiquement étendu. Il regroupe l'ensemble des termes reconnus par le système de dialogue. Ces termes sont tagués par des tags sémantiques et grammaticaux ;
- d'une grammaire métier qui représente les concepts métiers souhaités et leurs spécifications. La grammaire est utilisée comme base pour les *scenarii* de dialogue avec l'utilisateur.

Dans un deuxième volet, nous avons proposé un modèle de représentation de données qualitatives vagues et imprécises fondé sur nos 2-tuples sémantiques et utilisant des couples sémantiques comme entrées du système de représentation. Notre but premier était de générer des partitionnements flous répondant le plus fidèlement possible aux besoins exprimés par un expert du domaine. Nous avons donc proposé un modèle *ad hoc* où l'utilisateur peut indiquer, à l'aide des couples sémantiques, les termes qu'il souhaite utiliser ainsi que leurs positions sur l'axe des valeurs (l'univers de discours).

L'algorithme de partitionnement flou que nous avons proposé a permis de générer des partitionnements flous fidèles aux attentes des experts en associant à chaque couple sémantique le 2-tuple sémantique qui lui correspond le mieux et ce, en composant deux demi 2-tuples à partir d'une hiérarchie linguistique.

Nous avons également proposé un modèle de calcul pour les 2-tuples sémantiques s'inspirant de celui des 2-tuples linguistiques afin de permettre de raisonner à partir de données modélisées à l'aide de nos outils.

Les 2-tuples sémantiques ont ainsi permis de garder l'avantage des calculs sans perte d'information des 2-tuples linguistiques tout en palliant le manque de justesse ou de fidélité du partitionnement flou dont pouvaient souffrir ces derniers et notamment quand les données de départ sont fortement déséquilibrées sur leur axe.

Nous avons ensuite proposé une interprétation sémantique (sémantique que nous avons caractérisé comme étant floue) fondée sur une représentation sous forme de 2-tuples sémantiques. Ainsi, il nous a été possible d'attacher une sémantique adéquate aux termes du lexique en les liant aux divers partitionnements flous contextualisés, définis par l'expert du domaine. De cette manière, un même terme est interprété différemment selon le contexte dans lequel il est utilisé. L'expression personnelle faisant appel au langage naturel peut trouver son interprétation sémantique grâce à nos 2-tuples.

De surcroît, nous avons proposé une méthode de quantification des relations sémantiques entre synonymes fondée sur ce que nous avons appelé un *taux de ressemblance*. Ceci a abouti à une méthode de placement automatique des termes sur l'axe des valeurs en tirant parti de leurs relations sémantiques.

Enfin, nous avons proposé une méthode d'interprétation des modificateurs sémantiques où ces derniers se traduisent par une translation symbolique appliquée au 2-tuple sémantique lié au terme auquel ils sont associés.

Afin de réaliser une implémentation de notre modèle, nous avons, dans un troisième temps, proposé une extension du langage FCL afin d'intégrer la modélisation sous forme de 2-tuples (linguistiques et sémantiques). Ainsi, nous en avons réalisé une extension de la librairie *jFuzzyLogic*, elle-même écrite en Java.

De surcroît, des tests grandeur nature ont été réalisés. Désormais, les alertes peuvent être des *alertes floues* (inférences floues). Des mises en situation de notre agent intelligent de dialogue en langage naturel ont eu lieu chez Deveryware, suivies d'une mise en production.

Finalement, cette thèse nous a permis, au demeurant, d'explorer davantage les articulations qui peuvent exister entre les techniques de la logique floue et celles du traitement automatique du langage naturel. En particulier, nous avons mis en relief quelques contributions mutuelles que peuvent s'échanger ces deux domaines.

Dans la section suivante, nous présentons des perspectives de recherche concernant nos travaux.

2 Perspectives

2.1 Pour les 2-tuples sémantiques

Vers un modèle entièrement linguistique

Quand nous traitons des outils linguistiques, l'objectif premier est d'éviter à l'utilisateur de fournir des chiffres précis car il n'est pas toujours en mesure de les produire. Il convient donc de lui offrir un contexte qualitatif au lieu de celui quantitatif habituellement rencontré.

Ainsi, dans un couple sémantique (s, v) décrivant une donnée en particulier, il est possible que l'utilisateur ne sache pas exactement la position v à fournir.

Par exemple, considérons les cinq notes suivantes (dans un système de notation américain) : (A, B, C, D, E). L'utilisateur sait que :

- les notes D et E sont synonymes d'échec ;
- la note A est la meilleure note possible ;
- la note B n'est pas très loin derrière A ;
- la note C est au milieu.

Si nous remplaçons la valeur v par un terme linguistique, nous obtenons ce que nous pouvons appeler *un facteur d'étirement* et les cinq couples sémantiques pourraient par exemple être : (A, *TrèsCollé*); (B, *Loin*); (C, *Collé*); (D, *MoyennementCollé*); (E,N/A) (voir figure 5.1).

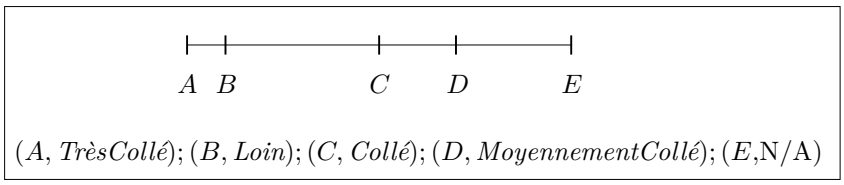


FIGURE 5.1 – Exemple d'utilisation des facteurs d'étirement

(A, *TrèsCollé*) veut dire que A est très collé au terme qui lui succède (à savoir B). (E,N/A) veut dire que E est le dernier terme et donc que le facteur d'étirement n'est pas applicable.

Cette amélioration permettrait ainsi de demander à l'utilisateur :

- soit les couples sémantiques (s, v), avec la valeur v sous forme de terme linguistique (facteur d'étirement) ;
- soit seulement les termes en les plaçant visuellement sur une échelle, et dans ce cas les facteurs d'étirement peuvent être automatiquement déduits afin d'obtenir les couples (s, v) ;
- soit les couples sémantiques (s, v) avec cette fois-ci les valeurs v sous forme numérique comme nous l'avons présenté dans le chapitre 4.2.

Il est à noter que le premier cas permettrait de traiter des couples entièrement linguistiques (s, v), ce qui nous orienterait vers un modèle entièrement linguistique.

Nous précisons également que nos facteurs d'étirement peuvent s'apparenter (dans la finalité) aux densités utilisées par Herrera & Martínez dans leur partitionnement flou (*cf.* paragraphe 2.1.2). Mais, dans notre cas, le facteur d'étirement étant lié à chaque terme, il permettrait de générer une représentation plus précise des termes.

Vers une simplification des arbres binaires

Le modèle des 2-tuples linguistiques, en utilisant des couples $(s_i^{n(t)}, \alpha)$ et un niveau de hiérarchie linguistique associé, peut être vu comme une méthode alternative pour représenter les nœuds d'un arbre. Nous pouvons dresser un parallèle entre la profondeur des nœuds et les niveaux de la hiérarchie linguistique.

En effet, considérons un arbre binaire, pour simplifier. Nous pouvons associer le nœud racine au premier niveau de la hiérarchie linguistique qui est $l(1, 3)$ (*cf.* figure 2.5). Ensuite, nous associons ses nœuds fils au deuxième niveau de la hiérarchie, à savoir $l(2, 5)$, sachant qu'un niveau est obtenu à partir de son prédécesseur suivant la règle : $l(n + 1, 2n(t) - 1)$.
Nous continuons ainsi de suite jusqu'à ce tous les nœuds de l'arbre soient associés à un niveau de la hiérarchie.

Dans le cas simple d'un arbre binaire (chaque nœud a soit deux nœuds fils, soit aucun fils) la déduction de la position — et donc du 2-tuple $(s_i^{n(t)}, \alpha)$ — de chaque nœud est facile : la position est unique, à gauche du nœud parent dans le niveau suivant de la hiérarchie pour le nœud fils gauche et à droite pour le nœud fils droit.

L'algorithme permettant de simplifier un arbre binaire à l'aide d'un ensemble de 2-tuples linguistiques est donné par l'algorithme 3.

Si nous considérons l'exemple graphique donné par la figure 5.2, nous obtenons l'ensemble de 2-tuples suivant (ordonné par niveau de la hiérarchie) :
$\{(s_1^3, 0), (s_1^5, 0), (s_3^5, 0), (s_5^9, 0), (s_7^9, 0), (s_9^{17}, 0), (s_{11}^{17}, 0)\}$, où $a \leftarrow (s_1^3, 0)$, $b \leftarrow (s_1^5, 0)$, $c \leftarrow (s_3^5, 0)$, $d \leftarrow (s_5^9, 0)$, $e \leftarrow (s_7^9, 0)$, $f \leftarrow (s_9^{17}, 0)$ et $g \leftarrow (s_{11}^{17}, 0)$.
Le dernier graphe de la figure représente la sémantique obtenue par l'algorithme de partitionnement des 2-tuples linguistiques 2.1.2.

De cette manière, l'algorithme permet d'"aplatir" un arbre binaire en un ensemble de 2-tuples, ce qui peut être très utile, par exemple, pour exprimer la distance entre les nœuds.

Algorithme 3 Algorithme de simplification

Require: o est un nœud, T est un arbre binaire, o' est le nœud racine de l'arbre T
1: $o' \leftarrow (s_0^3, 0)$
2: **for** chaque nœud $o \in T, o \neq o'$ **do**
3: soit (s_i^j, k) le nœud parent de o
4: **if** o est nœud fils gauche **then**
5: $o \leftarrow (s_{2i-1}^{2j-1}, 0)$
6: **else**
7: $o \leftarrow (s_{2i+1}^{2j-1}, 0)$
8: **end if**
9: **end for**
10: **return** l'ensemble de 2-tuples linguistiques, un par nœud

L'inverse reste vrai : un ensemble de termes linguistiques peut être représenté sous forme d'arbre binaire.

Un des avantages de cette simplification est de considérer une nouvelle dimension pour les données d'un problème. Cette nouvelle dimension est la distance entre les différents résultats possibles du problème (les nœuds pouvant être des décisions, des choix, des préférences, etc.) et ceci permettrait par exemple de classer et noter les résultats (suivant ces distances), un peu comme si nous avions des B-arbres.

Le fait que le niveau de la hiérarchie associé à chaque nœud n'est pas le même, suivant la profondeur de ce dernier, est intéressant car il offre une granularité différente qui, à l'image des granules de Zadeh, permettrait de lier une position donnée dans l'arbre à un niveau de précision.

Vers une inférence de 2-tuples sémantiques

Une autre voie à explorer est celle de la construction d'un système d'inférence entièrement fondé sur les 2-tuples sémantiques. En effet, nous avons présenté des opérateurs permettant de réaliser des calculs à l'aide des 2-tuples sémantiques. Néanmoins, l'inférence qui est utilisée repose encore sur des outils issues de la logique floue de Zadeh comme l'application de la t-norme et t-conorme lors de l'interprétation des règles d'inférence.

Il convient donc d'étudier la possibilité d'appliquer le modèle d'inférence de la logique floue aux 2-tuples sémantiques mais en redéfinissant chacun des opérateurs intervenant dans le processus afin de les adapter.

Ceci reviendrait à étudier les t-normes et t-conormes applicables aux 2-tuples et les conditions auxquelles elles obéissent.

2.2 Pour le traitement du langage

En ce qui concerne notre approche de traitement du langage naturel et notamment dans le cadre d'un dialogue, une première amélioration consisterait à prendre en consi-

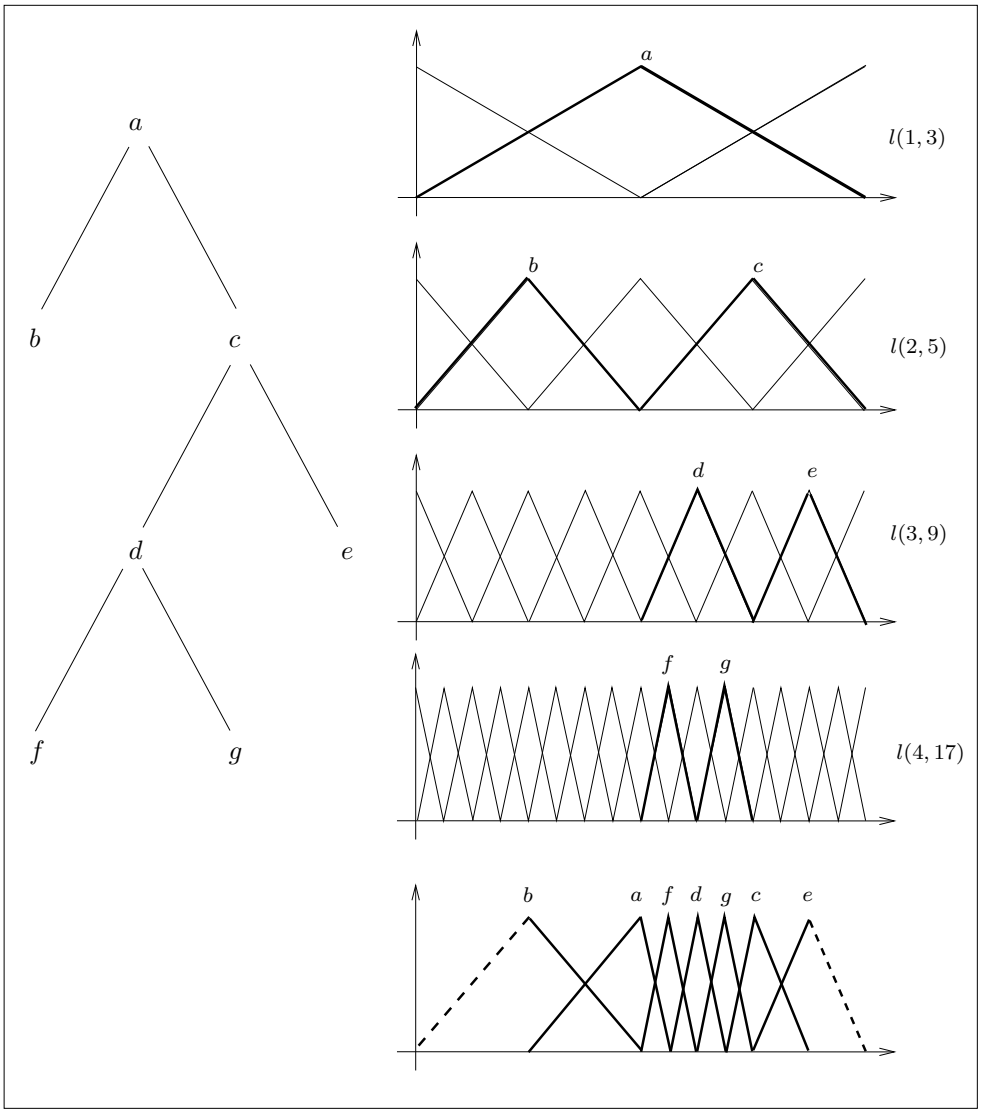


FIGURE 5.2 – Exemple de simplification d'un arbre binaire.

dération les tags grammaticaux dans l'analyse lexicale des réponses formulées par l'utilisateur.

En effet, l'analyse grammaticale permettrait d'encadrer un peu plus le dialogue avec l'utilisateur mais aussi elle apporterait une dimension supplémentaire à l'analyse sémantique, ce qui améliorerait le niveau de compréhension de l'agent intelligent de dialogue.

Bien évidemment, ce gain de précision et de justesse se ferait au détriment de la souplesse de langage que nous avons privilégiée pour notre première version de l'agent intelligent.

Un deuxième gain que peut sans doute apporter l'analyse grammaticale est celui de la personnalisation des questions/réponses de l'agent. En effet, nous pourrions mettre

en place un système d'apprentissage qui apprendrait pour chaque utilisateur sa façon de formuler les phrases. Ce qui nous permettrait de mettre en place un générateur de réponses qui se rapprocherait le plus possible de la manière de parler de l'utilisateur.

Une deuxième perspective serait d'introduire dans notre approche l'utilisation d'une ou plusieurs ontologies métier(s).

Les ontologies auraient tout d'abord comme but d'unifier notre lexique et nos grammaires métiers en une seule entité.

Au sein de cette ontologie nous pouvons citer les composants suivants :

- classes : les classes représenteraient les entités décrites dans notre grammaire métier actuelle ;
- individus : les individus seraient donc les termes de notre lexique actuel ;
- relations : elles décriraient les liens entre les termes (synonymie, antonymie, appartenance à un ensemble flou particulier, etc.) ;
- attributs : ils remplaceraient les tags à la fois sémantiques et grammaticaux.

Une approche fondée sur une ontologie faciliterait également l'extension à la fois du lexique, de la grammaire et de tags puisqu'il suffirait d'étendre l'ontologie soit *via* une autre ontologie soit en y insérant de nouveaux individus, de nouvelles classes ou de nouvelles relations.

Enfin, l'ontologie permettrait de faciliter la gestion de la désambiguïsation sémantique, car un même individu (le terme *proche*, par exemple) pourrait appartenir à plusieurs classes (par exemple *temps* et *espace*) mais les relations qui lui seraient associées permettraient de contextualiser le choix de l'interprétation sémantique qui en serait faite.

Glossaire

API *Application Programming Interface*. 54, 60, 104

BTS Base Transceiver Station. 4

BUM *Basic Unit-interval Monotonic*. 33

CW *Computing with Words*. 16, 43, 45, 87

ELH *Extended Linguistic Hierarchies*. 30, 68, 71

FCL Fuzzy Control Language. 77–79, 95–97, 101

GPS Global Positioning System. 3, 4

GSM *Generalized Symbolic Modifiers*. 35–37, 48, 88

IA Intelligence Artificielle. 1–3

LOWA *Linguistic Ordered Weighted Averaging operator*. 27

MCDM *Multi Criteria Decision Making*. 11, 79

MPG *modus ponens généralisé*. 21

OWA *Ordered Weighted Averaging operator*. 26, 27

PST *Parts of Speech Tagging*. 42, 54, 107

RPG Role Playing Game. 1

TAG *Tree-Adjoining Grammar*. 58–61, 108

TALN Traitement Automatique du Langage Naturel. 2, 38, 49

TF-IDF *Term Frequency-Inverse Document Frequency*. 54

TGQ *Theory of Generalized Quantifiers*. 44

WMA *Weighted Median Aggregation*. 27

WSD *Word Sense Desambiguation*. 41

Index

Bibliographie

[Abchir, 2011] Abchir, M. (2011). A jFuzzyLogic Extension to Deal With Unbalanced Linguistic Term Sets. *ISCAMI*, pages 53–54.

[Abchir et Truck, 2011] Abchir, M.-A. et Truck, I. (2011). Towards a new fuzzy linguistic preference modeling approach for geolocation applications. In *Eurofuse 2011*, volume 107 de *Advances in Intelligent and Soft Computing*, pages 413–424. Springer Berlin Heidelberg.

[Abchir et Truck, 2013] Abchir, M.-A. et Truck, I. (2013). Towards an extension of the 2-tuple linguistic model to deal with unbalanced linguistic term sets. *Kybernetika International Journal*, 49(1) : 164–180.

[Abchir et al., 2012a] Abchir, M.-A., Truck, I., et Pappa, A. (2012a). Dealing with Natural Language Interfaces in a Geolocation Context. In *The 10th International FLINS Conference on Computational Intelligence in Decision and Control*, pages 806–811.

[Abchir et al., 2012b] Abchir, M.-A., Truck, I., et Pappa, A. (2012b). Interpretation of semantically tagged data using fuzzy linguistic 2-tuples. In *IJCCI*, pages 429–432.

[Abchir et al., 2013] Abchir, M.-A., Truck, I., et Pappa, A. (2013). Fuzzy semantics in closed domain question answering. In *Decision Aid Models for Disaster Management and Emergencies*, volume 7, pages 171–188. Springer/Atlantis Press.

[Ambriola et Gervasi, 1997] Ambriola, V. et Gervasi, V. (1997). Processing natural language requirements. In *International Conference on Automated Software Engineering*, page 36, Los Alamitos, CA, USA. IEEE Computer Society.

[Arnauld et Lancelot, 1803] Arnauld, A. et Lancelot, C. (1803). *Grammaire générale et raisonnée de Port-Royal*. Perlet.

[Bartczuk et al., 2012] Bartczuk, L., Dziwiński, P., et Starczewski, J. T. (2012). A new method for dealing with unbalanced linguistic term set. 7267 : 207–212.

[Barwise et Cooper, 1981] Barwise, J. et Cooper, R. (1981). Generalized quantifiers and natural language. *Linguistics and Philosophy*, 4(2) : 159–219.

[Bergmair et Bodenhofer, 2006] Bergmair, R. et Bodenhofer, U. (2006). Syntax-driven analysis of context-free languages with respect to fuzzy relational semantics. In *Proceedings of the 2006 IEEE World Congress on Computational Intelligence (WCCI)*, pages 9647–9654.

[Bloomfield, 1935] Bloomfield, L. (1935). *Language*. Unwin University Books. Allen & Unwin.

[Booth, 1989] Booth, P. (1989). *An Introduction to Human-Computer Interaction*. Lawrence Erlbaum Associates, Publishers, New Jersey, USA.

[Boutilier et al., 2004] Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H., et Poole, D. (2004). CP-nets : A tool for representing and reasoning with conditional *Ceteris Paribus* Preference Statements. *Journal of Artificial Intelligence Research*, 21 : 135–191.

[Brants, 2000] Brants, T. (2000). Tnt - a statistical part-of-speech tagger. *CoRR*, cs.CL/0003055.

[Brill, 1992] Brill, E. (1992). A simple rule-based part of speech tagger. In *ANLP*, pages 152–155.

[Carlsson et Fuller, 2000] Carlsson, C. et Fuller, R. (2000). Benchmarking and linguistic importance weighted aggregations. *Fuzzy sets and systems*, 114(1) : 35–42.

[Cengiz et al., 2010] Cengiz, S., Vedat, T., et A., F. B. (2010). Pneumatic motor speed control by trajectory tracking fuzzy logic controller. *Sadhana*, 35(1) : 75–86.

[Chakravarthy et Jiang, 2009] Chakravarthy, S. et Jiang, Q. (2009). *Stream Data Processing : A Quelity of Service Perspective*. Springer.

[Charniak et al., 1993] Charniak, E., Hendrickson, C., Jacobson, N., et Perkowitz, M. (1993). Equations for part-of-speech tagging. In *AAAI*, pages 784–789.

[Châtel et al., 2010] Châtel, P., Truck, I., et Malenfant, J. (2010). LCP-nets : A linguistic approach for non-functional preferences in a semantic SOA environment. *Journal of Universal Computer Science*, pages 198–217.

[Chomsky, 1965] Chomsky, N. (1965). *Aspects of the Theory of Syntax*. Massachusetts Institute of Technology. M.I.T. Press.

[Chomsky, 1975] Chomsky, N. (1975). *The Logical Structure of Linguistic Theory*. Springer.

[Chomsky, 1986] Chomsky, N. (1986). *Knowledge of language, its nature, origin, and use*. Praeger, New York.

[Cingolani et Alcala-Fdez, 2012] Cingolani, P. et Alcala-Fdez, J. (2012). jfuzzylogic : a robust and flexible fuzzy-logic inference system language implementation. In *Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on*, pages 1–8.

[Cordón et al., 2001] Cordón, O., Herrera, F., et Zwir, I. (2001). Linguistic modeling by hierarchical systems of linguistic rules. *IEEE Transactions on Fuzzy Systems*, 10 : 2–20.

[Costa, 1990] Costa, B. e. (1990). Multiple criteria decision aid : An overview. In *Readings in multiple criteria decision aid*, pages 3–14. Springer-Verlag.

[De Glas, 1989] De Glas, M. (1989). Knowledge representation in a fuzzy setting. Rapport interne 89 48, Université Paris 6.

[Delgado et al., 1993] Delgado, M., Verdegay, J., et Vila, M. (1993). On aggregation operations of linguistic labels. *International journal of intelligent systems*, 8 : 351–370.

[Dienes, 1949] Dienes, Z. (1949). On an implication function in many-valued systems of logic. *J. Symbolic Logic*, 14 : 95–97.

[Diou et al., 2006] Diou, C., Katsikatsos, G., et Delopoulos, A. (2006). Constructing Fuzzy Relations from WordNet for Word Sense Disambiguation. In *Semantic Media Adaptation and Personalization, 2006. SMAP'06. First International Workshop on*, pages 135–140.

[Dong et al., 2013] Dong, Y., Hong, W.-C., et Xu, Y. (2013). Measuring consistency of linguistic preference relations : a 2-tuple linguistic approach. *Soft Computing*, 17(11) : 2117–2130.

[Dubois et Dubois-Charlier, 1997] Dubois, J. et Dubois-Charlier, F. (1997). *Les verbes français*. Larousse.

[Dursun et Karsak, 2014] Dursun, M. et Karsak, E. (2014). An integrated approach based on 2-tuple fuzzy representation and qfd for supplier selection. In Kim, H. K., Ao, S.-I., Amouzegar, M. A., et Rieger, B. B., éditeurs, *IAENG Transactions on Engineering Technologies*, volume 247 de *Lecture Notes in Electrical Engineering*, pages 621–634. Springer Netherlands.

[Espinilla et al., 2011] Espinilla, M., Liu, J., et Martínez, L. (2011). An extended hierarchical linguistic model for decision-making problems. *Computational Intelligence*, 27(3) : 489–512.

[Ferrari et Prince, 2000] Ferrari, S. et Prince, V. (2000). *Création et extension automatiques de dictionnaires terminologiques multi-lingues spécialisés à partir de corpus monolingues*.

[Foucault, 1967] Foucault, M. (1967). La grammaire générale de port-royal. *Langages*, 2(7) : 7–15.

[Glöckner et Knoll, 1997] Glöckner, I. et Knoll, A. (1997). Fuzzy quantifiers for processing natural language queries in content-based multimedia retrieval systems.

[Gonzales et al., 2008] Gonzales, C., Perny, P., et Queiroz, S. (2008). GAI-Networks : Optimization, Ranking and Collective Choice in Combinatorial Domains. *Foundations of computing and decision sciences*, 32(4) : 3–24.

[Green et Rubin, 1971] Green, B. et Rubin, G. (1971). Automated Grammatical Tagging of English. Rapport technique, Department of Linguistics, Brown University.

[Harris, 1951] Harris, Z. (1951). *Methods in structural linguistics*. University of Chicago Press.

[Harris, 1991] Harris, Z. (1991). *A theory of language and information : a mathematical approach*. Clarendon Press.

[Herrera et al., 2008] Herrera, F., Herrera-viedma, E., et Martínez, L. (2008). A fuzzy linguistic methodology to deal with unbalanced linguistic term sets. *IEEE Transactions on Fuzzy Systems*, pages 354–370.

[Herrera et Martínez, 2000] Herrera, F. et Martínez, L. (2000). A 2-tuple fuzzy linguistic representation model for computing with words. *IEEE Transactions on Fuzzy Systems*, 8(6) : 746–752.

[Herrera et Martínez, 2001] Herrera, F. et Martínez, L. (2001). A model based on linguistic 2-tuples for dealing with multigranularity hierarchical linguistic contexts in multiexpert decision making. *IEEE Transactions on Systems, Man and Cybernetics. Part B : Cybernetics*, pages 227–234.

[Huang et al., 2006] Huang, H.-H., Kuo, Y.-H., et Yang, H.-C. (2006). Fuzzy-rough set aided sentence extraction summarization. In *Innovative Computing, Information and Control, 2006. ICICIC'06. First International Conference on*, volume 1, pages 450–453.

[Jacquemin et al., 2002] Jacquemin, B., Brun, C., et Roux, C. (2002). Enriching a Text by Semantic Disambiguation for Information Extraction. In *LREC 2002 Proceedings : Using Semantics for Information Retrieval and Filtering : State of the Art and Future Research*. LREC.

[Kuhn et Mori, 1990] Kuhn, R. et Mori, R. D. (1990). A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6) : 570–583.

[Lafourcade et Prince, 2001] Lafourcade, M. et Prince, V. (Juillet 2001). Synonymies et vecteurs conceptuels. In *TALN 2001*, pages 233–242.

[Larsen, 1980] Larsen, P. (1980). Industrial applications of fuzzy logic control. *International Journal of Man-Machine Studies*, 12 : 3–10.

[Le et al., 2012] Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G., Dean, J., et Ng, A. Y. (2012). Building high-level features using large scale unsupervised learning. In Langford, J. et Pineau, J., éditeurs, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 81–88. ACM.

[Łukasiewicz, 1920] Łukasiewicz, J. (1920). O logice trójwartościowej. (Traduction anglaise : On Three-Valued Logic. In : *Jan Łukasiewicz Selected Works*. L. Borkowski, ed., North Holland, 87–88, 1990). *Ruch Filozoczny*, 5 : 169–171.

[Makki et al., 2008] Makki, J., Alquier, A.-M., et Prince, V. (2008). Ontology Population via NLP Techniques in Risk Management. In *ICSWE : Fifth International Conference on Semantic Web Engineering*, volume 1, pages 79–85.

[Mamdani, 1977] Mamdani, E. (1977). Application of fuzzy logic to approximate reasoning using linguistic synthesis. *IEEE Transactions on Computers*, 26(12) : 1182–1191.

[Mamdani et Assilian, 1975] Mamdani, E. et Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1) : 1–13.

[Martinet, 1962] Martinet, A. (1962). *A functional view of language*. Clarendon Press.

[Martinich, 1985] Martinich, A. (1985). *The philosophy of language*. Oxford University Press.

[Melekhova et al., 2010] Melekhova, O., Abchir, M., Châtel, P., Malenfant, J., Truck, I., et Pappa, A. (2010). Self-Adaptation in Geotracking Applications : Challenges, Opportunities and Models. In *The 2nd International Conference on Adaptive and Self-adaptive Systems and Applications (ADAPTIVE'2010)*, pages 68–77. IEEE.

[Mostowski, 1957] Mostowski, A. (1957). On a generalization of quantifiers. *Fundamenta Mathematicae*, 44 : 12–36.

[Novák, 1991] Novák, V. (1991). Fuzzy Logic, Fuzzy Sets, And Natural Languages. *International Journal of General Systems*, 20(1) : 83–97.

[Novák, 1992] Novák, V. (1992). *The alternative mathematical model of linguistic semantics and pragmatics*. IFSR international series on systems science and engineering. Plenum Press.

[Novák et Perfilieva, 1999] Novák, V. et Perfilieva, I. (1999). Evaluating linguistic expressions and functional fuzzy theories in fuzzy logic. In *Computing with Words in Information/Intelligent Systems 1*, volume 33, pages 383–406. Physica-Verlag HD.

[Orduna et al., 2013] Orduna, R., Jurio, A., Paternain, D., Bustince, H., Melo-Pinto, P., et Barrenechea, E. (2013). Segmentation of color images using a linguistic 2-tuples model. *Information Sciences*, xx : à paraître.

[Pappa, 2006] Pappa, A. (2006). Robust tagging system for lexicon creation. In *Proceedings of the 2006 International Conference on Artificial Intelligence, ICAI 2006, Las Vegas, Nevada, USA, June 26-29, 2006, Volume 2*, pages 711–717.

[Pérez-Asurmendi et Chiclana, 2013] Pérez-Asurmendi, P. et Chiclana, F. (2013). Social choice voting with linguistic preferences and difference in support. In *Aggregation Functions in Theory and in Practise*, pages 249–260. Springer.

[Perlmutter, 1983] Perlmutter, D. M. (1983). *Studies in Relational Grammar 1*. Studies in Relational Grammar. University of Chicago Press.

[Prince, 1994] Prince, V. (1994). Interpreting common words in context : a symbolic approach. In *ECAI*, pages 545–549.

[Prince et Sabah, 1992] Prince, V. et Sabah, G. (1992). Coping with vague and fuzzy words : A multi-expert natural language system which overcomes ambiguities. In *In Acts of PRICAI'92, Seoul*.

[Reichenbach, 1934] Reichenbach, H. (1934). Wahrscheinlichtkeitslogik. *Erkenntnis*, 5 : 37–43.

[Rosson et Carroll, 2009] Rosson, M. B. et Carroll, J. M. (2009). Scenario-based design. *Human-computer interaction : Development process*, pages 1032–1050.

[Ruspini, 1969] Ruspini, H. (1969). A New Approach to Clustering. *Information and Control*, 15 : 22–32.

[Silberztein, 1994] Silberztein, M. (1994). Intex : A corpus processing system. In *COLING*, pages 579–583.

[Singhal, 2012] Singhal, A. (2012). Introducing the knowledge graph : things, not strings. *Official Google Blog, May*.

[Sun et al., 2002] Sun, J., Karray, F., Basir, O., et Kamel, M. (2002). Fuzzy logic-based natural language processing and its application to speech recognition.

[Sung et You, 2009] Sung, W. et You, K. (2009). Adaptive precision geolocation algorithm with multiple model uncertainties. In *Adaptive Control*, volume 1, pages 323–336. In-tech.

[Tisserant et al., 2012] Tisserant, G., Prince, V., et Roche, M. (2012). Détection de relations sémantiques à partir de texte. In *SFC'12 : Société Francophone de Classification*, page 4.

[Truck et Akdag, 2006] Truck, I. et Akdag, H. (2006). Manipulation of qualitative degrees to handle uncertainty : Formal models and applications. *International Journal of Knowledge and Information Systems*, 9(4) : 385–411.

[Truck et Akdag, 2009] Truck, I. et Akdag, H. (2009). A tool for aggregation with words. *International Journal of Information Sciences, Special Issue : Linguistic Decision Making : Tools and Applications*, 179(14) : 2317–2324.

[Truck et al., 2001a] Truck, I., Akdag, H., et Borgi, A. (2001a). A symbolic Approach for Colorimetric Alterations. In *Proceedings of the 2nd International Conference in Fuzzy Logic and Technology (EUSFLAT)*, pages 105–108.

[Truck et al., 2001b] Truck, I., Akdag, H., et Borgi, A. (2001b). Colorimetric Alterations by way of Linguistic Modifiers : A Fuzzy Approach vs. A symbolic Approach. In *Proceedings of the Symposium in International ICSC-NAISO Congress on Computational Intelligence : Methods and Applications (FLA)*, pages 702–708.

[Van Leekwijck et Kerre, 1999] Van Leekwijck, W. et Kerre, E. (1999). Defuzzification : criteria and classification. *FUZZY SETS AND SYSTEMS*, 108(2) : 159–178.

[Vincent et al., 2004] Vincent, B., Pascal, L., et Cyril, M. (2004). Modélisation et intégration de connaissances métier pour l'identification de défauts par règles linguistiques floues. *Traitement du signal*, 21(3) : 227–247.

[Wang et Hao, 2006] Wang, J. et Hao, J. (2006). A new version of 2-tuple fuzzy linguistic representation model for computing with words. *IEEE Transactions on Fuzzy Systems*, 14(3) : 435–445.

[Weizenbaum, 1966] Weizenbaum, J. (1966). Eliza — a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1) : 36–45.

[Winograd, 1972] Winograd, T. (1972). Procedures as a representation for data in a computer program for understanding natural language. *Cognitive Psychology*, 3(1) : 1–191.

[Yager, 1988] Yager, R. (1988). On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *Systems, Man and Cybernetics, IEEE Transactions on*, 18(1) : 183–190.

[Yager, 1993] Yager, R. R. (1993). Families of {OWA} operators. *Fuzzy Sets and Systems*, 59(2) : 125–148.

[Yager, 1998] Yager, R. R. (1998). Fusion of ordinal information using weighted median aggregation. *International Journal of Approximate Reasoning*, 18(1–2) : 35–52.

[Yager et Troiano, 2005] Yager, R. R. et Troiano, L. (2005). On some properties of mixing owa operators with t-norms and t-conorms. In *EUSFLAT Conf.*, pages 1206–1212.

[Yousfi-Monod et Prince, 2007] Yousfi-Monod, M. et Prince, V. (2007). Knowledge Acquisition Modeling through Dialog Between Cognitive Agents. *International Journal of Intelligent Information Technologies*, 3 : 60–78.

[Zadeh, 1996] Zadeh, L. (1996). Fuzzy logic = computing with words. *FuzzySystems, IEEE Transactions on*, 4(2) : 103–111.

[Zadeh, 1965] Zadeh, L. A. (1965). Fuzzy sets. *Information Control*, 8 : 338–353.

[Zadeh, 1971] Zadeh, L. A. (1971). Quantitative fuzzy semantics. *Information Sciences*, 3(2) : 159–176.

[Zadeh, 1972] Zadeh, L. A. (1972). A fuzzy-set-theoretic interpretation of linguistic hedges. *Journal of Cybernetics*, 2(3) : 4–34.

[Zadeh, 1975] Zadeh, L. A. (1975). The concept of a linguistic variable and its application to approximate reasoning, i, ii and iii. In *IS*, volume 8.

[Zadeh, 1978] Zadeh, L. A. (1978). Pruf – a meaning representation language for natural languages. *Int. J. Man-Machine Studies*, 10 : 395–460.

[Zadeh, 1983] Zadeh, L. A. (1983). A computational approach to fuzzy quantifiers in natural languages. *Computers & Mathematics with applications*, 9(1) : 149–184.

Résumé

Dans le domaine du "calcul à l'aide de mots" (CW : *Computing with words*), les approches linguistiques floues ont démontré leur pertinence dans de nombreux problèmes de prise de décision. En effet, elles permettent de modéliser le raisonnement humain en remplaçant les mots, les évaluations, les préférences, les choix, les souhaits, etc. par des variables *ad hoc*, telles que les sous-ensembles flous ou des variables plus complexes.

Dans cette thèse, nous partons d'un problème concret en géolocalisation : la configuration des boîtiers permettant le suivi des mobiles à surveiller et la mise en place des alertes liées au suivi. Il s'agit de mettre en place un système offrant la possibilité de passer des objectifs métiers de l'utilisateur final exprimés sous la forme de paramètres linguistiques (grâce à un dialogue en langage naturel) à une combinaison appropriée des paramètres techniques de l'application. La recherche a visé à montrer comment on peut améliorer une interface utilisateur existante pour passer d'un traitement quantitatif à un traitement qualitatif des occurrences d'événements et des contraintes de géolocalisation. Nous avons ainsi défini les extensions théoriques qui semblaient nécessaires dans le CW : un modèle, fondé sur les *2-tuples sémantiques* que nous introduisons, permet de représenter, avec une grande précision et une grande justesse, des ensembles de termes linguistiques même lorsque ces derniers sont positionnés de façon fortement déséquilibrée sur leur axe. Ces 2-tuples sémantiques ont été mis en œuvre pour interpréter *sémantiquement* les termes linguistiques issus du dialogue, en leur rattachant une sémantique floue contextuelle.

Mots-clés: Linguistique floue, sémantique floue, 2-tuples sémantiques, géolocalisation.

Abstract

In the domain of Computing with words (CW), fuzzy linguistic approaches are known to be relevant in many decision-making problems. Indeed, they allow us to model the human reasoning in replacing words, assessments, preferences, choices, wishes, etc. by *ad hoc* variables, such as fuzzy sets or more sophisticated variables.

In this thesis, we present a new fuzzy representation model to deal with unbalanced linguistic term sets that allow us to handle data with precision and accuracy. This model is based on our fuzzy semantic 2-tuples that we introduce. We apply these semantic 2-tuples to perform a fuzzy semantic interpretation of words in a natural language dialog context for a geolocation application. In this thesis, we start from a concrete geolocation problem : how to configure the devices that track the mobiles and how to set up the alerts related to the tracking ? The idea is to offer the possibility to go from the end-user business-level objectives expressed through linguistic parameters (*via* a natural language dialogue) to an appropriate combination of technical parameters. We show how to improve a user interface to offer a qualitative processing instead of a quantitative one. We define theoretical extensions in the CW framework : a model, based on *semantic 2-tuples* that are introduced, permit to represent linguistic term sets with accuracy and precision, even if they are *very unbalanced*. These semantic 2-tuples can interpret semantically the user's linguistic terms during the dialogue, and attach a contextual fuzzy semantics to them.

Keywords: Fuzzy linguistics, fuzzy semantics, semantic 2-tuples, geolocation.