Département de formation doctorale en informatique **École doctorale IAEM Lorraine**
UFR STMIA

Inférence et traitement automatique des langues

MÉMOIRE

présentée et soutenue publiquement le 21 septembre 2004

pour l'obtention de l'

Habilitation à diriger des recherches
Université Henri Poincaré – Nancy I
(Spécialité Informatique)

par

Claire Gardent

Composition du jury

Président :	Jean-Marie Pierrel, Université Henri Poincaré, Nancy I
Rapporteurs :	Francis Corblin, Université de Paris IV Gérard Huet, INRIA Rocquencourt Patrick Saint-Dizier, CNRS-IRIT
Examinateur :	Christian Rétoré, Université de Bordeaux 1

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thloria.

Remerciements

Le chemin jusqu'à la rédaction de ce mémoire d'HDR a été long, sinueux et passionant ! Je voudrais remercier ici tous les chercheurs, collègues et amis qui m'y ont tenu compagnie.

Tout d'abord, Margaret King, Directrice de l'Institut Dalle Molle pour la Sémantique et les Science Cognitives de Genève. C'est elle qui m'a fait découvrir les joies de ce chemin et je lui en suis pour toujours reconnaissante.

A Essex puis à Edimbourg, Ann de Roeck et Ewan Klein ont guidé mes premiers pas et m'ont aidé à contourner les premiers obstacles, de la maitrîse d'abord, du doctorat ensuite tandis que Gabriel Bès, pourvoyeur de fonds et mentor linguistique, m'initiait aux finesses de l'analyse du Français. C'est à ces trois personnes que je dois ma formation de base de chercheur et je voudrais en particulier les remercier pour l'ouverture interdisciplinaire qu'ils m'ont encouragée à adopter et qui convient parfaitement à mon tempérament de gémeaux.

A Utrecht, Albert Visser m'a donné une première impression de la vision logique de la linguistique tandis qu'à Amsterdam, l'esprit critique, constructif et profondément original de Henk Zeevat m'a permis de progresser sur la question de la relation entre syntaxe, sémantique, phrase et discours.

C'est à Sarrebruck que le chemin s'est soudain peuplé d'informaticiens. Je remercie Michael Kohlhase pour sa collaboration énergique et particulièrement enthousiaste pour toutes les questions alliant linguistique et unification d'ordre supérieur. Denys Duchier m'a fait découvrir l'élégance de la programmation par contraintes. Sa clarté, sa gentillesse, son humour et son extrème intelligence font de lui un co-randonneur qu'on ne peut qu'accepter avec enthousiasme. Enfin les étudiants de Sarrebruck ont la chance de bénéficier d'une éducation de haut calibre en linguistique informatique. Leur compagnie est elle aussi d'un grand plaisir et je remercie en particulier Stefan Thater, Marilisa Amoia et Kristina Striegnitz pour leur diligence, leur ouverture d'esprit et leur indépendance dans leur travail de Hiwi, de maitrise puis de thèse.

A philadelphie puis à Edimbourg, Bonnie Webber a été une source d'inspiration de bien des façons et je me considère privilégiée d'avoir pu travaillé avec elle sur les questions de discours qui nous tiennent toutes deux à coeur.

Enfin Nancy, dernière étape, m'a permis de poser mes bagages et de développer un nouveau paysage. En premier lieu, je tiens à remercier Jean-Marie Pierrel pour m'y avoir attirée puis installée. Je remercie tout aussi chaleureusement Laurent Romary pour l'accueil sympathique et ouvert qu'il a bien voulu me faire au sein de son équipe ainsi que pour le soutien, souriant et continu, qu'il a toujours accordé à mes axes de recherches. Sur les chemins les plus ardus, l'éclat de la jeunesse est toujours un réconfort et c'est avec joie que je remercie les thésards, Marilisa Amoia, Eric Kow, Hélène Manuélian, Yannick Parmentier et Kristina Striegnitz, pour leur travail, leur assiduité et la confiance qu'ils me témoignent.

Sans jury, pas de HDR. Je tiens en dernier lieu à remercier mes rapporteurs et examinateurs : Francis Corblin, Gérard Huet, Patrick Saint-Dizier pour m'avoir fait l'honneur de leurs commentaires détaillés sur ce mémoire ; Christian Rétoré et Jean-Marie Pierrel pour leur présence le jour J ainsi que pour leurs questions pertinentes, inspiratrices de nouvelles réflexions.

Table des matières

Première partie

Introduction

Le traitement automatique des langues (TAL) a pour objet de développer des algorithmes mettant en jeu un savoir linguistique et extra-linguistique pour une application donnée. Plus précisément, il vise à :

1. recueillir les données linguistiques et extra-linguistiques mises en oeuvre par l'activité langagière ;
2. modéliser les faits linguistiques c'est-à-dire prédire les contraintes régissant les interactions entre les données ;
3. définir et optimiser des algorithmes spécialisés (d'analyse, de génération, de résolution anaphorique, etc.) utilisant les données et les modélisations linguistiques (grammaire, contraintes sur les liens anaphoriques, etc.) ;
4. mettre en oeuvre données, contraintes et algorithmes dans des applications informatiques mettant en jeu la langue naturelle. Une application TAL pourra ainsi mettre en oeuvre une modélisation de l'activité langagière dans des domaines tels que la traduction automatique, l'extraction d'information, le résumé automatique ou les systèmes de dialogue homme/machine ;
5. évaluer les résultats obtenus soit par le biais d'une application, soit, plus directement, sur la base d'une suite de tests construite artificiellement ou d'un standard étalon.

Ce mémoire se situe entre les points 2. et 3. ci-dessus. Il se focalise sur la relation entre la sémantique computationnelle, c'est-à-dire le traitement automatique du sens de la langue naturelle et certaines techniques d'inférence développées dans le domaine informatique. Parce que les langues naturelles partagent avec les langues artificielles et en particulier, avec les langages de programmation certaines caractéristiques, il est possible de transposer, moyennant adaptations, certaines représentations et certains algorithmes développés dans le domaine informatique, au domaine de la linguistique informatique. Les techniques d'analyse syntaxique par exemple, se sont au départ largement inspirées des algorithmes d'analyses développés à l'origine pour les langages de programmation et les grammaires pour la langue naturelle, du format utilisé pour décrire la syntaxe des langages de programmation. Il ne faut pas réinventer la roue. Le TAL a besoin d'algorithmes et de procédures à la sémantique claire et au comportement efficace. L'informatique de son coté, vise essentiellement à produire de tels objets. Avant de définir pour la linguistique des opérations et algorithmes spécialisés, il importe de considérer avec attention ce que l'informatique peut apporter à la sémantique computationnelle. Notre travail s'inscrit dans cette perspective et examine en quoi, certaines techniques développées et optimisées avec des objectifs non linguistiques, peuvent être utiles pour la sémantique computationnelle. Nous considérons trois grands types d'inférence :

- l'inférence *structurale* ou *syntaxique* par le biais de l'unification d'ordre supérieure ;
- l'inférence *logique* par le biais des prouveurs de théorèmes et des générateurs de modèles ;
- et l' inférence *ensembliste* par le biais de la programmation basée sur les contraintes.

Pour chacun de ces types d'inférence, nous nous attachons à montrer la pertinence linguistique du calcul impliqué et les applications qui peuvent en être faites dans le domaine du TAL. Nous montrons par exemple, que l'unification d'ordre supérieur est un outil précis et empiriquement bien adapté au traitement interprétatif des énoncés désaccentués et des ambiguïtés lâches/strictes[1]. Ou encore que l'approche par contraintes permet un traitement déclaratif et transparent de la génération des arbres satisfaisant une description – un problème dont nous montrons qu'il a d'importantes applications dans différents domaines du TAL.

[1]Sloppy/strict en anglais.

Plus généralement, le travail présenté dans ce mémoire peut être vu comme un premier pas vers la création d'un «catalogue» de tâches d'inférence en TAL. Il identifie des tâches d'inférence précises, en analyse ou en génération, pour un phénomène linguistique donné ou pour une sous-tâche spécifique ; indique comment ces tâches d'inférence peuvent être modélisées et exécutées dans différents types de cadres de calculs (calcul logique, calcul par contraintes, etc.) ; et spécifie les directions à poursuivre pour une évaluation en taille réelles des propositions faites. A long terme, l'objectif est de permettre l'utilisation et l'évaluation sur des données linguistiques, de techniques et d'outils d'inférence optimisés jusque là sur des données d'ordre différent (les données mathématiques pour les prouveurs de théorèmes ou les problèmes d'ordonnancement pour les contraintes par exemple).

Au cours des dernières décennies, le travail en TAL s'est largement focalisé sur le développement de ressources à grande échelle d'une part et sur la production d'algorithmes de traitement performants et robustes d'autre part.

Pour le recueil des données et la modélisation des contraintes linguistiques, on a ainsi assisté au développement de ressources lexicales et grammaticales de grande taille, soit par des méthodes symboliques et largement manuelles (WordNet, VerbNet, FrameNet), soit par des méthodes statistiques et automatisées (développement de grammaires par des méthodes d'apprentissage par exemple).

Pour l'utilisation des données et la vérification des contraintes impliquées, les algorithmes les plus performants et les plus robustes se basent soit sur des approches statistiques (analyseurs stochastiques, étiquetteurs à la Brill, etc.), soit sur l'utilisation d'automates et de transducteurs d'automates à états finis (plateforme d'analyse XIP de Xerox, analyseurs morphologiques, etc.).

La recherche présentée dans ce mémoire se démarque de ces approches premièrement, en ce qu'elle se situe en aval des travaux portant sur le recueil des données et la modélisation de contraintes linguistiques pour se concentrer sur leur utilisation par des techniques d'inférences et deuxièmement, en ce qu'elle se limite à l'étude de techniques symboliques plutôt que statistiques. Nous élicitons *infra* nos motivations pour ce double positionnement.

Ressources linguistiques *versus* techniques algorithmiques.

Contrairement aux travaux visant le développement de ressources à grande échelle et d'algorithmes performants, le travail présenté dans ce mémoire se restreint à la présentation d'exemples et de ressources «jouets».

Une première raison en est que les ressources sémantiques (bases de connaissances, lexiques sémantiques) et syntaxico-sémantiques (grammaires) nécessaires à une traitement à grande échelle ne sont pas disponibles. Il n'existe pas par exemple, de lexique électronique recensant et structurant le sens des mots d'une langue de façon à ce qu'ils puissent être utilisés par des démonstrateurs de théorèmes[2]. De même, il n'existe pas de grammaire librement disponible pour le français qui associe à toute expression du français une représentation sémantique[3]. Plus généralement le pro-

[2]Wordnet structure des sens de mots mais il ne les recense pas : on ne peut pas, à partir de WordNet, obtenir la définition du sens d'un mot et l'utiliser pour l'inférence. A l'autre extrême, des dictionnaires informatisés tels le TLFI (Trésor de la Langue Française Informatisé) listent bien les définitions lexicographiques mais celles ci développées manuellement par différents linguistes sur un espace de temps relativement long ne sont pas utilisables directement pour l'inférence. Il faudrait pour ce faire, pouvoir les «normaliser» c'est-à-dire les désambiguïser et les traduire dans un langage formel utilisable pour l'inférence. C'est le chat qui se mord la queue : lorsque les définitions du TLFI pourront être normalisées, les problèmes de la sémantique computationnelles seront en grande partie résolus.

[3]Le système XIP de Xerox permet d'associer une représentation sémantique aux expressions du français mais il n'est pas librement disponible.

blème de la représentation du sens véhiculé par les expressions langagières et de sa construction par des techniques informatiques est un problème largement ouvert et que nous n'abordons pas dans le cadre de ce mémoire. Nous revenons cependant sur ce point dans le chapitre final où nous présentons différentes suggestions et pistes de recherche allant dans cette direction.

Une deuxième raison pour la dimension «prototypique» de l'approche présentée concerne la problématique de l'évaluation. Pour évaluer l'adéquation d'une approche, on utilise souvent une ressource étalon annotée pour cette évaluation. Ainsi par exemple, les analyseurs syntaxiques peuvent être évalués à partir de corpus arborés c'est-à-dire de corpus dont les phrases sont annotées avec une structure syntaxique. Pour le type de phénomènes sémantiques considérés dans ce mémoire cependant, les ressources étalons n'existent de nouveau pas. Une alternative possible est celle des suites de test c'est-à-dire, d'ensembles d'exemples annotés, construits systématiquement et manuellement et que l'on peut comparer aux données produites par le système à évaluer. Mais là encore se pose le problème des ressources absentes au sens où de telles suites existent principalement pour l'évaluation d'analyseurs syntaxiques. Enfin on pourrait envisager évaluer le comportement sémantique d'un système par une évaluation «directe» c'est-à-dire par l'évaluation des performances du système par rapport à une tâche réelle comme par exemple, la tâche de question/réponse. Cependant ce type d'évaluation pose la question de l'évaluation componentielle du système (quel(s) module(s) du système est(sont) responsable(s) d'une erreur ?). Elle nécessite également la construction d'un système entier de TAL ce qui dépasse largement les objectifs de ce mémoire.

Enfin, de même que nous n'avons pas tenté une évaluation à grande échelle, nous n'avons pas poursuivi le développement d'algorithmes performants – par manque de temps, par manque de compétence. L'adaptation et l'optimisation d'un algorithme à des fins spécifiques (par exemple, l'adaptation d'un algorithme d'unification d'ordre supérieur au processus de reconstruction sémantique impliqué par l'interprétation de l'ellipse) exige des connaissances informatiques solides et une compréhension en profondeur de l'algorithme étudié. C'est à notre avis, une travail typiquement interdisciplinaire pour lequel les meilleurs résultats passent par une collaboration entre linguistes et informaticiens.

Approche symbolique *versus* approche statistique.

Plus que le développement de ressources à large échelle ou d'outils performants, ce mémoire vise donc une modélisation symbolique des processus sous-tendant le traitement sémantique de la langue naturelle. Pour ce domaine spécifique, les méthodes statistiques nous semblent peu appropriées et leur utilisation prématurée. Ainsi, si l'on peut aujourd'hui développer des analyseurs syntaxiques performants sur la base de techniques stochastiques, c'est là le fruit indirect de longues années de recherche en syntaxe. Pour atteindre à un niveau d'expertise comparable dans le domaine de la sémantique, il nous semble essentiel d'avoir une réflexion approfondie sur les outils et les représentations nécessaires à une modélisation adéquate du sens véhiculé par l'activité langagière. Quelle est la signature adaptée a la représentation de ce sens ? Quelles sont les procédures de calcul les plus appropriées pour construire le sens d'une phrase ? d'un dialogue ? d'un texte ? Quelles sont les inférences nécessaires à la reconstruction des implicites ? Tel est le type de questions auxquelles il nous semble important de trouver des réponses pour poser les fondements d'un traitement sémantique «profond» de la langue naturelle.

Ce mémoire apporte une contribution allant dans ce sens. Il identifie de façon précise des phénomènes linguistiques (ambiguïté, désaccentuation, etc.) et des tâches de TAL (lexicalisation, génération, etc.) pour lesquels l'inférence (syntaxique, logique ou ensembliste) est pertinente. Il montre par ailleurs, comment ces techniques d'inférence peuvent être mise en jeu pour une modélisation idoine des phénomènes et tâches considérées. Enfin il met en avant les perspectives nouvelles offertes par ces approches «inférentielles» au regard des techniques concurrentes.

Ainsi par exemple, le chapitre sur les contraintes montre *inter alia* comment les contraintes ensemblistes peuvent être utilisées pour modéliser l'inverse de l'analyse syntaxique, c'est-à-dire, le processus de réalisation de surface (comment à partir d'une représentation sémantique, construire la ou les phrases associée(s) par la grammaire considérée à cette représentation sémantique). Puis, il indique, via une comparaison avec les approches concurrentes, en quoi l'utilisation de ce mécanisme ouvre de nouvelles perspectives sur ce problème. La proposition a une portée restreinte en soi mais elle illustre bien l'intérêt des contraintes ensemblistes pour la langue naturelle : déclarativité et transparence des analyses ; possibilité d'exprimer des contraintes locales (par exemple, à un arbre lexical) ou globales (par exemple, exclusion de deux extractions longue distance dans une même phrase). C'est le type d'approche poursuivie de façon plus générale dans le traitement par les contraintes des grammaires de dépendances de Denys Duchier ou encore dans les grammaires des propriétés de Gabriel Bès et Philippe Blache. C'est aussi celui qui est utilisé actuellement à Nancy pour le développement d'un compilateur de grammaires facilitant l'écriture en particulier, de grammaires d'arbres adjoints avec une dimension sémantique. En bref, le traitement des descriptions d'arbres basé sur les contraintes a de multiples applications en TAL et permet soit une utilisation directe dans des applications *offline* (le compilateur de métagrammaire par exemple) ou optimisées (les grammaires de dépendances de Denys Duchier), soit une utilisation indirecte par le biais de résultats formels et de transpositions dans des cadres de traitement plus efficace (traitement des descriptions par des algorithmes de traitement de graphes par exemple).

Plan du mémoire

Ce mémoire comporte trois parties, une pour chacun des types d'inférence examinés : inférence syntaxique, inférence logique et inférence par contraintes. Les trois techniques ont en commun qu'elles sont toutes appliquées à la représentation sémantique de la langue naturelle. Elle diffèrent dans la façon dont l'inférence est utilisée et dans l'objectif visé.

Partie I – Inférence syntaxique

L'inférence que nous qualifions de «syntaxique» est une inférence qui permet, à partir de la comparaison de deux structures, d'en inférer une troisième. L'unification en est un exemple prototypique puisque que la comparaison de deux structures par l'opération d'unification aboutit soit à la production d'un unificateur le plus général, soit à un message d'erreur (les deux structures ne sont pas unifiables).

Dans le domaine de la sémantique computationelle, une application naturelle de l'inférence syntaxique concerne la comparaison de la structure syntaxique de deux représentations sémantiques. Naturellement pour que l'opération fasse sens (c'est-à-dire, pour qu'elle ne produise pas systématiquement un message d'erreur), il faut que les représentations comparées soit comparables et plus spécifiquement, il faut que ces représentations diffèrent uniquement dans leur degré de spécification, l'une étant plus spécifiée que l'autre ou encore le degré de spécification relatif des deux représentations étant variable d'une partie à l'autre de la représentation. D'un point de vue linguistique, cela signifie que l'on utilisera l'unification pour comparer les représentations sémantiques de deux énoncés semblables sémantiquement typiquement, un énoncé elliptique ou anaphorique et son antécédent ou encore deux énoncés «parallèles» au plan sémantique. Dans les deux cas, l'unification sert à la fois à vérifier le parallélisme sémantique de deux propositions et à «résoudre» des représentations sous spécifiées par le jeu des instanciations de variables libres. Suivant les cas et les analyses, la vérification du parallélisme sera plus ou moins complexe, potentiellement triviale pour une analyse de l'ellipse ou relativement complexe pour le traitement d'un énoncé désaccentué faisant intervenir ellipses et anaphores.

Dans la première partie de ce mémoire, nous utilisons l'unification d'ordre supérieur pour modéliser la relation de redondance sémantique reliant un énoncé désaccentué à son énoncé antécédent. Prenant comme point de départ l'analyse unificationnelle proposée pour l'ellipse du groupe verbal par D'andrinople, Shieber et Pereira, nous commençons par modifier cette analyse pour pallier deux inconvénients à savoir, la non-compositionalité de l'analyse et sa sur-génération. Nous montrons ensuite comment cette analyse modifiée permet de rendre compte des contraintes régissant l'interprétation des énoncés désaccentués et en particulier, des interactions entre désaccentuation, anaphores nominales, opérateurs liés au focus, quantification, définis et indéfinis. Enfin, nous élargissons l'analyse au champ des ambiguïtés strictes/lâches arguant que la contrainte proposée pour traiter les énoncés désaccentués est en fait une contrainte réglant l'interprétation des énoncés dits «parallèles» et à laquelle l'ambiguïté stricte/lâche est soumise. Nous montrons que les prédictions empiriques faites par cette analyse sont confirmées par les données et en particulier, qu'elle rend compte adéquatement, des différents types d'ambiguïtés stricte/lâche existantes ainsi que de leur interaction avec les phénomènes de liage et de quantification.

Partie II – Inférence logique

L'inférence logique est sans doute celle la plus largement évoquée lorsque l'on parle d'inférence dans le TAL. C'est l'inférence basée sur la sémantique des connecteurs logiques (la conjonction, la disjonction, l'implication, la négation, la quantification existentielle ou universelle, etc.) et dont le calcul permet de déterminer la validité d'une proposition (cette proposition est-elle vraie dans tous les modèles ?) ou encore une relation de conséquence entre deux propositions (la vérité de cette proposition entraîne-t-elle systématiquement la vérité de cette autre proposition ?).

Les travaux TAL des années 70 dans ce domaine avaient une approche très globale au sens où l'objectif poursuivi était la modélisation des processus de compréhension impliqués par l'activité langagière. Ainsi les travaux de Schank par exemple, visait une modélisation de la compréhension impliquée dans la lecture d'un texte : le système développé devait pouvoir répondre à des questions arbitraires sur le contenu sémantique d'un texte (des histoires pour enfants).

Pour qu'une approche aussi globale ait des chances de succès, il faudrait que plusieurs grands problèmes de recherche soient résolus. Dans le cadre d'une approche symbolique, il faudrait en particulier savoir calculer la représentation sémantique d'un texte y compris ses présuppositions et ses implicatures ; pouvoir modéliser l'ensemble des connaissances encyclopédiques et lexicales mises en jeu par le processus de compréhension ; et disposer de prouveurs suffisament efficaces pour traiter les masses de données impliquées dans des délais raisonnables.

Parce que ces problèmes sont loin d'être résolus, le travail présenté dans ce mémoire vise une approche plus locale qui consiste à isoler des tâches d'inférence nécessitant des connaissances extra-linguistiques restreintes, à évaluer leur utilité pour le traitement des langues naturelles et à explorer les modalités de leur intégration dans un système de TAL. Nous explorons ainsi deux grands types d'inférence logique : l'inférence basée sur la logique des prédicats et celle basée sur les logiques des descriptions.

Dans le cadre de la logique des prédicats, nous explorons l'utilisation de l'inférence pour la Désambiguïsation d'expressions telles que les anaphores (nominales et relationnelles) et le réciproque «each other». Contrairement au cas de sous-spécification représenté par les énoncés elliptiques ou désaccentués, le type d'ambiguïté exploré dans cette deuxième partie nécessite non pas la reconstruction d'un sens laissé implicite, mais plutôt un choix entre plusieurs interprétations possibles. C'est en outre sur la base du contexte linguistique et extra-linguistique (connaissances lexicales et encyclopédiques) que l'interprétation privilégiée pourra être identifiée. Nous montrons que pour déterminer l'interprétation privilégiée de ces expressions, trois critères doivent être utilisés conjointement : celui de consistance, celui d'informativité et celui de minimalité. Nous montrons en outre qu'une implémentation de ces critères passe par l'utilisation conjointe (pour la consistance et l'informativité) ou isolée (pour la minimalité) de démonstrateurs de théorèmes et de générateur de modèles.

Mais comme le montre la discussion des exemples, si les requêtes d'inférence soulevées par les données examinées sont relativement spécifiques, l'interaction des trois critères de désambiguïsation avec le contexte linguistique et extra-linguistique nécessite une modélisation très fine et surtout, des connaissances encyclopédiques qui peuvent être assez générales. Une interaction complexe entre contexte linguistique et résolution de l'ambiguïté rend en outre difficile la conception d'une suite de tests en vue d'une évaluation systématique des performances des outils d'inférence sur ce type de problème.

Nous nous tournons donc dans un deuxième temps vers des problèmes linguistiques où l'inférence logique s'appuie sur des connaissances principalement lexicales et où les requêtes d'inférence peuvent être facilement calculées à partir des données. C'est cet objectif qui est poursuivi dans la partie portant sur les logiques des descriptions en traitant de la lexicalisation d'une part et des divergences lexicales entre question et réponse d'autre part.

La lexicalisation – qui est une sous-tâche de la tâche de génération – consiste à sélectionner, à partir d'un ensemble de faits et d'une base de connaissances, l'ensemble des concepts lexicalisés évoqués par ces faits. Dans la littérature et les systèmes de génération existants, la lexicalisation est généralement traitée comme une tâche de «filtrage» (matching) prenant en compte la relation de subsomption définie par l'hyperonymie dans une taxinomie donnée. Les faits donnés en entrée doivent être partitionnés de façon à ce que chaque élément de cette partition corresponde à une définition de concept dans la taxinomie. Et la correspondance est caractérisée par une comparaison de structures (en l'occurrence des graphes) ne prenant pas en compte la sémantique des connecteurs logiques.

Nous argumentons qu'un traitement adéquat de la lexicalisation passe par l'inférence logique. En effet, la tâche de lexicalisation vise à identifier parmi un ensemble de faits, soit des concepts simples, soit des définitions lexicales. Or un définition lexicale d'une part, peut être arbitrairement complexe (elle peut en particulier contenir des connecteurs booléens tels que la disjonction ou la négation) et d'autre part, est généralement récursive au sens où elle fait intervenir des termes qui sont eux mêmes définis. Identifier à partir d'un ensemble de faits, les définitions lexicales impliquées est donc une tâche non pas de filtrage ou, dans les termes employés ici, d'inférence syntaxique, mais bien plutôt d'inférence logique : la subsomption (l'implication) et la sémantique des connecteurs booléens doivent pouvoir être prises en compte. Nous montrons comment implémenter un traitement logique de la lexicalisation dans le cadre des logiques de descriptions et indiquons comment intégrer ce traitement avec une algorithme de réalisation permettant de produire, à partir, d'un ensemble de concepts, une expression verbalisant ces concepts.

Afin de pouvoir évaluer le module de lexicalisation ainsi proposé, il est nécessaire de disposer d'une base de connaissances répertoriant des sens lexicaux. Dans cette optique, nous développons actuellement une base de connaissances lexicales répertoriant les sens prédictibles sur la base des données morphodérivationnelles (par exemple, que «conducteur» réfère à «une personne participant comme agent dans une éventualité de conduite») et à partir de laquelle il sera possible d'examiner et d'évaluer les résultats obtenus par l'approche proposée en particulier, en ce qui concerne sa flexibilité et la variabilité des sorties. Les données utilisées par un générateur peuvent en effet avoir un degré de granularité très variable si bien que la flexibilité est un critère important. L'utilisation de la taxinomie devrait permettre d'identifier des équivalences entre faits et concepts à différents niveaux de la hiérarchie et ainsi d'offrir à la fois une meilleure flexibilité dans l'utilisation du module de lexicalisation et une plus grande variabilité dans la verbalisation.

Outre son utilité pour la lexicalisation, notons qu'une base de sens lexicaux basée sur les connaissances morphodérivationnelles peut également être utile pour des applications telles que les systèmes question/réponse. En effet, pareille base recense par le biais des équivalences entre concepts, un savoir paraphrastique entre mots ou syntagmes. Or comme le montre les dernières évolutions des systèmes de question/réponse, le savoir paraphrastique est un facteur essentiel dans le bon comportement des systèmes de question/réponse.

Pour la tâche de lexicalisation, le sens lexical est essentiel. Mais comme nous l'avons remarqué *supra* (cf. note de bas de page 2), définir un répertoire des sens lexicaux (un lexique informatisé) est une tâche ardue et même très probablement «linguistique-difficile». Afin d'élargir la base de connaissances basée sur les sens lexicaux identifiables par le biais de la morphologie dérivationnelle, nous nous tournons donc dans un deuxième temps vers un domaine lexical moins épineux que celui des sens lexicaux à savoir, le domaine des relations lexicales classiques. Ces relations (synonymie, hyperonymie, troponymie, antonymie) sont d'un accès plus facile – elles sont largement encodées dans le WordNet anglais par exemple. Elle sont également plus faciles à établir puisque basiques plutôt que complexes. Dans la partie de ce mémoire traitant des divergences lexicales entre question et réponse, nous montrons que ces relations sont pertinentes pour la tâche de question/réponse et spécifions la tâche d'inférence nécessaire à un traitement adéquat de ces divergences proposant ainsi une application permettant d'évaluer le degré d'adéquation de la modélisation du savoir lexical encodé.

En résumé, notre travail sur l'utilisation des logiques de descriptions tend à identifier le type de savoir lexical dont la modélisation et l'utilisation informatique est envisageable ainsi qu'à préciser des tâches de TAL faisant intervenir ce type de savoir et permettant une évaluation des ressources développées.

Partie III – Inférence ensembliste

Comme son nom l'indique, l'inférence ensembliste se base sur la sémantique des relations ensemblistes (intersection, union, complément, etc.) pour déduire d'un ensemble de faits ensemblistes (valeurs d'ensembles et relations entre ensembles), de nouveaux faits.

Au plan informatique, l'inférence ensembliste peut être implémentée par le biais de variables et de contraintes ensemblistes, dans le cadre de la programmation par contraintes. Ainsi le langage de programmation Oz/Mozart permet l'utilisation de variables et de contraintes ensemblistes à partir desquelles toutes les opérations ensemblistes peuvent être exprimées.

Dans les années 80, Martin Kay puis Stuart Shieber et Fernando Pereira ont prêché l'utilisation des grammaires d'unification et plus généralement de l'unification en TAL comme fournissant un outil particulièrement «propre» pour la modélisation des faits syntaxiques. Leur argument reposait sur deux grands points. Premièrement, l'unification et plus précisément, les variables d'unification permettent une représentation compacte des faits linguistiques. L'exemple typique à cet égard, était celui de la régle d'accord entre verbe et sujet qui pouvait être formulée par une règle unique dans le cas des grammaires d'unification mais exigeait une prolifération des règles dans le cas des grammaires ne permettant pas le partage de variables. Deuxièmement, l'unification permet une analyse «déclarative» c'est-à-dire, une analyse où le linguiste n'a pas à se soucier de l'ordre dans lequel les données seront traitées par l'algorithme de traitement (analyseur, réalisateur, etc.). Les vertus des grammaires d'unification se sont rapidement imposées et les années

80 ont vu le développement de grammaires d'unification, d'environnements de développement pour ces grammaires et d'analyseurs spécialisés.

Issue de la programmation logique, l'approche par contraintes bénéficie d'avantages similaires aux traitements basés sur l'unification. Elle permet d'une part la sous-spécification par le biais de variables portant sur des domaines finis et favorise d'autre part, déclarativité et transparence. Lorsqu'un problème a une formulation ensembliste naturelle, cette formulation trouve en effet une implémentation directe dans un langage de programmation tel que Oz si bien qu'il est possible de formuler un problème en termes de contraintes sans avoir à prendre en compte la façon dont ces contraintes seront évaluées.

Dans la troisième partie de ce mémoire, nous considérons les inférences permises par les contraintes ensemblistes et leur utilité pour le TAL. Nous identifions deux grands types d'applications : d'une part, la représentation compacte d'arbres et leur traitement et d'autre part, la construction du contenu sémantique d'une description définie.

La représentation compacte de plusieurs arbres par le biais en particulier, d'une description d'arbres laissant certains aspects structuraux indéterminés, a plusieurs applications en linguistique. Elle est en particulier souvent utilisée pour la sous spécification de structures syntaxiques (cf. les Description Tree Grammars de Owen Rambow) et de structures sémantiques (sous spécification de la portée d'un opérateur, d'un quantificateur ou d'un modificateur). Dans les chapitres 9, 10, nous montrons d'abord que les contraintes ensemblistes permettent une modélisation relativement directe des contraintes régissant la relation entre descriptions d'arbres et arbres satisfaisant ces descriptions. Nous proposons ensuite différentes applications de cette technique au TAL. Premièrement, nous montrons que les descriptions d'arbres permettent un traitement «cognitif» des ambiguités interprétatives discursives. En effet, les ambiguités sémantiques apparaissant au niveau des relations entre deux phrases sont du même type que celles que l'on trouve dans l'analyse syntaxique de la phrase : certaines sont traitées déterministiquement, d'autres provoquent en cas d'erreur un délai dans l'interprétation et le sentiment clair de devoir réviser la première interprétation obtenue, d'autres enfin (les ambiguités dites «globales») se projettent au niveau de la phrase. Nous montrons qu'une modélisation par les contraintes ensemblistes permet à la fois de représenter de façon compacte ces différents types d'ambiguïté et de simuler le processus humain dans un traitement incrémental.

Une deuxième application des descriptions d'arbres et de leur traitement au TAL concerne le traitement syntaxique avec les grammaires basées sur les descriptions d'arbres. Sur ce point, nous montrons que la programmation par contraintes permet une modélisation réversible du processus de traitement avec une grammaire : modulo quelques modifications mineures concernant les phases d'initiation et de vérification, le même algorithme d'évaluation des contraintes peut être utilisé à la fois pour produire une représentation sémantique à partir d'un texte (analyse) et pour produire un texte à partir d'une représentation sémantique (génération).

Enfin les contraintes ensemblistes peuvent être utilisée pour modéliser la production de descriptions définies en génération. Nous montrons comment implémenter un solveur s'acquittant de cette tâche et explicitons en quoi l'analyse proposée permet un nouvel angle d'approche sur le problème.

Deuxième partie

Inférence syntaxique

Sous le terme d'*inférence syntaxique*, nous désignons tout calcul dont les règles d'inférence reflètent un savoir purement formel des structures considérées. Ainsi l'unification est un exemple d'inférence syntaxique au sens où les règles d'inférence qu'elle utilise font intervenir un savoir exclusivement syntaxique sur la forme des termes qu'elle permet de comparer – par opposition, nous qualifierons d'*inférence sémantique*, l'inférence logique mise en oeuvre par un prouveur de théorèmes parce que celle ci met en jeu des règles d'inférence reflétant la sémantique des connecteurs et quantificateurs logiques.

[39] (DSP) montre qu'une analyse de l'ellipse du groupe verbal (GV) basée sur l'unification d'ordre Supérieur (UOS) est intéressante pour deux raisons. D'une part, elle a un bon pouvoir explicatif au sens où elle rend compte d'un nombre important d'observations concernant l'interaction entre l'ellipse du groupe verbal, les phénomènes de quantification et la pronominalisation. D'autre part, elle appuie la construction sémantique de l'ellipse (i.e. la construction de la formule logique représentant le sens d'une ellipse tel qu'il est déterminé par le contexte) sur un outil, l'UOS, qui est bien compris tant au plan mathématique qu'au plan informatique. Au plan mathématique, l'UOS a une définition précise qui permet une délimitation exacte des limites et des prédictions de l'approche. Au plan informatique, l'UOS a une interprétation aussi bien déclarative que procédurale, ce qui permet de préserver la transparence et facilite l'implémentation.

Dans les chapitres qui suivent, nous poursuivons l'entreprise amorcée par DSP. Nous commençons (chapitre 1) par modifier leur analyse remédiant ainsi à deux problèmes théoriques sur lesquels cette analyse achoppe : la non-compositionalité et la sur-génération. Nous montrons ensuite que cette modification de l'analyse initiale permet d'élargir son champ d'application. Elle permet en particulier de modéliser les contraintes sémantiques qui régissent l'interprétation non seulement de l'ellipse verbale mais également des énoncés désaccentués (Chapitre 2) et des anaphores à interprétation lâche (Chapitre 3).

Chapitre 1

L'analyse unificationnelle

Parce que les langues naturelles véhiculent le sens par des moyens extrèmement divers (sens littéral ou «assertion», mais aussi anaphores, ellipses, deixis, implicatures, présuppositions etc.), calculer le sens d'une expression de la langue naturelle par des moyens automatiques est une tâche non triviale. Pour le niveau assertionnel néanmoins, les travaux de Montague [139, 140] montrent comment associer de façon systématique une signification à une expression langagière[4]. Plus précisément, ces travaux montrent comment construire le sens littéral d'une expression langagière indépendamment des effets contextuels (résolution des pronoms ou des présuppositions par exemple).

L'approche unificationnelle bâtit sur la proposition de Montague et l'étend au traitement de l'ellipse en la combinant avec l'utilisation de l'unification d'ordre supérieure et d'un mécanisme de stockage pour les quantificateurs.

1.1 Langage et sens

L'être humain utilise le langage en relation avec le monde qui l'entoure soit pour le décrire, soit pour poser une question sur son état soit encore pour émettre une instruction ayant pour objet de le modifier. Les théories du sens du langage diffèrent par l'importance qu'elles accordent aux différents éléments du triangle langage-locuteur-monde et par le statut qu'elles leur accordent.

Une première école, représentée entre autre par Wittgenstein [198] dans son tractatus logico-philosophicus, part de l'observation que le langage humain sert à décrire le monde réel. Dans cette approche, le locuteur se représente des faits[5] et cette représentation est un modèle de la réalité[6]. La relation entre ce modèle et la réalité est une relation «géométrique» :

> What a picture must have in common with reality, in order to be able to depict it – correctly or incorrectly – in the way it does, is its pictorial form[7]

Dans le Tractatus logico-philosophicus, le modèle est une abstraction représentant le monde et la relation entre modèle et réalité reste implicite.

Une deuxième école se fonde sur l'observation que nous utilisons le langage pour exprimer nos pensées. Dans cette école, le but central d'une théorie du sens est d'identifier le langage

[4]Même pour le niveau assertionnel, l'approche de Montague est loin de rendre compte de la «créativité sémantique» du langage. Nous reviendrons sur ce point dans le chapitre final.

[5]«We picture facts to ourselves», 2.1

[6]«A picture is a model of reality», 2.12.

[7]Trad. Ce qu'une image doit avoir en commun avec la réalité afin de la décrire – correctement ou incorrectement – de la façon dont elle le fait, est sa forme graphique.

conceptuel (language of thoughts) mis en oeuvre. C'est l'approche adoptée dans le domaine de la linguistique générative au début des années 70. Suivant Katz et Fodor [107], les linguistes adoptent pendant cette période l'idée d'un langage conceptuel universel et s'attachent à montrer comment la syntaxe permet de construire ces représentations conceptuelles à partir d'un ensemble de traits sémantiques. Dans cette approche, les représentations calculées servent à évaluer et à comparer le sens des expressions du langage humain : Une expression est-elle sémantiquement anormale ? ambiguë ? Deux expressions sont-elles dans une relation de paraphrase ? d'antonymie ? Telles sont pour l'approche conceptuelle les questions d'intérêt. Les représentations conceptuelles ne sont pas interprétées et la relation au monde, ou même à un modèle abstrait, n'y joue aucun rôle.

Dans chacune de ces deux approches, le rôle du locuteur et plus généralement de la situation d'élocution n'est pas pris en compte. Un troisième courant philosophique place locuteur et situation d'élocution au centre de la problématique. C'est l'approche représentée successivement par Searle [170], Austin et Grice [77] où l'usage qui est fait du langage est un élément essentiel de la théorie du sens. Le sens d'une expression n'y est pas déterminé uniquement par le sens des mots la constituant. Il prend également en compte les présuppositions, les implicatures et les actes de paroles résultant de l'usage en contexte de cette expression.

La notion de sens adoptée dans ce mémoire se situe dans la lignée du Tractatus logico-philosophicus de Wittgenstein. Nous adoptons en effet une approche inspirée du traitement de Montague. Or celui-ci, en développant un calcul sémantique dans le cadre du lambda calcul explicitement typé, concrétise les «images» de Witgenstein : ce sont les modèles dans lequel les termes de ce lambda calcul sont interprétés.

1.2 L'approche de Montague

Dans l'approche de Montague, le sens d'une expression langagière est déterminée par ses conditions de vérité, c'est-à-dire par les conditions qui doivent être vérifiées pour que cette expression soit évaluée comme étant «vraie» dans un modèle donné. Ainsi par exemple, la phrase «Gabrielle joue avec Sarah» sera vraie dans un modèle qui contient deux individus nommés Gabrielle et Sarah et tel que ces deux individus jouent ensemble. Pour une logique du premier ordre par exemple, un modèle est une paire $\langle D, I \rangle$ où D est un ensemble d'objets (le domaine de discours) et I est une fonction d'interprétation assignant à chaque expression suivant son arité : un objet, un ensemble d'objets ou un ensemble de tuplets d'objets. Ainsi, le nom propre «Gabrielle» pourra être associé par la fonction d'interprétation à un objet dans le domaine de discours tandis que le verbe «joue» sera associé avec un ensemble de paires, c'est-à-dire avec une relation binaire sur les éléments du domaine de discours.

C'est dans ce cadre vériconditionnel et basé sur la théorie des modèles, que Montague développe un calcul sémantique permettant d'associer à chaque expression de la langue naturelle une représentation sémantique. Le langage de représentation sémantique qu'il utilise est une logique intensionnelle d'ordre supérieure extrêmement expressive qui inclut une hiérarchie de types (avec des expressions pour chacun des types), la quantification d'ordre supérieur (variables et quantificateurs pour chaque type), l'abstraction lambda pour tous les types, des opérateurs modaux et temporels et le moyen de former pour toute expression soit son extension, soit son intension. Comme les phénomènes traités dans ce mémoire ne font intervenir ni les contextes intensionnels, ni les contextes modaux ou temporels, nous présentons ici une version simplifiée de l'approche Montagovienne, une version basée sur un Lambda-calcul typé non intensionnel, inspiré du Lambda-calcul utilisé par Montague et que nous appellerons Λ_M. Avant de présenter ce langage de représentation sémantique, nous introduisons les notions de base du Lambda-calcul typé [28] nécessaires à la compréhension de l'approche unificationnelle.

1.2.1 Le Lambda-calcul typé

Dans le Lambda-calcul typé, chaque expression a un type unique. L'ensemble des types T est défini à partir d'un ensemble fini de types élémentaires T_0 :

1. $T_0 \in T$
2. Si $\alpha, \beta \in T$ alors $(\alpha \to \beta) \in T$

Si A est l'ensemble des objets de type a et B l'ensemble des objets de type β, alors $\alpha \to \beta$ est le type des fonctions dont le domaine est A et l'image B.

Les **termes** du Lambda-calcul sont composés d'atomes, d'applications et d'abstractions. Abstraction et application permettent de créer puis d'appliquer des fonctions.

1. Pour chaque type α, on a un ensemble V_α de variables et un ensemble C_α de constantes. Ces ensembles sont disjoints et définissent l'ensemble $\mathcal{A}$ des atomes :

$$\mathcal{A} = \cup_{\alpha \in T}(C_\alpha \cup V_\alpha)$$

2. Si M est un terme de type $(\alpha \to \beta)$ et si N est un terme de type α, alors l'**application** $(M\ N)$ est de type β.
3. Si M est un terme de type β et $x \in V_\alpha$, alors l'**abstraction** $\lambda x.M$ est de type $(\alpha \to \beta)$.

L'ensemble des termes du Lambda-calcul est le plus petit super-ensemble de $\mathcal{A}$ obtenu par fermeture sous l'application et l'abstraction.

L'ensemble des variables libres d'un terme M, noté $FV(M)$, est défini par récurrence de la façon suivante :

1. $FV(x) = \{x\}$ si x est une variable
2. $FV(MN) = FV(M) \cup FV(N)$
3. $FV(\lambda x.M) = FV(M) - \{x\}$

Pour tous termes M, N, x, on note $[M/x]N$ le résultat de la **substitution** par M de x dans N, pour toutes les occurrences libres de la variable x et en renommant les variables liées de façon à éviter des conflits. Plus précisément :

1. $[M/x]x = M$
2. $[M/x]a = a$ pour tout atome $a \neq x$
3. $[M/x](PQ) = ([M/x]P[M/x]Q)$
4. $[M/x](\lambda x.P) = \lambda x.P$
5. $[M/x](\lambda y.P) = \lambda y.[M/x]P$ si $y \neq x$ et $y \notin FV(M)$ ou $x \notin FV(M)$
6. $[M/x](\lambda y.P) = \lambda z.[M/x][z/y]P$ si $y \neq x$ et $y \in FV(M)$ et $x \in FV(P)$ et z est une nouvelle variable.

L'égalité entre deux termes fait intervenir différents types de réductions :

Réduction α : $\lambda x.M \rhd_\alpha \lambda y.([y/x]M)$ – le nom d'une variable liée est sans importance

Réduction β : $(\lambda x.M)(N) \rhd_\beta [N/x](M)$ – l'application d'une fonction $(\lambda x.M)$ a un argument N consiste à substituer toutes les occurrences de x dans le corps de la fonction M par cet argument

Dans le cadre d'une théorie extensionnelle des fonctions $(\forall x(f(x) = g(x)) \rightarrow f = g)$, la réduction η est également valide :

$$\lambda x.M(x) \rhd_\beta M \text{ si x} \notin FV(M)$$

Un terme a une forme normale s'il ne contient aucun redex, c'est-à-dire aucun terme de la forme $\lambda x.MN$. Pour tout terme M, il est possible de dériver de M par une série finie (et peut être vide) de réductions et/ou d'expansions β et/ou α un terme N en forme normale. Le théorème de Church-Rosser montre en outre que cette forme normale est unique modulo α conversion.

Ceci conclut notre brève introduction du Lambda-calcul typé. Pour plus de détails, voir [83].

1.2.2 Un langage pour la représentation sémantique

Le langage de représentation sémantique Λ_M utilisé dans ce mémoire est un Lambda-calcul explicitement typé issu de la théorie des types simple de Church [28]. Sa syntaxe repose sur l'ensemble de types suivant :

1. t est un type, le type des propositions logiques
2. e est un type, le type des individus du domaine de base sous-jacent
3. Si α et β sont des types, alors $\alpha \rightarrow \beta$ est un type

Pour chaque type α, Λ_M comprend un ensemble de variables Var_α et un ensemble de constantes Con_α. Les termes de Λ_M sont les suivants :

1. Pour chaque type α, toutes les constantes et toutes les variables de type α sont des termes de type α
2. Pour toute paire de types α, β, si M est un terme de type $\alpha \rightarrow \beta$ et si N est un terme de type α, alors $M(N)$ est un terme de type β
3. Pour toute paire de types α, β, si x est une variable de type α et si N est un terme de type β, alors $\lambda x.N$ est un terme de type $\alpha \rightarrow \beta$
4. Si ϕ et ψ sont de type t, alors $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \rightarrow \psi$ et $\neg\psi$ sont des termes de type t
5. Si ϕ est de type t et si x est une variable alors $\forall x.\phi$ et $\exists x.\phi$ sont des termes de type t

L'ensemble D_α de dénotations possibles pour les termes de type α est défini comme suit :

1. D_e est D, le domaine de discours
2. D_t est $\{0, 1\}$, les valeurs de vérité
3. Pour toute paire de types α, β, $D_{(\alpha \rightarrow \beta)}$ est l'ensemble des fonctions de D_α vers D_β

La sémantique de Λ_M peut être donnée par différents types de modèles selon l'ontologie et la logique considérées.

Le choix de la logique codera «en dur» certaines décisions. Ainsi une logique classique, parce qu'elle inclut la loi des tiers exclus, imposera une évaluation binaire selon laquelle une proposition est soit vraie, soit fausse. Par contraste, une logique intuitioniste en excluant la loi des tiers, permet de modéliser l'aspect partiel de l'information (un fait est vrai, faux ou indéterminé). Une logique dynamique postule une vue relationnelle du sens permettant de refléter le flux d'information d'un texte ou d'un discours – et en particulier, l'information pertinente pour le traitement des anaphores. Enfin une logique linéaire pose des contraintes fortes sur l'utilisation des hypothèses, contraintes qui se sont révélées utiles dans le traitement de la syntaxe de la langue naturelle par exemple [162].

Le choix de l'ontologie, c'est-à-dire du type des individus présents dans le domaine de base, permet de moduler les effets du choix de la logique. Une ontologie riche permet en particulier de passer outre certaines limitations de la logique classique.

Prenons par exemple l'aspect partiel de l'information. Dans une logique classique, les phrases (1.1a) et (1.1b) sont équivalentes : si l'une est vraie, l'autre l'est aussi et inversement.

(1.1) a. Jean marche.
 b. Jean marche et Pierre chante ou ne chante pas.

Par compositionalité, les phrases (1.4a) et (1.4b) seront elles aussi équivalentes :

(1.2) a. Marie voit Jean marcher.
 b. Marie voit Jean marcher et Pierre chanter ou ne pas chanter.

Ce qui est contraire à l'intuition. En effet, nos ressources sont limitées et nous agissons dans une partie relativement petite du monde. Il est donc fort possible que Marie voit Jean sans voir Pierre. Dans ce cas et si Jean marche, (1.4a) est vrai. Mais que dire de (1.4b) ? Il semble que dans une situation où Marie voit Jean marcher et ne voit pas Pierre, l'analyse la plus intuitive est celle qui assigne à (1.4b) une valeur de vérité indéterminée : la proposition n'est dans ce cas ni vrai ni fausse mais plutôt indéterminée.

A première vue, de tels exemples, et plus généralement l'aspect partiel de l'information dont nous disposons, suggèrent qu'une logique classique est inadaptée pour le traitement de la langue naturelle. Mais pareille conclusion serait trop hâtive et comme [125, 144] l'ont montré il est souvent possible de développer une logique partielle qui rende compte des faits linguistiques et ait une traduction dans une logique classique. En particulier, [144] spécifie une version partielle de la grammaire de Montague qui capture adéquatement la nature partielle de notre mécanisme interprétatif. Cette version partielle repose sur l'utilisation d'un lambda calcul multi-valué dont Muskens montre qu'il peut être enchâssé dans la théorie des types utilisée par Montague.

Un autre exemple illustrant la relation entre logique et ontologie est celui de la dynamicité. Au début des années 80, Hans Kamp et Irene Heim [103, 82] proposent un traitement sémantique des anaphores inter-phrastiques et des indéfinis qui repose sur la notion de contexte discursif. Ces traitements ont trouvé des formulations aussi bien dans des logiques classiques [103, 145] que dans des logiques dite «dynamiques» inspirées de la logique de Hoare pour la sémantique de programmes. [145] montre en particulier comment une ontologie riche permet de réconcilier une sémantique dynamique où le sens est vu comme une relation entre contexte avec une logique classique incluant outre les individus et les propositions d'une logique «standard», des états et des fonctions sur ces états.

1.2.3 Lexique et grammaire : le calcul sémantique

Dans la grammaire de Montague, le **lexique** associe à chaque mot de la langue naturelle une catégorie syntaxique, un type sémantique et un Lambda-terme. Les catégories syntaxiques, inspirées des grammaires catégorielles de Adjukiewicz, sont définies de la façon suivante :

1. t est une catégorie syntaxique
2. e est une catégorie syntaxique
3. Si A et B sont des catégories syntaxiques, alors A/B et $A//B$ sont des catégories syntaxiques

Intuitivement, une constituent de catégorie A/B (ou $A//B$) est un constituent qui, lorsqu'il est combinée avec un constituent de catégorie B forme un constituant de catégorie A. Cette définition récursive des catégories syntaxiques facilite la mise en correspondance entre catégories syntaxiques et types sémantiques, correspondance qui est définie par une fonction f de l'ensemble des catégories syntaxiques vers l'ensemble des types sémantiques comme suit[8] :

1. $f(t) = t$
2. $f(e) = e$
3. Si A et B sont des catégories syntaxiques, alors $f(A/B) = f(A//B) = \langle f(B), f(A)\rangle$.

Les deux types de catégories A/B et $A//B$ permettent de rendre compte du fait que des constituants de catégories linguistiques distinctes peuvent avoir des dénotations de même type sémantique. Par exemple, nom commun et verbe intransitif dénotent tous deux une propriété, c'est-à-dire un objet de type $e \to t$. Pour refléter à la fois leur ressemblance sémantique et leurs différences syntaxiques, Montague attribue au nom propre la catégorie $t//e$ et aux verbes intransitifs la catégorie t/e. Notons que le slash n'est pas directionnel si bien qu'une catégorie de type A/B sera appropriée à la fois pour un constituant combinant avec un constituant de catégorie B placé à sa droite et pour un constituant cherchant cet argument à sa gauche.

Nous adoptons ici et dans le reste du mémoire une approche plus standard et présupposons un lexique où les catégories syntaxiques sont celles utilisées par les linguistes : PN pour *Proper Nouns*, Det pour *Determiner*, V_i pour *Verbe Intransitif* et V_t pour *Verbe Transitif*[9].

Mot	Lambda-terme	Type	Catégorie
John	$\lambda P.P(j)$	$(e \to t) \to t$	PN
Every	$\lambda P.\lambda Q.\forall x(P(x) \to Q(x))$	$(e \to t) \to ((e \to t) \to t)$	Det
man	$\lambda x.M(x)$	$(e \to t)$	N
runs	$\lambda x.R(x)$	$(e \to t)$	V_i
likes	$\lambda O \lambda x.O(\lambda y.L(x,y))$	$((e \to t) \to t) \to (e \to t)$	V_t

De même que le lexique associe à chaque mot, un Lambda-terme, la grammaire associe à chaque règle syntaxique, une règle sémantique. Si l'on transpose l'approche de Montague à une grammaire hors contexte standard, on aura par exemple les paires de règles suivantes :

[8] Rappelons que nous ne prenons pas en compte ici les phénomènes intensionnels. Lorsque, comme c'est le cas dans l'approche Montagovienne, l'intensionnalité est prise en compte la fonction f associe aux catégories syntaxiques des types intensionnels, c'est-à-dire des fonctions d'un ensemble d'indices (ou mondes possibles) vers l'ensemble de dénotations approprié.

[9] Parce que le travail de recherche présenté dans ce mémoire s'est fait sur l'anglais, nous laissons les exemples dans leur version initiale et conservons les catégories syntaxiques utilisés par les linguistes pour cette langue.

	Régles	Exemple
1.	NP → Det N	NP(Det(Every),N(man))
	NP' = Det'(N')	$\lambda P.\lambda Q.\forall x(P(x) \to Q(x))(\lambda x.M(x))$
		$= \lambda Q.\forall x(M(x) \to Q(x))$
2.	NP → PN	NP(PN(John))
	NP' = PN'	$\lambda P.P(j)$
3.	VP → V NP	VP(V(likes),NP(PN(John)))
	VP' = V'(NP')	$\lambda O \lambda x.O(\lambda y.L(x,y))(\lambda P.P(j))$
		$= \lambda x.L(x,j)$
4.	S → NP VP	S (NP(Det(Every),N(man)),VP(V(likes),NP(PN(John)))
	S' = NP'(VP')	$\lambda Q.\forall x(M(x) \to Q(x))(\lambda x.L(x,j))$
		$= \forall x(M(x) \to L(x,j))$

Nous ignorons pour l'instant les problèmes de portée liés aux quantificateurs. Nous y reviendrons dans la section qui suit.

1.3 L'analyse unificationnelle de DSP

L'analyse unificationnelle de l'ellipse verbale proposée par DSP bâtit sur l'approche de Montague et l'étend au traitement des ellipses par le biais d'équations qui sont résolues par unification.

1.3.1 L'unification

La tâche d'unification consiste à trouver, étant donné deux descriptions x et y, un objet z qui satisfait ces deux descriptions[10]. Cette tâche est généralement considérée dans le cadre d'une logique donnée et reformulée de la façon suivante :

> Etant donné deux termes M et N, est il possible de trouver une substitution qui rende M et N égaux ?

Par exemple, les termes $f(x,y)$ et $f(g(y,a),h(a))$ sont unifiables parce qu'en remplaçant x par $g(h(a),a)$ et y par $h(a)$, on obtient deux termes identiques.

Une **substitution** est une fonction d'un ensemble de variables vers un ensemble de termes qui peut être décrite par un ensemble d'assignations. Par exemple, la substitution σ qui associe x à $g(h(a),a)$ et y à $h(a)$ sera notée $\{x \leftarrow g(h(a),a), y \leftarrow h(a)\}$. Une substitution est *aussi* une fonction d'un ensemble de termes vers un ensemble de termes par l'intermédiaire de son application. L'application de la substitution σ au terme M s'écrit $\sigma(M)$ et dénote la forme normale du terme tel que chaque variable x_i dans M est remplacée par $\sigma(x_i)$.

Deux termes M et N sont unifiables s'il existe une substitution σ telle que $\sigma(M) = \sigma(N)$. Si c'est le cas, σ est un **unificateur** de M et N et $\sigma(M)$ est appelée, l'**unifié** de M et de N. Deux termes M et N sont dans une relation de **filtrage** s'il existe une substitution σ telle que $M = \sigma(N)$

[10]Pour une introduction aux utilisations de l'unification, voir [114].

L'unificateur σ de deux termes M et N est **l'unificateur le plus général** (UPG) de M et N si pour tout autre unificateur θ de M et N, il existe une substitution τ telle que $\tau\sigma = \theta$. Intuitivement, l'UPG de deux termes est le plus simple de tous leurs unificateurs.

Pour la logique du premier ordre, il existe un UPG unique pour chaque paire de termes unifiables et le problème d'unification est décidable [163].

Pour les logiques d'ordre supérieur en revanche, [93] montre que l'unification pour les logiques du troisième ordre est un problème indécidable (il n'existe pas de procédure qui permette de déterminer si deux termes du troisième ordre sont ou non, unifiables). Dans le cas où deux termes sont unifiables, on sait en outre [76] qu'il n'existe pas nécessairement d'UPG unique. Néanmoins, [150] présente un algorithme d'unification pour le lambda calcul typé qui permet d'énumérer étant donnée une équation solvable, toutes les solutions de cette équation tandis que [94] présente un algorithme moins redondant retournant, pour toute équation solvable, un pré-unificateur. L'algorithme de Huet est en outre simplifié si, comme nous en faisons l'hypothèse ici, la règle de η réduction est admise.

1.3.2 Ellipse et unification

Analyse de base

L'analyse unificationnelle de DSP [39] repose sur deux composantes :

1. des représentations sémantiques faisant intervenir des variables d'ordre supérieur et
2. des équations permettant d'établir la valeur de ces variables soit par unification, soit par filtrage.

Le langage de représentation sémantique utilisé est celui présenté en section 1.2.2 et la notion d'égalité admise par l'algorithme d'unification est celle d'égalité $\alpha\beta\eta$: deux termes seront considérés égaux ssi ils peuvent être rendus identiques par une série de réductions et/ou d'expansions (réductions inverses) α, β et/ou η .

L'exemple suivant illustre le fonctionnement de cette analyse.

(1.3) John loves Sarah. Peter does too.

La représentation sémantique et l'équation associées par l'analyse unificationnelle à cet énoncé sont :

Représentation	$love(j,s) \wedge R(p)$
Équation	$R(j) = love(j,s)$

où R représente la sémantique manquante dans la **phrase cible**, c'est-à-dire la phrase contenant l'ellipse. Pour l'équation donnée, l'UOS retourne la substitution[11] :

$$\{R \leftarrow \lambda x.love(x,s)\}$$

ce qui permet de reconstruire la sémantique de la phrase cible comme suit :

$$\lambda x.love(x,s)(p) \equiv_\beta love(p,s)$$

Plus généralement, l'analyse proposée par DSP repose sur une représentation sémantique et sur une équation que l'on peut schématiser comme suit :

[11] Comme nous le verrons plus tard, l'UOS retourne outre la solution indiquée, d'autres solutions qui d'un point de vue linguistique ne sont pas valides. Ce problème est discuté plus en détail dans la section 1.5 où une solution est proposée pour éviter les substitutions inadéquates.

(1.4)

Représentation	$S \wedge R(C_1, \ldots, C_n)$
Équation	$R(S_1, \ldots, S_n) = S$

où S est la représentation sémantique de la **proposition source** (la proposition contenant l'antécédent de l'ellipse), $S_1, \ldots, S_n$ et $C_1, \ldots, C_n$ sont les représentations sémantiques des éléments parallèles sources et cibles respectivement et R représente la relation commune aux deux propositions. Il est important de noter que DSP ne prétendent pas résoudre le problème de la détermination du parallélisme et que la définition de ce qu'est un élément parallèle est laissée ouverte. Intuitivement, un élément parallèle source/cible est un constituent réalisé (non implicite) de la phrase source/cible qui a une contrepartie, également réalisée, dans la phrase cible/source. Ainsi dans l'exemple 1.3 ci-dessus, *John* est un élément parallèle source dont la contrepartie dans la phrase cible est *Pierre* – et *vice versa*, *Pierre* est un élément parallèle cible.

Interaction avec les anaphores nominales

L'interaction entre le traitement unificationnel de l'ellipse proposé par DSP et la résolution des anaphores peut se concevoir de deux façons.

Une première possibilité est de considérer les contraintes équationnelles comme plaçant une contrainte additionnelle sur la façon dont une anaphore peut être résolue. Dans cette optique, la résolution des anaphores se fait de la manière habituelle tandis que l'UOS élimine les résolutions incompatibles avec les équations posées par l'approche unificationnelle.

Une seconde possibilité est de représenter les anaphores inter-phrastiques comme les ellipses, c'est-à-dire par le biais de variables libres d'un type approprié. La valeur de ces variables serait déterminée, comme pour l'ellipse, par unification. Cette stratégie, bien qu'élégante, présente plusieurs inconvénients. Tout d'abord, elle impose une distinction assez artificielle entre anaphores intra- et inter-phrastiques. Par ailleurs, elle ne permet pas de tirer profit des résolveurs spécialisés existants (cf. par exemple l'algorithme de résolution des anaphores pronominales de [135]). Enfin, et surtout, il n'est pas clair comment cette approche pourrait prendre en compte les contraintes discursives ou d'accord morphologique et sémantique qui régissent la relation anaphore/antécédent.

Dans l'article [39], l'approche implicitement adoptée par DSP est la première et c'est ici aussi, cette première approche qui sera présupposée.

Interaction avec les phénomènes de portée

Un point fort de l'approche unificationnelle présentée par DSP est qu'elle couvre un éventail relativement large d'interactions entre ellipses, anaphores pronominales et ambiguïtés de portée. Bien que ces dernières soient ici peu discutées, nous présentons pour complétude, l'approche proposée par DSP pour en rendre compte. Dans cette approche, l'interprétation d'une phrase est représentée par une formule de la forme $\Gamma \models m$ où Γ est un ensemble d'hypothèses et m est un terme dans lequel des variables libres introduites par Γ peuvent apparaître.

Une **hypothèse** est un triplet de la forme $\langle q\ x\ p \rangle$ où q représente l'interprétation d'un déterminant, x est une variable libre et p un terme propositionnel contenant la variable libre x. L'hypothèse $\langle q\ x\ p \rangle$ *introduit* la variable x. Un syntagme nominal quantifié est en outre interprété comme une variable (e.g., x) introduite par une hypothèse dont le premier élément (q) représente l'interprétation de son déterminant et le troisième (p) l'interprétation de son groupe nominal. Enfin pour qu'une dérivation soit complète, il faut que toutes les hypothèses aient été utilisées. Nous illustrons le fonctionnement de cette analyse par l'exemple *Every person left.* Le syntagme nominal (SN) quantifié *every person* est représenté par

$$\langle \text{ every x person(x)} \rangle \models x$$

L'interprétation du syntagme verbal appliquée à celle du SN donne l'interprétation

$$\langle \text{ every x person(x)} \rangle \models \text{ left(x)}$$

Utiliser l'hypothèse consiste à appliquer le quantificateur *every* à sa restriction et à sa portée. Pour des raisons techniques discutée dans [39], cette application est faite sur une Lambda-abstraction de la conjonction de la restriction et de la portée (le type de *every* est ici $((e \rightarrow t \times t) \rightarrow t))$ et on obtiendra ainsi :

$$\models \text{ every } (\lambda x. \langle \text{ person(x), left(x)} \rangle \text{)}$$

Ce traitement de la quantification – inspiré du mécanisme de stockage de Cooper – interagit avec la résolution de l'ellipse au sens où les équations à résoudre peuvent être calculées sur les représentations obtenues avant ou après l'utilisation des hypothèses. Suivant le cas, on aura ainsi différentes valeurs pour l'ellipse. C'est cette interaction qui permet de rendre compte de certaines des ambiguïtés interprétatives résultant de l'interaction entre ellipse, quantificateurs et pronoms [173].

Évaluation

Comme nous l'avons mentionné plus haut, le mérite de l'analyse unificationnelle est son bon pouvoir prédictif et sa large couverture empirique. En particulier :

- l'analyse prédit (plutôt qu'elle ne stipule) les ambiguïtés interprétatives résultant de l'interaction entre ellipse verbale et anaphores pronominales ([39]) ;
- elle rend compte du fait que le domaine déterminant l'interprétation d'une ellipse est un domaine déterminé par le parallélisme plutôt que par la syntaxe. En particulier, elle rend compte du fait que cette interprétation peut être déterminée par un domaine plus large que celui de la phrase minimale contenant l'antécédent de l'ellipse ([39]) ;
- lorsqu'elle est associée à un traitement des ambiguïtés de portée, elle prédit correctement les interactions entre ellipse, quantification et anaphores pronominales ([173]).

Néanmoins, cette analyse n'est pas sans défaut. Telle qu'elle est présentée dans DSP, l'analyse unificationnelle souffre en effet de deux inconvénients :

- elle ne fait pas le lien avec une construction sémantique à la Montague ;
- elle sur-génère, c'est-à-dire qu'elle produit des solutions qui, bien que mathématiquement correctes, ne sont pas acceptables au plan linguistique.

Les deux sections qui suivent ont pour objet d'élucider ces problèmes et de leur apporter une solution. La section 1.4 propose une modification des équations utilisées qui rend l'analyse compatible avec une construction sémantique à la Montague et qui, de surcroît, permet d'étendre son champ d'application aux énoncés désaccentués (chapitre 2) et aux interprétations dites «lâches» (en anglais «sloppy») (chapitre 3). La section 1.5 propose une solution au problème de la sur–génération en s'appuyant sur l'utilisation d'une variante de l'UOS développée dans la communauté informatique pour guider la stratégie de recherche des prouveurs automatiques : l'unification Colorée d'ordre Supérieur (UCOS).

1.4 Analyse unificationnelle et compositionalité

Dans l'analyse de DSP, l'idée clé est que l'interprétation d'une phrase elliptique met en jeu d'une part, la reconstruction d'une relation à partir de la phrase source et d'autre part, l'application de cette relation aux représentations sémantiques des éléments parallèles de la phrase cible. Cette analyse soulève la question suivante :

> *Comment cette procédure de reconstruction sémantique s'articule-t-elle avec le processus de construction sémantique standard qui depuis les travaux de Montague, est d'usage dans les grammaires informatiques de la langue naturelle telles que les grammaires syntagmatiques généralisées (Generalised Phrase Structure Grammar [74]), les grammaires lexicales fonctionnelles (Lexicalised Phrase Structure Grammar [106]), les grammaires catégorielles (Categorial Grammar [182]) et les grammaires guidées par les têtes (Head-driven Phrase Structure Grammar [152]) ?*

Et plus spécifiquement :

> *Quelle est la relation entre la représentation de la phrase elliptique qui peut être dérivée des équations de l'analyse unificationnelle et celle qui est produite par un processus de construction sémantique standard ?*

Prenons un exemple concret :

(1.5) John runs but Peter does not.

Dans l'hypothèse d'un processus de construction à la Montague où toute catégorie syntaxique correspond à un type sémantique donné, le type sémantique d'une ellipse du groupe verbal est $e \rightarrow t$, le type des propriétés d'individus. La représentation de la phrase cible produite par un module standard de construction sémantique sera donc[12] (les indices indiquent le type de l'expression) :

$$neg_{t \rightarrow t}(R_{e \rightarrow t}(peter_e))$$

Considérons maintenant la représentation de la phrase cible stipulée par l'analyse de DSP. Conformément à la représentation donnée ci-dessus, la représentation de la phrase source pour l'exemple (1.5) est celle donnée en (1.6a). Les éléments parallèles source étant *pos* et *john*, la représentation de la phrase cible sera celle donnée en (1.6b). Si, comme il l'est indiqué, les noms propres dénotent des éléments de type e et la négation, une fonction de type $t \rightarrow t$, alors le type de la variable R est celui donné en (1.6c).

(1.6) a. $pos_{t \rightarrow t}(run_{e \rightarrow t}(john_e))$
b. $R(pos_{t \rightarrow t})(john_e)$
c. $R_{(t \rightarrow t) \rightarrow e \rightarrow t}$

En résumé, alors qu'un module de construction sémantique standard assigne à une ellipse de GV une variable R de type $e \rightarrow t$, la représentation de l'ellipse qui peut être déduite de l'analyse DSP est une fonction de type $(t \rightarrow t) \rightarrow e \rightarrow t$. Il n'est donc pas possible de combiner l'analyse unificationnelle de l'ellipse telle qu'elle est formulée par DSP avec une grammaire dont l'interface syntaxe/sémantique adopte l'hypothèse compositionnelle de Montague. Cette incompatibilité devient criante si, comme le suggèrent DSP, on tente d'élargir l'analyse unificationnelle à un champ d'application plus étendu que celui de l'ellipse :

[12] La représentation indiquée suppose que la négation porte sur l'ensemble de la phrase. Il arrive que la portée de la négation se limite au groupe verbal. Dans ce cas, la représentation serait $(neg_{((e \rightarrow t) \rightarrow (e \rightarrow t))}(R_{(e \rightarrow t)}))(peter_e)$. Peu importe. Le point pertinent ici est que la variable représentant le groupe verbal est de type $(e \rightarrow t)$.

> [...] many other elliptical phenomena and related phenomena subject to multiple readings akin to the strict and sloppy readings discussed here may be analysed using the same techniques [39, page 450][13].

La terminologie «sloppy» (que nous traduirons ici par «lâche»[14]) et «stricte» a été introduite par [166]. L'**ambiguïté lâche/stricte** fait intervenir deux phrases. La première de ces phrases, la *phrase source*, contient une anaphore que nous appellerons *l'anaphore source*. La deuxième phrase, la phrase *cible* contient soit une anaphore, soit une ellipse que nous désignerons par le terme de *proforme cible*. Enfin l'interprétation de la proforme cible dépend de l'interprétation de l'anaphore source *in situ*, c'est-à-dire au point du discours où elle apparaît. Par exemple, dans le texte 1.7, la phrase source est *John loves his cat*, la phrase cible *Peter does too*, l'anaphore source *his* (qui dans la phrase source réfère à *John*) et la proforme cible est l'ellipse *does* dont l'interprétation dépend de l'interprétation de *his*.

(1.7) John^1 loves his$_1$ cat and Peter does too.
a. $love(j)(cat(j)) \wedge love(p)(cat(j))$ Strict
b. $love(j)(cat(j)) \wedge love(p)(cat(p))$ Lâche

L'interprétation de la proforme cible est dite **stricte** si elle préserve l'interprétation initiale de l'anaphore source, c'est-à-dire si l'interprétation de l'anaphore source est la même dans la phrase source et dans la phrase cible. Elle est dite **lâche** lorsque l'anaphore source est «ré-interprétée» dans la phrase cible et résolue à l'élément cible qui est parallèle à l'antécédent de l'anaphore source dans la phrase source. Ainsi l'interprétation (1.7a) est stricte : l'anaphore source *his* est interprétée dans la phrase source et dans la phrase cible comme référant à *John*. En revanche, l'interprétation (1.7a) est lâche au sens où l'anaphore source *his* est réinterprétée dans la phrase cible comme référant à l'élément parallèle cible de l'antécédent source *John*, c'est-à-dire à *Peter*.

Revenons à la question de la compositionalité dans l'analyse unificationnelle de l'ellipse. Dans l'exemple (1.8) ci-dessous, la différence entre la représentation sémantique postulée par DSP et celle générée par un module de construction sémantique est très nette.

(1.8) I'll help you if you want. I'll kiss you even if you don't.
a. Interprétation stricte : *Je t'aide si tu veux que je t'aide. Je t'embrasse même si tu ne veux pas que je t'aide.*
b. Interprétation lâche : *Je t'aide si tu veux que je t'aide. Je t'embrasse même si tu ne veux pas que je t'embrasse.*

Le verbe *help* étant l'élément parallèle source, l'équation postulée par l'analyse de DSP est :

$$R(help) = want(you, help(i, you)) \rightarrow help(i, you)$$

En conséquence, la représentation sémantique de la phrase cible *Je t'embrasse même si tu ne le veux pas* devra être :

$$R(kiss)$$

Cette représentation est très éloignée de ce que l'on pourrait attendre d'une construction sémantique standard comme par exemple :

$$\neg want(R(you)) \rightarrow kiss(i,you)$$

[13] Beaucoup d'autres phénomènes elliptiques et apparentés sujets à des interprétations multiples semblables aux interprétations strictes et lâches discutées ici peuvent être analysés en utilisant les mêmes techniques.
[14] Terme suggéré par Gérard Huet.

L'élément manquant est une contrainte disant que la représentation postulée par DSP pour la phrase cible doit pouvoir être unifiée avec la représentation sémantique générée pour cette phrase par le module de construction sémantique. Si l'on intègre cette contrainte, les représentations et contraintes postulées sont les suivantes :

(1.9)

Représentation	$S \wedge C$
Equations	$R(S_1, \ldots, S_n) = S$
	$R(C_1, \ldots, C_n) = C$

Comme en 1.4, S est la représentation sémantique de la **proposition source**, $S_1, \ldots, S_n$ et $C_1, \ldots, C_n$ sont les représentations sémantiques des éléments parallèles sources et cibles respectivement et R représente la relation commune à la proposition source et à la proposition cible. L'élément nouveau est C qui représente la représentation sémantique générée pour la phrase cible par le processus de construction sémantique. La seconde équation exige donc que la représentation C construite par le module de calcul sémantique puisse être unifiée avec la représentation stipulée pour la cible par DSP à savoir $R(C_1, \ldots, C_n)$.

Dans le cadre de cette adaptation, l'exemple (1.5) est ré-analysé comme suit.

(1.10) a. $pos(run(john)) \wedge neg(P_{e \to t}(peter))$
b. $R_{(t \to t) \to e \to t}(pos)(john) = pos(run(john))$
c. $R(neg)(peter) = neg(P(peter))$
d. $\{R \leftarrow \lambda O \lambda x.O(run(x)), P \leftarrow \lambda x.run(x)\}$
e. $\lambda O \lambda x O(run(x))(neg)(peter) \equiv neg(run(peter))$

La représentation sémantique de (1.5) est (1.10a) où la représentation de la phrase cible est celle produite par un module de construction sémantique à la Montague et où en particulier, l'ellipse du GV est représentée par une variable (P) du type approprié à savoir le type $e \to t$. Les équations sont celles données en (1.10b-c) où R représente la sémantique commune aux deux phrases et P l'ellipse du groupe verbal. L'UOS donne la solution indiquée en (1.10d) : la valeur de R dénote la relation commune aux deux structures tandis que la valeur de P est un Lambda-terme représentant l'interprétation de l' ellipse.

Il est ainsi possible en modifiant légèrement l'analyse de DSP d'élucider la relation entre le traitement unificationnel de l'ellipse et la sémantique compositionnelle de la phrase. En outre, il devient clair que l'approche de DSP peut de fait être généralisée à d'autres phénomènes que l'ellipse du groupe verbal. Le point clé est que l'analyse met maintenant en jeu non pas une, mais deux variables libres; et que si la variable R représente toujours la sémantique commune aux deux phrases, les variables libres apparaissant dans la représentation sémantique de la phrase cible peuvent représenter toutes sortes d'anaphores du discours – pas uniquement une ellipse. Par exemple, si l'on a le discours suivant[15] :

(1.11) A : John^1 drove his$_1$ wife to the station.
B : No, PETER [drove his wife to the station.]

Bien qu'il n'y ait pas d'ellipse dans la phrase cible, l'énoncé est ambigu entre une interprétation stricte et une interprétation lâche[16]. L'analyse unificationnelle peut en rendre compte puisque seules les deux résolutions du pronoms cible *his* correspondant aux interprétations strictes et lâches permettent de résoudre les équations postulées par cette analyse :

[15]Comme il est d'usage, les indexes et exposants marquent la coréférence.

[16]Les parenthèses carrées délimitent la partie désaccentuée de la phrase cible. Comme nous le verrons dans le chapitre 2, il est clair que dans de tels cas, l'ambiguïté du pronom *sa* est contrainte par le parallélisme c'est-à-dire qu'elle est de fait une ambiguïté lâche/stricte plutôt qu'une simple ambiguïté de résolution.

	Stricte	Lâche
Repr.	$drove(j, wife(j), s)$ $\wedge drove(b, wife(j), s)$	$drove(j, wife(j), s)$ $\wedge drove(b, wife(b), s)$
Eqns	$R(j) = drove(j, wife(j), s)$ $R(b) = drove(b, wife(j), s)$	$R(j) = drove(j, wife(j), s)$ $R(b) = drove(b, wife(b), s)$
Sols.	$\{R \leftarrow \lambda z.drove(z, wife(j), s)\}$	$\{R \leftarrow \lambda z.drove(z, wife(z), s)\}$

Plus généralement, nous verrons dans les chapitres 2 et 3 que la modification que nous proposons pour l'analyse de DSP permet d'étendre sa couverture empirique aux énoncés désaccentués d'une part, et à une théorie généralisée de l'interprétation lâche d'autre part.

D'un point de vue calculatoire, la modification proposée n'est pas anodine. En effet, l'introduction d'une variable additionnelle fait que les équations posées ne sont plus des problèmes de *filtrage* comme c'est généralement le cas dans l'approche proposée par DSP mais bien des problèmes d'unification. Or la complexité de la procédure de filtrage est moindre que celle de l'unification. On sait par exemple, que si le problème d'unification d'ordre supérieur n'est pas décidable, celui du filtrage pour un Lambda-calcul du troisième ordre l'est [45].

En pratique cependant, il est possible de contourner le problème en résolvant les deux équations posées de façon successive plutôt que simultanée. Comme la première équation $R(S_1, \ldots, S_n) = S$ est l'équation de DSP, ce sera une équation de filtrage relativement simple à résoudre. La résolution de cette équation produira en outre la ou les valeur(s) de la variable R apparaissant dans la deuxième équation $R(C_1, \ldots, C_n) = C$ la réduisant ainsi elle aussi à une équation de filtrage.

1.5 Le problème de la sur-génération

Nous considérons maintenant le deuxième problème rencontré par l'analyse de DSP, celui de la sur-génération. On sait (cf. [39]) qu'une approche unificationnelle «naïve» entraîne une sur-génération systématique. Le problème peut être illustré par l'exemple suivant. Supposons que l'on ait le texte (1.12) et que la valeur de l'ellipse verbale soit déterminée par l'équation (1.12a). L'algorithme de Huet donne non pas une, mais deux solutions : (1.12b) et (1.12c).

(1.12) John loves Mary. Peter does too.
 a. $R(j) = love(j, m)$
 b. $\{R \leftarrow \lambda x.love(x, m)\}$
 c. $\{R \leftarrow \lambda x.love(j, m)\}$

Il est clair qu'au plan linguistique seule la première solution (1.12b) est acceptable puisque (1.12c) aurait pour résultat d'assigner comme représentation sémantique à la proposition *Peter does too* le λ–terme $\lambda x.love(j, m)(p)$ qui par β réduction est équivalent au terme $love(j, m)$. En d'autres termes, *Peter does too* aurait pour dénotation la proposition dénotée par *John loves Mary*!

Si l'on considère des exemples légèrement plus complexes faisant intervenir par exemple, une interaction entre ellipse verbale et anaphore pronominale, la sur-génération s'accentue. Ainsi pour le discours (1.13), l'équation (1.13a)[17] a quatre solutions dont deux seulement sont correctes : (1.13b) et (1.13c).

(1.13) John love his cat. Peter does too.
 a. $R(j) = love(j, c(j))$

[17] La représentation sémantique de cet exemple a été simplifiée et devrait être : $\exists x.(love(j, x) \wedge cat(x) \wedge of(x, j) \wedge \forall y.(cat(y) \wedge of(y, j) \rightarrow y = x))$. Le raisonnement par rapport au problème de la sur-génération étant le même, je garde par la suite cette représentation simplifiée dans un souci de lisibilité.

b. $\{R \leftarrow \lambda x.love(x, c(j))\}$
c. $\{R \leftarrow \lambda x.love(x, c(x))\}$
d. $\{R \leftarrow \lambda x.love(j, c(j))\}$
e. $\{R \leftarrow \lambda x.love(j, c(x))\}$

Afin d'éliminer de telles solutions de l'ensemble des solutions acceptables au plan linguistique, le traitement de l'ellipse verbale proposé par [39] fait intervenir une restriction informelle, la **Restriction des Occurrences Primaires** (ROP), qui peut être élicitée de la façon suivante :

> Etant donné un étiquetage *primaire* versus *secondaire* des occurrences de symboles, la ROP exclut de l'ensemble des solutions valides au plan linguistique, toute solution contenant une occurrence primaire.

Intuitivement, une **occurrence primaire** est une occurrence qui est *directement* associée avec un élément parallèle source. Ni la notion d'association directe, ni celle d'élément parallèle ne sont définies formellement par DSP mais à un niveau intuitif, une occurrence primaire représente un élément de la proposition source dont l'élément parallèle cible est «explicitement réalisé» c'est-à-dire ni anaphorique ni contrôlé.

Pour voir comment cette restriction fonctionne, considérons son application à l'exemple (1.12) ci-dessus. Dans (1.12a), la représentation j de *John* est une occurrence primaire car elle est directement associée avec un élément parallèle source. Il s'ensuit que la solution (1.12c) est exclue de l'ensemble des solutions acceptables sur le plan linguistique.

De même pour l'exemple (1.13), les solutions (1.13d) et (1.13e) seront exclues de l'ensemble des solutions valides au plan linguistique car elles contiennent une occurrence primaire : la première occurrence de j c'est-à-dire, l'occurrence de j représentant l'élément parallèle source *John*.

Bien que l'intuition sous-tendant la ROP soit claire, deux objections peuvent être formulées.

Premièrement, la restriction est informelle et, de ce fait, ne fournit pas une base solide pour une évaluation mathématique et informatique. Comme DSP eux-mêmes le remarquent : une théorie générale de la ROP est nécessaire.

Deuxièmement, la méthode est une méthode «generate and test» : toutes les solutions valides au plan mathématique sont générées avant que les solutions enfreignant la ROP et donc non valides au plan linguistique, ne soient éliminées. Si cela peut suffire dans une analyse théorique, il serait préférable, pour une implémentation, de ne pas produire ces solutions du tout. Dans ce qui suit, nous montrons comment l'utilisation d'une variante de l'UOS connue sous le nom d'unification Colorée d'ordre Supérieur (UCOS) permet de résoudre ce problème.

1.5.1 L'unification colorée d'ordre supérieur (UCOS)

Dans le domaine du raisonnement automatique, il existe une technique pour guider la recherche des preuves par induction connue sous le nom de coloration de termes [95, 96] et qui utilise une version modifiée de l'UOS, l'unification Colorée d'ordre Supérieur. Une caractéristique cruciale de ce type d'unification est qu'elle permet de prendre en compte une information arbitraire associée aux occurrences des termes (les couleurs) et de maintenir et exploiter cette information pendant le processus d'inférence. Cette caractéristique différencie la coloration de techniques sémantiques d'annotation telles que les sortes qui, elles, maintiennent les attributs des symboles et non les attributs des occurrences de symboles. Les substitutions bien formées produites par L'UCOS (qui est donc définie pour une variante colorée du Lambda-calcul typé où les symboles sont annotés par des combinaisons booléennes de constantes de couleurs) doivent respecter la contrainte suivante :

> Pour toute constante de couleur c et toute variable V_{c} de couleur c, une substitution colorée bien formée doit associer à V_{c} un terme monochrome de couleur c, c'est-à-dire un terme dont les symboles sont tous de couleur c.

Dans le cadre du Lambda-calcul coloré, la ROP peut être modélisée de la façon suivante. Tandis que les occurrences primaires sont colorées pe, les variables libres sont colorées npe[18] Etant donné la restriction mentionnée ci-dessus pour les substitutions colorées bien formées, cette coloration exclut toute solution contenant une occurrence primaire puisque les variables libres étant de couleur npe doivent être associées à des termes npe-monochromes. En d'autres termes, aucune substitution ne pourra contenir d'occurrence primaire, c'est-à-dire de symbole de couleur pe. Par exemple, le discours (1.14) aura la représentation sémantique (1.14a) et l'équation (1.14b) qui a pour solution unique (1.14c). En revanche, (1.14d) n'est pas une solution puisque cette substitution associe à une variable de couleur npe, un terme contenant un symbole de couleur pe, c'est-à-dire un terme qui n'est pas npe-monochrome.

(1.14) John loves Mary. Peter does too.
  a. $love(j_{\mathsf{pe}}, m) \wedge R_{\mathsf{npe}}(p)$
  b. $love(j_{\mathsf{pe}}, m) = R_{\mathsf{npe}}(j_{\mathsf{pe}})$
  c. $\{R_{\mathsf{npe}} \leftarrow \lambda x.love(x, m)\}$
  d. $\{R_{\mathsf{npe}} \leftarrow \lambda x.love(j_{\mathsf{pe}}, m)\}$

1.5.2 Le champ d'application de la ROP

Comme nous venons de le voir, la restriction sur les occurrences primaires proposée par DSP est nécessaire pour un traitement adéquat de l'ellipse verbale. Mais le champ d'application de cette contrainte ne se réduit pas au traitement de l'ellipse verbale. Ainsi [156, 61] utilise l'UOS pour modéliser l'interprétation du focus, [72] pour l'interprétation des constructions comparatives et [56] pour les énoncés désaccentués. Pour chacune de ces approches, des restrictions similaires à la ROP doivent être mises en jeu afin d'éviter la sur-génération.

Comparaisons

Comme l'ellipse, les structures comparatives peuvent donner lieu à une ambiguïté lâche/stricte. le texte (1.15) par exemple, est ambigu entre (1.15a), (1.15b) et (1.15c).

(1.15) Jean^1 gave her$_1$ sister a more beautiful present than Carol.
  a. Jean gave Jean's sister a more beautiful present than Jean gave Carol.
  Focus = *her sister*
  b. Jean gave Jean's sister a more beautiful present than Carol gave Jean's sister.
  Focus = *Jean*; interprétation stricte
  c. Jean gave Jean's sister a more beautiful present than Carol gave Carol's sister.
  Focus = *Jean*; interprétation lâche

Un traitement des structures comparatives (et superlatives) est présenté dans [72] qui se base sur l'utilisation de l'unification. Dans cette approche, la représentation associée à (1.15) est :

$$\exists m[\forall s[R(c, s), >(m, s)], \exists y[\wedge[book(y), beau(y, m)], give(j, sis(j), y)]]$$

[18] Pour des raisons d'exposition, nous ne considérons pour l'instant que les couleurs pe (primaire pour l'ellipse) et npe (non primaire pour l'ellipse). Comme nous le verrons dans la prochaine section (section 1.5.2), le champ d'application étendu de la ROP et les interactions entre phénomènes linguistiques rendent en réalité nécessaire une formalisation plus sophistiquée.

En mots : il existe un degré m de beauté du cadeau offert par Jean à sa soeur tel que pour tout degré s qui est dans la relation R avec Carol, m est plus grand que s. Sans entrer dans les détails de cette analyse, on peut noter que cette représentation fait intervenir une variable d'ordre supérieur R. Dans l'approche de Gawron, la valeur de cette variable est déterminée par une équation faisant intervenir les éléments comparés à savoir :

1. l'élément sur lequel porte le focus dans la phrase source et l'élément qui lui est contrasté dans la phrase cible – ici : *Jean* et *Carol*
2. les deux degrés mis en comparaison – ici : m et s

Etant donné ces éléments cibles et sources, l'équation posé par Gawron pour l'exemple (1.15b-c) ci-dessus est la suivante :

$$R(j, m) = \exists y[\wedge[book(y), beau(y, m)], give(j, sis(j), y]$$

Comme Gawron lui-même le remarque, cette équation donne outre les deux solutions bien formées (1.16a-b), cinq autres solutions qui ne sont pas valides au plan linguistique dont en particulier (1.16c-d).

(1.16) a. $\{R \leftarrow \lambda z \lambda x.\exists y[\wedge[book(y), beau(y, z], give(x, s(j), y]\}$
b. $\{R \leftarrow \lambda z \lambda x.\exists y[\wedge[book(y), beau(y, z], give(x, s(x), y]\}$
c. $\{R \leftarrow \lambda z \lambda x.\exists y[\wedge[book(y), beau(y, z], give(j, s(x), y]\}$
d. $\{R \leftarrow \lambda w \lambda x.\exists y[\wedge[book(y), beau(y, z], give(j, s(x), y]\}$

Focus

Depuis [98], on sait qu'une même phrase énoncée avec différents contours prosodiques peut avoir des conditions de vérité différentes. Les exemples suivants illustrent ce point où les lettres capitales indiquent une accentuation, c'est-à-dire un focus[19] prosodique.

(1.17) a. John only gave flowers to MARY.
b. John only gave FLOWERS to Mary.

Alors que (1.17a) dit que la seule personne à qui John a donné des fleurs est Mary, (1.17b) indique que les seules choses que John ait données à Mary sont des fleurs.

Pour rendre compte de cette interaction entre focus et conditions de vérité, une **valeur de focus**[20] est généralement utilisée qui dénote l'ensemble des objets sémantiques obtenus en remplaçant la partie de la représentation sémantique représentant le focus par des expressions de type approprié. Par exemple, [61] définit la valeur de focus au moyen de l'équation suivante :

Équation pour la Valeur de Focus	$Sem = V(F)$

où *Sem* est la représentation sémantique de la phrase analysée lorsque l'on omet les opérateurs liés au focus (e.g. *seulement, aussi*) ; F est la représentation sémantique du focus syntaxique ; et V est une variable servant à déterminer la valeur de focus de la façon suivante :

[19]Le terme «focus» a été utilisé pour différentes choses. Ici la terminologie adoptée est celle de Jackendoff : j'utilise le terme «focus» pour désigner la valeur sémantique de cette partie de la phrase qui correspond au focus syntaxique c'est-à-dire à une phrase contenant le focus prosodique et obéissant au règles de projection du focus prosodique vers le focus syntaxique (cf. [122] pour une discussion approfondie de ces règles).

[20]La valeur de focus est définie et désignée de différentes façon par différents auteurs : ainsi [98] utilise le terme *ensemble présuppositionnel* , [165] le terme *ensemble d'alternatives* et [120] le terme *ground.*

Valeur de Focus Si V est de type $\alpha = \vec{\beta_k} \rightarrow t$ et n est le nombre de foci ($n \leq k$), alors la Valeur de Focus dérivable à partir de V, notée $\overline{V}$, est $\{V(t^1 \dots t^n) \mid t^i \in \mathit{wff}_{\beta_i}\}$.

Cette définition donne une valeur de focus équivalente au «squelette présuppositionnel» (presupposition skeleton) de [119]. Ainsi étant donné la phrase (1.17a) ci-dessus, l'équation pour la valeur de focus est (1.18a) avec pour solution possible (1.18b) ; la valeur de focus $\overline{V}$ qui en est dérivée est (1.18c) et la représentation sémantique est (1.18d), aussi équivalente à (1.18e). En français : la seule proposition de la forme *John a donné des fleurs à x* qui soit vraie est la proposition *John a donné des fleurs à Mary.*

(1.18) a. $give(j,f,m) = V(m)$
b. $\{V \leftarrow \lambda x.give(j,f,x)\}$
c. $\overline{V} = \{give(j,f,x) \mid x \in \mathit{wff}_e\}$
d. $\forall P.[P \in \overline{V} \wedge P \rightarrow P = give(j,f,m)]$
e. $\forall P.[P \in \{give(j,f,x) \mid x \in \mathit{wff}_e\} \wedge P \rightarrow P = give(j,f,m)]$

Comme l'analyse de DSP pour l'ellipse, cette analyse du focus sur-génère au sens où des solutions (valeurs de focus) sont produites qui ne reflètent pas les interprétations possibles de l'énoncé analysé. Ainsi, l'équation (1.18a) a pour solution possible, outre la solution correcte (1.18b), la solution incorrecte :

(1.19) $\{V \leftarrow \lambda x.give(j,f,m)\}$

Il est clair cependant, qu'une contrainte similaire à la ROP de DSP peut être utilisée pour éliminer les solutions incorrectes. Il suffit pour ce faire de traiter tout terme directement associé au focus comme étant une occurrence primaire, c'est-à-dire une occurrence qui doit être exclue des solutions valides. Pour rendre compte du fait que ces occurrences sont primaires pour l'interprétation du focus et non de l'ellipse verbale, on peut associer à ces occurrences primaires la couleur pf plutôt que pe comme c'est le cas pour les occurrences primaires de l'ellipse. Par ailleurs, de la même façon qu'une variable représentant une ellipse verbale est colorée npe afin d'exclure de l'ensemble des solutions celles qui assignent à cette variable, un terme contenant une occurrence primaire pour l'ellipse, une variable représentant une valeur de focus sera colorée npf excluant ainsi de l'ensemble des solutions toute substitution qui associe à cette valeur de focus un terme contenant une occurrence primaire pour le focus, c'est-à-dire un terme de couleur pf. Modifiant l'équation (1.18a) dans ce sens, on aura alors l'équation colorée (1.20a) avec pour unique solution (1.20b). La substitution (1.20c) n'est pas une solution valide puisqu'elle assigne à une variable de couleur npf, un terme contenant un symbole de couleur pf, c'est-à-dire un terme qui n'est pas npf-monochrome.

(1.20) a. $give(j,f,m_{\mathsf{pf}}) = V_{\mathsf{npf}}(m_{\mathsf{pf}})$
b. $\{V_{\mathsf{npf}} \leftarrow \lambda x.give(j,f,x)\}$
c. $\{V_{\mathsf{npf}} \leftarrow \lambda x.give(j,f,m_{\mathsf{pf}})\}$

Analyse compositionnelle et énoncés désaccentués

Comme nous l'avons vu en section 1.4, il est possible de modifier l'analyse de DSP de façon à la rendre compatible avec une sémantique compositionnelle à la Montague. Nous montrons maintenant que cette analyse modifiée sur-génère fortement. Nous montre ensuite comment l'UCOS permet de remédier à ce problème. Reprenons l'exemple (1.13) ci-dessus répété ici en (1.21) :

(1.21) John^1 loves his$_1$ cat. Peter does too.

Rappelons que si, comme les indexes l'indiquent, le pronom «his» réfère à «John», (1.21) a deux lectures possibles :

(1.22) a. John aime le chat de John et Peter aime le chat de John.
b. John aime le chat de John et Peter aime le chat de Peter.

Pour cet exemple, l'analyse unificationnelle modifiée impose les équations suivantes :

(1.23) $C(j) = l(j, c(j)) \qquad C(p) = R(p)$

Pour ces équations, l'UOS retourne neuf solutions dont trois seulement sont valides au plan linguistique : *Sol. 1* qui donne l'interprétation stricte et *Sol. 3* et *6* qui retournent une interprétation lâche.

(1.24)
Sol.1 $\{C \leftarrow \lambda x.l(x, c(j)), R \leftarrow \lambda x.l(x, c(j))\}$
Sol.2 $\{C \leftarrow \lambda x.l(x, c(j)), R \leftarrow \lambda x.l(p, c(j))\}$
Sol.3 $\{C \leftarrow \lambda x.l(x, c(x)), R \leftarrow \lambda x.l(x, c(x))\}$
Sol.4 $\{C \leftarrow \lambda x.l(x, c(x)), R \leftarrow \lambda x.l(p, c(x))\}$
Sol.5 $\{C \leftarrow \lambda x.l(x, c(x)), R \leftarrow \lambda x.l(x, c(p))\}$
Sol.6 $\{C \leftarrow \lambda x.l(x, c(x)), R \leftarrow \lambda x.l(p, c(p))\}$
Sol.7 $\{C \leftarrow \lambda x.l(j, c(x)), R \leftarrow \lambda x.l(j, c(p))\}$
Sol.8 $\{C \leftarrow \lambda x.l(j, c(x)), R \leftarrow \lambda x.l(j, c(x))\}$
Sol.9 $\{C \leftarrow \lambda x.l(j, c(j)), R \leftarrow \lambda x.l(j, c(j))\}$

Là encore, pour éliminer les solutions indésirables, une restriction similaire à la ROP est nécessaire. Elle peut être formulée de la manière suivante :

> **ROP 2.** Les occurrences de symboles représentant des éléments contrastifs (c'est-à-dire tout élément dont l'élément parallèle est (i) réalisé phonétiquement et (ii) sémantiquement distinct) ne peuvent apparaître ni dans la valeur attribuée à R, la représentation sémantique de l'ellipse verbale, ni dans la valeur attribuée à C, la sémantique partagée par la proposition cible et par la proposition source.

Appliquée à l'exemple (1.21), cette contrainte exige que les occurrences directement associées avec les éléments contrastifs «John» et «Peter» (c'est-à-dire la première occurrence de j dans $l(j, c(j))$ et l'occurrence unique de p) n'apparaissent pas dans les valeurs assignées à R et à C ce qui a pour effet d'exclure de l'ensemble des solutions valides au plan linguistique les solutions *2,4,5,6,7,8* et *9* c'est-à-dire, toutes les solutions incorrectes.

Pour implémenter cette contrainte dans le cadre de l'UCOS, la technique est la même que pour la ROP de DSP : alors que les représentations des éléments contrastifs sont colorées c, les variables libres sont colorées npc. Comme les variables libres sont de couleur npc, elles doivent être associées à des termes npc-monochromes. Il s'ensuit qu'aucune substitution ne peut contenir de symbole de couleur pc . Concrètement, l'équation associée à l'exemple (1.21) sera (1.25) avec pour seules solutions valides au plan mathématique les solutions (1.25a) et (1.25b). Toutes les autres solutions sont exclues du fait qu'elles n'associent pas aux variables R_{npc}, C_{npc} des termes npc-monochromes.

(1.25) $C_{\mathsf{npc}}(j_{\mathsf{c}}) = l(j_{\mathsf{c}}, c(j)) \quad C_{\mathsf{npc}}(p_{\mathsf{c}}) = R_{\mathsf{npc}}(p_{\mathsf{c}})$
Sol.1 $\{C_{\mathsf{npc}} \leftarrow \lambda x.l(x, c(x)), R_{\mathsf{npc}} \leftarrow \lambda x.l(x, c(x))\}$
Sol.2 $\{C_{\mathsf{npc}} \leftarrow \lambda x.l(x, c(x)), R_{\mathsf{npc}} \leftarrow \lambda x.l(x, c(p_{\mathsf{c}}))\}$
Sol.3 $\{C_{\mathsf{npc}} \leftarrow \lambda x.l(x, c(x)), R_{\mathsf{npc}} \leftarrow \lambda x.l(p_{\mathsf{c}}, c(p_{\mathsf{c}}))\}$
Subst.4 $\{C_{\mathsf{npc}} \leftarrow \lambda x.l(x, c(x)), R_{\mathsf{npc}} \leftarrow \lambda x.l(p_{\mathsf{c}}, c(x))\}$

Subst.5 $\{C_{\mathsf{npc}} \leftarrow \lambda x.l(j_{\mathsf{c}}, c(x)), R_{\mathsf{npc}} \leftarrow \lambda x.l(j_{\mathsf{c}}, c(p_{\mathsf{c}}))\}$
Subst.6 $\{C_{\mathsf{npc}} \leftarrow \lambda x.l(j_{\mathsf{c}}, c(x)), R_{\mathsf{npc}} \leftarrow \lambda x.l(j_{\mathsf{c}}, c(x))\}$
Subst.7 $\{C_{\mathsf{npc}} \leftarrow \lambda x.l(x, c(j)), R_{\mathsf{npc}} \leftarrow \lambda x.l(x, c(j))\}$
Subst.8 $\{C_{\mathsf{npc}} \leftarrow \lambda x.l(x, c(j)), R_{\mathsf{npc}} \leftarrow \lambda x.l(p_{\mathsf{c}}, c(j))\}$
Subst.9 $\{C_{\mathsf{npc}} \leftarrow \lambda x.l(j_{\mathsf{c}}, c(j)), R_{\mathsf{npc}} \leftarrow \lambda x.l(j_{\mathsf{c}}, c(j))\}$

Chapitre 2

UOS et énoncés désaccentués

La désaccentuation est une réduction prosodique prononcée du tout, ou d'une partie, d'un énoncé. Elle est typique de plusieurs types de contexte tels les corrections (2.1a), les phrases écho (2.1b), les variantes (2.1c), les paires questions-réponses (2.1d), les conditionnels répétés (2.1e), les comparaisons (2.1f) et les contrastes formels (2.1g)[21].

(2.1) a. A : John only likes Mary. B : No, PETER [only likes Mary].
b. A : John only likes Mary. B : That is true : [John only likes Mary].
c. A : John only likes Mary. B : So ? Even PETER [only likes Mary].
d. A : Who only likes Mary ? B : PETER [only likes Mary].
e. John wants to go to Paris and if [he goes to Paris], Mary will go with him.
f. John wants to go to Paris more than [he wants to go to] ROME.
g. John wants to go to Paris but MARY [wants to go to] ROME.

Dans ce chapitre, nous examinons les cas de désaccentuation où l'énoncé contient à la fois une réduction et une proéminence prosodique. Comme [122, 164] et [121] l'ont montré, la fonction pragmatique de ces énoncés est de mettre en rapport l'énoncé désaccentué avec une proposition saillante de la façon suivante : la proéminence prosodique est en relation de contraste avec la partie correspondante de la proposition saillante tandis que la réduction est en relation de redondance sémantique. La question que ce chapitre explore est la suivante :

> *Quelle est exactement la relation de redondance qui accompagne la désaccentuation prosodique ?*

Et plus précisément :

> *Comment faut-il définir cette relation pour prédire de façon correcte les interactions de la désaccentuation avec des phénomènes sémantiques indépendants tels les opérateurs sensibles au focus, les anaphores, les définis et les indéfinis ?*

Il a souvent été démontré que la désaccentuation partage avec l'ellipse verbale certaines caractéristiques sémantiques. En particulier, [164] a montré que ces deux phénomènes sont régis par une même contrainte de redondance sémantique. Cette observation est illustrée par l'exemple suivant (comme les exposants l'indiquent, l'interprétation pertinente est celle où *Mary* et *her* coréférent) :

[21] Dans cet exemple et dans ce qui suit, la partie désaccentuée de l'énoncé est entourée par des parenthèses carrées tandis que les mots en majuscules indiquent les éléments marqués par une proéminence prosodique.

(2.2) a. First, John told Mary^1 I was bad-mouthing her$_1$ and then SUE [heard I was] [22]
b. First, John told Mary^1 I was bad-mouthing her$_1$ and then SUE [heard I was bad-mouthing her] [23]

Dans les deux cas, la dernière clause peut être interprétée de façon lâche : *I was bad-mouthing Sue*. Pourtant dans l'énoncé elliptique (2.2a), *Sue* ne fait pas partie de la clause contenant l'antécédent de l'ellipse *was*. Cela suggère que la contrainte de redondance qui régit l'ellipse doit être formulée au niveau du domaine propositionnel (plutôt que clausal ou même verbal). Le deuxième exemple (2.2b) montre en outre que cette contrainte s'applique aussi bien à l'ellipse qu'aux énoncés désaccentués. Conjointement, ces observations suggèrent qu'un principe commun puisse être identifié qui sous-tende les théories de l'ellipse et de la désaccentuation et permette ainsi d'expliquer ces similarités entre ellipses et énoncés désaccentués.

Dans ce chapitre, nous argumentons que le traitement unificationnel de l'ellipse présenté par DSP, parce qu'il peut sans difficulté être étendu aux énoncés désaccentués, fournit un tel principe. Le chapitre est structuré comme suit. La section 2.1 présente l'analyse proposée et identifie les hypothèses de travail. La section 2.2 applique cette analyse aux données et en particulier à l'interaction de la désaccentuation avec l'anaphore, les opérateurs liés au focus, les définis et les indéfinis. La section 2.3 compare l'analyse proposée avec deux analyses voisines : le traitement de la désaccentuation avancé par [164] et la traitement du parallélisme du discours de [87]. Section 2.4 conclut en identifiant les problèmes laissés sans réponse.

2.1 Une analyse unificationnelle pour les énoncés désaccentués

Comme nous l'avons vu dans le chapitre précédent, la relation reconstruite par l'analyse unificationnelle de DSP pour l'ellipse n'est pas nécessairement la relation correspondant à l'interprétation conventionnelle d'un groupe verbal. Par exemple, pour la phrase (2.3a), l'équation postulée par DSP est celle donnée en (2.3b) si bien que la relation reconstruite (2.3c) est une relation ternaire plutôt que, comme il l'est normalement associé avec un GV, une relation unaire :

(2.3) a. The person who introduced Mary^1 to John^2 would not give her$_1$ his$_2$ telephone number and nor would the person who introduced Sue to Peter.
b. $R(pwi(m,j),m,j) = refuse(pwi(m,j),give(m,phone(j)))\}$
c. $\{R \leftarrow \lambda x.\lambda y.\lambda z.refuse(x,give(y,phone(z)))\}$

Il s'ensuit que les contraintes équationnelles postulées par DSP (ainsi que par l'analyse modifiée présentée en section 1.4) doivent être vues non pas comme servant à reconstruire la représentation sémantique des ellipses du GV mais plutôt comme implémentant une contrainte d'ordre plus général. Cette contrainte, que nous appellerons «contrainte de redondance», exige qu'une proposition elliptique partage avec sa proposition source une sémantique commune qui est celle formalisée par le terme reconstruit par unification. Ce terme reconstruit peut être identique au terme représentant le sens de l'ellipse, mais ne l'est pas nécessairement. En section (2.2), nous montrons que cette contrainte de redondance, proposée initialement par DSP pour l'interprétation des GVs elliptiques, peut également être utilisée pour rendre compte de la sémantique des énoncés désaccentués. Auparavant cependant, nous identifions les hypothèses de travail sur lesquelles repose l'analyse proposée.

[22] D'abord Jean a dit à Marie^1 que je l$_1$'avais insultée. Puis SUE [l'a entendu dire]
[23] D'abord Jean a dit à Marie^1 que je l$_1$'avais insultée. Puis SUE [a entendu dire que je l'avais insultée]

2.1.1 Parallélisme et accent contrastif

Reprenons les représentations et équations proposées dans le chapitre précédent pour un traitement de l'ellipse compatible avec une approche à la Montague :

Représentation	$S \wedge C$
Équations	$R(S_1, \ldots, S_n) = S$ $R(C_1, \ldots, C_n) = C$

Comme nous l'avons vu au chapitre 1 (cf. section 1.4), $S_1, \ldots, S_n$ et $C_1, \ldots, C_n$ représentent la sémantique des éléments parallèles sources et cibles respectivement. La notion de parallélisme n'est pas définie par DSP et nous ne la définirons pas non plus. Toutefois, dans le cas des énoncés désaccentués, il est possible de caractériser de façon plus précise les éléments parallèles cibles ainsi que leur relation aux éléments parallèles source. Comme nous l'avons vu plus haut, les éléments accentués qui apparaissent dans un énoncé désaccentué indiquent une information nouvelle ou contrastive alors qu'à l'inverse, les éléments désaccentués indiquent une information donnée, c'est-à-dire déjà mentionnée. Les équations régissant l'interprétation des énoncés désaccentués peuvent ainsi être affinées par les contraintes suivantes :

1. Les éléments parallèles cibles d'un énoncé désaccentué sont ses éléments accentués.
2. Chaque élément parallèle cible d'un énoncé désaccentué doit être dans une relation de contraste avec sa contrepartie dans la proposition source.

La notion de contraste est bien sûr une notion floue. Cependant nous verrons qu'il suffit d'exiger, pour rendre compte des données considérées, que deux éléments contrastifs aient une dénotation distincte.

2.1.2 Propositions explicites vs. propositions inférées

Comme le souligne [111], l'analyse unificationnelle de DSP n'exige pas que la proposition source soit une proposition explicitement réalisée. Dans certains cas, cette proposition peut représenter l'interprétation d'une proposition inférée à partir du contexte linguistique. Ainsi par exemple, Kehler et Shieber montrent que dans l'énoncé (2.4a), la proposition source nécessaire pour reconstruire l'interprétation de la clause elliptique est la proposition inférée *John hoped she would pass him.* Les énoncés (2.4b) et (2.4c) illustrent d'autres exemples de cas où il est probable que l'inférence, ou au moins un certain type de reconstruction, soit nécessaire pour établir la proposition source[24].

(2.4) a. Every^1 boy in Mrs. Smith^2's class hoped she$_2$ would pass him$_1$. In John's case, I think she will.
Trad. *Chaque garçon dans la classe de Mrs. Smith espère qu'elle le fera passer. Dans le cas de John, je pense qu'elle le fera.*
b. Irv and Mary want to dance together but Mary can't since her husband is here.
Trad. *Irv et Mary veulent danser ensemble mais Mary ne peut pas car son mari est là.*
c. Mary wants to go to Spain and Fred wants to go to Peru but because of limited resources, only one of them will.
Trad. *Mary veut aller en Espagne et Fred au Perou mais un seul d'entre eux le fera du fait de ressources limitées.*

[24]Exemples de [193].

Pour les énoncés désaccentués, la nécessité de recourir à des propositions sources inférées est encore plus nette. Il est en effet bien connu [164] que ces énoncés ne sont pas nécessairement précédés par un énoncé structurellement identique comme c'est le cas par exemple en (2.5a). Ainsi en (2.5b), l'énoncé désaccentué est relié à la proposition source par une relation d'implication. On note en effet que la désaccentuation de *insulted her* n'est acceptable que dans un contexte où traiter quelqu'un de Républicain est une insulte. Il s'ensuit que la proposition source légitimant l'énoncé désaccentué *PETER [insulted her]* est très probablement la proposition inférée *John insulted Mary* plutôt que la proposition explicite *John called Mary a republican.*

(2.5) a. First, JOHN insulted Mary and then, PETER [insulted her].
b. First, JOHN called Mary a republican and then, PETER [insulted her].
c. HE hit HER and then HE [punched] HIM.

Enfin, il y a des cas où la relation d'implication passe par une troisième proposition qui est une proposition impliquée à la fois par la proposition source et par la proposition cible. Ainsi en (2.5c), il n'y a pas de redondance sémantique entre les propositions. La désaccentuation n'est pas non plus rendue légitime par une implication de la phrase précédente. Néanmoins, le verbe *punched* peut être désaccentué. Comme l'a montré [188], de tels cas de désaccentuation sont légitimés par la présence d'un fond commun (*common ground*) entre les deux phrases, c'est-à-dire par le fait que ces phrases ont une implication commune, en l'occurrence : «x est violent envers y».

Dans ce qui suit, nous supposons donc que, dans le cas de la désaccentuation, propositions sources et propositions cibles peuvent toutes deux être dérivées par inférence à partir du contexte linguistique. Cette analyse diffère du traitement unificationnel de l'ellipse où seule la proposition source peut être inférée.

2.1.3 Contrainte de redondance

Pour conclure cette section, nous récapitulons le contenu de la contrainte de redondance proposée ici pour rendre compte des similarités sémantiques entre ellipse du GV et énoncés désaccentués.

Contrainte de redondance. Si C est la représentation sémantique d'une phrase cible contenant soit une ellipse du GV soit un énoncé désaccentué, si S est la proposition source de C, si $S_1, \ldots, S_n$ et $C_1, \ldots, C_n$ représentent la sémantique des éléments parallèles sources et cibles respectivement, et si R est une variable libre du type approprié, alors la contrainte de redondance exige que les équations suivantes soient vérifiées :

$$\begin{aligned} R(S_1, \ldots, S_n) &= S \\ R(C_1, \ldots, C_n) &= C \end{aligned}$$

Pour les énoncés désaccentués, il faut en outre que les éléments parallèles cibles soient accentués et que leur dénotation soient distinctes de celle des éléments parallèles source.

2.2 Prédictions empiriques

Dans un traitement unificationnel de l'ellipse et des énoncés désaccentués, la résolution des anaphores et des ellipses est contrainte par l'unification au sens où la représentation sémantique de ces deux constructions est unifiée avec celle des éléments parallèles sources. Cette analyse fait donc une prédiction forte quant à l'interaction entre désaccentuation et anaphores. Elle prédit que toute anaphore apparaissant dans la partie désaccentuée (ou elliptique) de la proposition cible est une **anaphore parallèle**, c'est-à-dire, une anaphore qui est obligatoirement co-référentielle[25] avec sa contrepartie dans la proposition source. Dans ce qui suit, nous montrons que cette prédiction est confirmée par les données empiriques. D'une part, elle capture de façon appropriée les restrictions placées par l'énoncé désaccentué sur son contexte linguistique (c'est-à-dire sur la proposition source). D'autre part, elle rend correctement compte de l'interaction entre désaccentuation et anaphores. Ceci inclut non seulement les anaphores «standards» tels que les pronoms et les définis, mais également des anaphores plus pragmatiques comme le domaine de quantification des opérateurs sensibles au focus.

2.2.1 Énoncés désaccentués et contexte linguistique

Tous les contextes ne permettent pas la désaccentuation. Par exemple, la paire d'énoncés (2.6a-b) ne forme pas un dialogue acceptable (*Sarah* n'a pas d'accent contrastif).

(2.6) a. A : John loves Mary.
b. B : $\star$ No, PETER [loves Sarah]

L'approche unificationnelle rend compte de cette inacceptabilité. Ainsi étant donné la contrainte de redondance indiquée en section (2.1.3), l'interprétation de (2.6) sera contrainte par les équations suivantes :

$$R(j) = love(j, m)$$
$$R(p) = love(p, s)$$

Il n'existe pas de substitution de valeurs pour la variable libre R qui permette d'identifier *love(j,m)* avec $R(j)$ et *love(p,s)* avec $R(p)$. En conséquence, l'unification échoue
indiquant ainsi que le dialogue (2.6) est mal formé. Par contraste, le dialogue bien formé :

(2.7) a. A : John loves Mary.
b. B : No, PETER [loves Mary]

est contraint par les équations :

$$R(j) = love(j, m)$$
$$R(p) = love(p, m)$$

pour lesquelles, la substitution :

$$\{R \leftarrow \lambda x.love(x,m)\}$$

est une solution. On voit donc que l'approche unificationnelle prédit correctement la différence d'acceptabilité entre (2.6) et (2.7) posant ainsi les bases pour une modélisation des contraintes placées par un énoncé désaccentué sur son contexte linguistique.

[25]Nous employons ici le terme «co-référentielle» dans un sens large capturant à la fois la coréférence classique (deux expressions référant au même individu) et l'identité de deux variables Lambda liées (deux expressions sont représentées par la même variable Lambda). Comme nous le verrons plus loin, le parallélisme évoqué englobe ainsi à la fois les interprétations strictes et les interprétations lâches.

2.2.2 Anaphores nominales

Comme l'a observé [123], les anaphores nominales qui apparaissent dans la partie désaccentuée d'un énoncé ont une préférence marquée pour une résolution «parallèle». Par exemple, étant donné l'énoncé initial (2.8), une continuation avec un pronom désaccentué telle que (2.8a) favorise une résolution parallèle des pronoms où *he* coréfére avec le GN parallèle source *John* et *him* avec le GN parallèle source *Bill*. Par contraste, si les pronoms sont accentués comme par exemple en (2.8b), l'interprétation préférentielle est donnée par une «résolution croisée» où le pronom sujet est identifié avec le GN source objet et le pronom objet, avec le GN source sujet.

(2.8) John^1 called Bill^2 a republican
a. and then [he_1] INSULTED [him_2].
b. and then HE_2 insulted HIM_1.

Dans ce qui suit, nous montrons que l'analyse unificationnelle rend compte de l'observation faite par Lakoff et, plus généralement, qu'elle prédit les interactions possibles et impossibles entre désaccentuation et anaphores nominales.

Anaphores pronominales

Commençons avec un exemple simple :

(2.9) A : John likes Mary^1
B : No, PETER [likes her_1]

Si dans (2.9b), *her* est désaccentué, (2.9b) a pour seule interprétation *Peter likes Mary*. En d'autres termes, la paire d'énoncés (2.9a-b) n'est bien formée que si le pronom *her* est identifié avec sa contrepartie dans la proposition source, c'est-à-dire avec *Mary*. En particulier, les dialogues (2.10-2.10a) et (2.10-2.10b) sont mal formés parce que le pronom cible n'est pas, ou ne peut pas, être résolu à son élément parallèle source.

(2.10) John likes Mary^1
a. $\star$ No, PETER [likes her_2]
b. $\star$ No, PETER [likes him_2]

Ces données sont capturées par l'analyse unificationnelle qui prédit que tout élément sémantiquement sous-spécifié apparaissant dans la partie désaccentuée (et donc redondante) d'un énoncé doit être résolu à son élément parallèle source. Cette prédiction résulte du fait que la partie désaccentuée de l'énoncé est représentée par une variable libre R dont la valeur est contrainte par les équations à être la même pour la proposition cible et pour la proposition source. Ainsi dans l'exemple (2.9), il y a deux cas de figure possibles. Soit le pronom dans la partie désaccentuée est identifié avec son élément parallèle source, soit il ne l'est pas. La première possibilité donne les équations suivantes :

$$\begin{aligned} R(j) &= like(j, m) \\ R(p) &= like(p, m) \end{aligned}$$

Dans ce cas, l'unification réussit, l'une des substitutions unifiante étant celle qui attribue à R la valeur $\lambda y.like(y, m)$. Par opposition, si le pronom désaccentué est résolu à un autre antécédent c'est-à-dire à une constante c distincte de m ou à une variable liée, l'unification échoue. Le premier cas conduit à un conflit de constantes entre la constante m représentant l'élément parallèle source et la représentation c du pronom résolu. Le deuxième cas est exclu par une contrainte sur les substitutions bien formées du Lambda-calcul qui interdit la capture accidentelle des variables libres.

Revenons en à l'exemple (2.8a) de Lakoff. Comme le contour prosodique l'indique, le contraste est entre *calling somebody a republican* et *insulting somebody*. En admettant que la représentation sémantique de (2.8a) soit celle donnée en (2.11a), les contraintes équationnelles seront celles données en (2.11b) pour lesquelles il existe la substitution unifiante (2.11c).

(2.11) a. $call\_rep(j,b,r) \wedge ins(j,b)$

b. $R(\lambda x.\lambda y.call(x,y,r)) = call(j,b,r)$
$R(\lambda x.\lambda y.ins(x,y)) = ins(j,b)$

c. $\{R \leftarrow \lambda P.P(J,b)\}$

Si en revanche, les pronoms cibles donnent lieu à une résolution croisée, on obtient les équations indiquées en (2.12) pour lesquelles il n'existe pas de substitution unifiante.

(2.12) $R(\lambda x.\lambda y.call(x,y,r)) = call(j,b,r)$ $\quad R(\lambda x.\lambda y.ins(x,y)) = ins(b,j)$

Ainsi les pronoms désaccentués sont correctement résolus par l'analyse unificationnelle à leur élément parallèle source. Mais cette analyse peut-elle également rendre compte du fait que comme illustré par l'exemple (2.8b), – répété ici en (2.13) – les pronoms accentués ne peuvent pas, eux, être identifiés avec leur élément parallèle source ?

(2.13) John^1 called Bill^2 a republican and then HE$_2$ insulted HIM$_1$.

La réponse est «oui». En effet, supposons une résolution parallèle où *HE* est résolu à *John* et *HIM* à *Bill*. Dans ce cas, les éléments accentués *HE* et *HIM* ne diffèrent pas de leur élément parallèle source. Or cette résolution est exclue par la contrainte (cf. 2.1.1) exigeant que les éléments parallèles cibles soient des éléments prosodiquement proéminents et donc contrastent avec leurs éléments parallèles sources. Par contre, si la résolution inverse est choisie, *John* contraste avec *Bill* (les deux éléments ont une dénotation distincte) et *vice versa*. La contrainte exigeant que les éléments parallèles soient dans une relation de contraste est alors satisfaite et l'on obtient les équations données en (2.14a) pour lesquelles l'UOS retourne la solution (2.14b).

(2.14) a. $R(j,b) = ins(j,b)$
$R(b,j) = ins(b,j)$

b. $\{R \leftarrow \lambda x \lambda y.ins(x,y)\}$

Ambiguïté lâche/stricte

Nous venons de voir que l'approche unificationnelle rend correctement compte de cas simples impliquant des pronoms désaccentués ou accentués. Nous considérons maintenant des cas plus complexes d'interaction entre désaccentuation et anaphores pronominales, des cas où cette interaction donne lieu non pas à une, mais à deux interprétations pour l'énoncé désaccentué. Ainsi que l'a montré [165], les énoncés désaccentués, comme les ellipses verbales, peuvent exhiber une ambiguïté lâche/stricte. Ainsi par exemple, l'énoncé elliptique (2.15a) et l'énoncé désaccentué (2.15b) donnent tous deux lieu à la fois à une interprétation stricte (*Bill took John's wife to the station*) et à une interprétation lâche (*Bill took Bill's wife to the station*).

(2.15) a. Jon^1 took his$_1$ wife to the station. No, BILL did.

b. Jon^1 took his$_1$ wife to the station. No, BILL [took his wife to the station].

Pour l'énoncé elliptique (2.15a), l'analyse DSP postule l'équation suivante :

$$R(j) = \exists x[st(x) \wedge tk(j, wife\_of(j), x)]$$

L'UOS retourne les substitutions suivantes :

$$\{R \leftarrow \lambda y.\exists x[st(x) \wedge tk(y, wife\_of(y), x)]\}$$
$$\{R \leftarrow \lambda y.\exists x[st(x) \wedge tk(y, wife\_of(j), x)]\}$$

Et la clause cible reçoit (entre autres[26]) les deux interprétations :

$$\exists x[st(x) \wedge tk(b, wife\_of(b), x)] \quad \textit{(interprétation lâche)}$$
$$\exists x[st(x) \wedge tk(b, wife\_of(j), x)] \quad \textit{(interprétation stricte)}$$

Prenons maintenant l'énoncé désaccentué (2.15b). L'interprétation lâche est celle où le pronom désaccentué *his* est résolu à *Bill* si bien que la contrainte de redondance sur les énoncés désaccentués impose les contraintes équationnelles suivantes :

$$R(j) = \exists x[st(x) \wedge tk(j, wife\_of(j), x)]$$
$$R(b) = \exists x[st(x) \wedge tk(b, wife\_of(b), x)]$$

Ces équations sont résolues par la substitution unifiante :

$$\{R \leftarrow \lambda y.\exists x[st(x) \wedge tk(y, wife\_of(y), x)]\}$$

Si par contre, le pronom est interprété de façon stricte, les contraintes équationnelles sont :

$$R(j) = \exists x[st(x) \wedge tk(j, wife\_of(j), x)]$$
$$R(b) = \exists x[st(x) \wedge tk(b, wife\_of(j), x)]$$

qui sont résolues par la substitution :

$$\{R \leftarrow \lambda y.\exists x[st(x) \wedge tk(y, wife\_of(j), x)]\}$$

L'interprétation stricte et l'interprétation lâche sont donc toutes deux autorisées par la contrainte de redondance sur les énoncés désaccentués. Toute autre résolution du pronom désaccentué *his* est exclue car elle résulterait en un échec du processus d'unification.

Considérons maintenant la variante suivante de (2.15b) :

(2.16) Jon^1 took his$_1$ wife to the station. No, BILL [took] HIS [wife to the station].

Dans ce cas, l'interprétation stricte n'est pas disponible et la clause désaccentuée est interprétée univoquement comme signifiant *Bill took Bill's wife to the station*. Comment cela peut-il être modélisé par l'analyse d'unification ? Notons que dans ce cas, il y a non pas un, mais deux éléments proéminents dans la clause cible et donc deux éléments cibles qui contrastent avec – doivent avoir une dénotation distincte de – deux éléments source. Ainsi puisque le pronom *his* dans la première clause est résolu à *John*, le pronom *HIS* dans la seconde clause – qui est son élément parallèle cible – doit être résolu à quelqu'un d'autre que *John* et la contrainte de redondance sera de la forme :

$$R(j, j) = \exists x[st(x) \wedge tk(j, wife\_of(j), x)]$$
$$R(b, b) = \exists x[st(x) \wedge tk(b, wife\_of(b), x)]$$

[26]Ici, comme dans ce qui suit, nous mentionnons uniquement les solutions fournies par l'UOS qui sont valides au plan linguistique. Faisant l'hypothèse d'un mécanisme d'élimination de la sur-génération tel celui décrit au chapitre 1, nous omettrons systématiquement dans le reste du texte de mentionner les solutions valides au plan mathématique mais invalides au plan linguistique.

Pour ces équations, l'UOS retourne la substitution :

$$\{R \leftarrow \lambda z.\lambda y.\exists x[st(x) \wedge tk(z, wife\_of(y), x)]\}$$

Plus généralement, l'analyse unificationnelle permet à un pronom accentué apparaissant dans un énoncé désaccentué d'être résolu à tout référent autre que son élément parallèle source. Comme les exemples qui suivent l'illustrent, ceci semble être la bonne généralisation :

(2.17) a. George told me that Jon^1 called Bill^2 a republican and then HE [insulted] HIM.
b. George told me that Jon^1 took his$_1$ wife to the station. No, BILL [took] HIS [wife to the station].

Dans les deux exemples, les pronoms accentués peuvent être résolus à n'importe lequel des antécédents présents dans la clause source à l'exception toutefois de leur élément parallèle source. Ainsi *HE* dans (2.17a) peut référer soit à *George* soit à *Bill* et *HIM* soit à *George* soit à *Jon*. De même, *HIS* en (2.17b) peut référer soit à *George* soit à *Bill*.

Enfin, considérons l'exemple suivant de [164] :

(2.18) Yesterday, Jon^1's friends paid off his$_1$ debts and today BILL^2 [paid off his debts] too.

Si dans la proposition cible, le pronom *his* est désaccentué, la seule interprétation possible est l'interprétation stricte. Si ce pronom est accentué en revanche, l'interprétation lâche est préférée. Dans le premier cas, l'interprétation stricte induit les contraintes équationnelles suivantes :

$$\begin{array}{l} R(j\text{'s friends}) = paid(j\text{'s friends}, j\text{'s debts}) \\ R(b) = paid(b, j\text{'s debts}) \end{array}$$

avec pour solution :

$$\{R \leftarrow \lambda x.paid(x, j\text{'s debts})\}$$

Par contraste, une interprétation lâche (*his* = *Bill*) est impossible puisque le syntagme *jon* dans la phrase source n'a pas d'élément parallèle dans la phrase cible.

Dans le second cas, c'est-à-dire lorsque la seconde occurrence de *his* est accentuée, les contraintes de contraste sur les éléments parallèles exigent que le référent de *Jon's friends* soit différent de celui de *BILL* et que celui de *his*$_1$ (dont l'antécédent est *Jon*) soit différent de celui de *HIS*. En particulier, une résolution possible est celle où *HIS* réfère à *Bill* qui correspond à une interprétation lâche. Dans ce cas, les contraintes équationnelles seront :

$$\begin{array}{l} R(j\text{'s friends}, j) = paid(j\text{'s friends}, j\text{'s debts}) \\ R(b, b) = paid(b, b\text{'s debts}) \end{array}$$

avec pour solution :

$$\{R \leftarrow \lambda x.\lambda y.paid(x, y\text{'s debts})\}$$

Définis et indéfinis

L'interaction entre désaccentuation et (in)définis suggère elle aussi que la désaccentuation est gouvernée par l'équivalence sémantique.

Définis désaccentués. Lorsqu'un défini apparaît dans un contexte désaccentué, il doit être résolu à son élément parallèle source. Les exemples (2.19a-f) illustrent cette observation, qui ne sont bien formés que lorsque le défini coréfére avec *the woman with the red hat*.

(2.19) Jon_2 likes [the woman with the red hat]$_1$
a. No, PETER [likes her_1].
b. No, PETER [likes (the neighbour's daughter)$_1$].
c. No, PETER [likes Sarah_1].
d. ⋆ No, PETER [likes her_4].
e. ⋆ No, PETER [likes Mary_4].
f. ⋆ No, PETER [likes him].

Ces cas rappellent les cas de désaccentuation faisant intervenir l'inférence (cf. 2.5a-c) au sens où le matériel désaccentué au lieu de répéter une partie de la clause source, introduit une information nouvelle. De plus là aussi, il y a une relation d'implication entre proposition source et proposition cible. Les paires bien formées de (2.19) ne sont acceptables que dans un contexte ou le défini désaccentué (*her/the neighbour's daughter/Sarah*) coréfère avec *the woman with the red hat*. Or si tel est le cas, *Jon likes the woman with the red hat* implique *John likes his wife/her/the neighbour's daughter/Sarah* et si l'implication appropriée est choisie, les équations seront solubles. Par exemple pour (2.19d), la proposition source peut être la proposition impliquée *Jon likes Sarah*. De cette façon, si le contexte est approprié, la contrainte de redondance sur les énoncés désaccentués peut être satisfaite.

Indéfinis. Examinons maintenant le cas où un pronom désaccentué dans la proposition cible est parallèle à un indéfini e.g.,

(2.20) a. Jon bought [a new car]^1. No, Peter [bought it_1].
b. Jon bought [a new car]^1. Yes, Jon [bought it_1].
c. Jon bought [a new car]^1 but Peter didn't [buy it_1].
d. Jon bought [a new car]^1, Peter [bought it_1] and Alex [bought it_1].

Une description indéfinie peut jouer le rôle d'un *terme référentiel* c'est-à-dire, peut permettre de référer à un objet spécifique (cf. [104]). Dans ce cas, l' indéfini signale la nouveauté (le locuteur pense que l'objet décrit est nouveau pour son interlocuteur) plutôt que la non-unicité. On parlera de *défini spécifique*.

Il s'ensuit que les exemples (2.20a–d) sont ambigus entre une lecture spécifique et une lecture non-spécifique de l'indéfini *a new car*. Dans le premier cas (interprétation spécifique), l'indéfini dénote une entité unique saillante dans le contexte. Appelons la e_1. Dans un tel contexte, *Jon bought a new car* implique *Jon bought* e_1 (où e_1 est la représentation sémantique de e_1). Si l'on prend cette proposition comme étant la proposition source, la clause désaccentué est validée par la contrainte de redondance pourvu que le pronom «it» soit résolu à la seule voiture saillante dans le contexte, c'est-à-dire à e_1.

Dans le second cas, il n'y a pas de voiture saillante unique dans le contexte si bien que la proposition source *Jon bought* e_1 ne peut être inférée. En outre, si l'on prend pour proposition source la proposition explicitement réalisée *Jon bought a new car*, aucune résolution du pronom ne peut être produite qui permette de résoudre les contraintes équationnelles implémentant la contrainte de redondance :

$$R(j) = \exists x.(car(x) \wedge bought(j,x))$$
$$R(p) = bought(p, ??)$$

Par ailleurs, si la clause désaccentuée contient une anaphore de propriétés (*one anaphora*) plutôt qu'un pronom, le discours devient acceptable même si l'indéfini dans la proposition source n'est pas spécifique.

(2.21) Jon bought (a red car)[1] but PETER [bought a] BLUE [one].

Dans ce cas, les équations imposées par la contrainte de redondance sont :

$$R(j, red) = \exists x[car(x) \wedge red(x) \wedge buy(j, x)]$$
$$R(p, blue) = \exists x[car(x) \wedge blue(x) \wedge buy(p, x)]$$

qui a pour solution :

$$\{R \leftarrow \lambda y.\lambda P.\exists x[car(x) \wedge P(x) \wedge buy(y, x)]\}$$

En bref, l'analyse unificationnelle prédit correctement qu'un pronom ne peut pas être désaccentué s'il est parallèle à un indéfini non spécifique ; et qu'il peut l'être s'il est parallèle à un indéfini spécifique.

2.2.3 Focus

Comme nous l'avons vu en section 1.5.2, il est généralement admis depuis [98] que l'interprétation des énoncés à focus (c'est-à-dire d'énoncés contenant des proéminences prosodiques) fait intervenir un objet théorique communément baptisé «Valeur de Focus» (VF). Rappelons que la valeur de focus est l'ensemble des objets sémantiques qui peut être obtenu en substituant au focus sémantique un objet de type approprié, c'est-à-dire compatible avec le type du reste de l'expression représentant la phrase contenant le focus. Par exemple, la valeur de focus pour (2.22a) est (2.22b), l'ensemble des propriétés de la forme *liking* y où y représente un individu (wff_τ dénote l'ensemble des formules bien formées de type τ).

(2.22) a. Jon only likes MARY
b. $\{\lambda x.l(x, y) \mid y \in wff_e\}$

Une seconde hypothèse usuelle est que certains éléments linguistiques **varient avec le focus** au sens où l'interprétation d'un énoncé contenant ces éléments variera avec son focus (si le focus change, l'interprétation de l'énoncé change). Par exemple dans (2.23a-b), l'opérateur lié au focus *only* varie avec le focus si bien que la différence de focus entre (2.23a) et (2.23b) induit une différence de sens entre les deux énoncés : dans un monde où Jon a présenté Paul à Mary et à Sarah, et aucune autre présentation n'a pris place, (2.23a) est faux tandis que (2.23b) est vrai.

(2.23) a. Jon only introduced Paul to MARY
b. Jon only introduced PAUL to Mary

Pour rendre compte de ces variations, la sémantique des éléments liés au focus (e.g. opérateurs de focus, adverbes quantificationels) est posée comme étant une fonction de la valeur de focus. Comme cette valeur de focus varie avec le focus, la co-variation entre focus et interprétation globale est ainsi assurée. L'exemple suivant illustre cette idée. Supposons que le sens de *only* soit donné par la règle suivante :

$$[\text{GN only GV}] \hookrightarrow \forall P[P \in VF \wedge P(GN') \rightarrow P = GV']$$

GN', GV' représentent le sens du GN et du GV respectivement tandis que VF représente la valeur de focus du GV. Cette règle dit donc que toutes les propriétés dans la valeur de focus du GV qui sont vraies de la dénotation du GN sujet doivent être la propriété dénotée par le GV. Par exemple, étant donnée la valeur de focus (2.22b) pour (2.22a), cette règle impose la sémantique suivante pour (2.22a) :

$$\forall P[P \in \{\lambda x.l(x,y) \mid y \in \mathit{wff}_e\} \wedge P(j) \rightarrow P = \lambda x.l(x,m)]$$

En français : la seule propriété de la forme *aimer quelqu'un* qui soit vraie de la dénotation j de *Jon* est la propriété d' *aimer Mary.*

Revenons à l'interaction entre focus et désaccentuation. La valeur de focus est une fonction du focus si bien que si le focus est indéterminé, la valeur de focus l'est aussi. Ceci est le cas pour les énoncés désaccentués où, dans la partie désaccentuée de l'énoncé, aucun élément n'est marqué prosodiquement et où pourtant au plan sémantique, il arrive qu'un élément fonctionne comme un focus (cet élément co-varie avec l'interprétation globale de l'énoncé). Prenons par exemple le dialogue (2.24) :

(2.24) a. Jon only likes MARY.
b. No, PETER [only likes Mary.]

Dans (2.24b), l'opérateur de focus *only* quantifie sur les propriétés de la forme *liking somebody* si bien que le sens de (2.24b) est (2.25a). Mais en l'absence de focus identifiable, la valeur de focus ne peut être déterminée. Aussi, sans traitement sémantique complémentaire, la représentation sémantique obtenue pour (2.24b) est (2.25b).

(2.25) a. $\forall P[P \in \{\lambda x.l(x,y) \mid y \in \mathit{wff}_e\} \wedge P(p) \rightarrow P = \lambda x.l(x,m)]$
b. $\forall P[P \in VF \wedge P(p) \rightarrow P = \lambda x.l(x,m)]$

En résumé, la désaccentuation soulève le problème suivant : comment peut on déterminer la valeur de focus, et à partir de là, le domaine de quantification des opérateurs de focus tels que *only*, lorsqu'aucun focus n'est identifiable au niveau prosodique ? La contrainte de redondance fournit une réponse à cette question. Comme l'ont montré [121] et [165], le domaine de quantification des opérateurs de focus peut être vu comme étant une anaphore pragmatique. Dans le cas de la désaccentuation, la valeur de cette anaphore est contrainte par la contrainte de redondance : elle doit être identique à la valeur de son élément parallèle source. Ainsi pour (2.24), les contraintes équationnelles gouvernant la désaccentuation du GV *only likes Mary* sont :

$$R(j) = \forall P[P \in \{\lambda x.l(x,y) \mid y \in \mathit{wff}_e\} \wedge P(j) \rightarrow P = \lambda x.l(x,m)]$$
$$R(p) = \forall P[P \in VF \wedge P(p) \rightarrow P = \lambda x.l(x,m)]$$

Pour ces équations, l'UOS retourne la substitution unifiante suivante :

$$\{R \leftarrow \lambda z \forall P[P \in \{\lambda x.l(x,y) \mid y \in \mathit{wff}_e\} \wedge P(z) \rightarrow P = \lambda x.l(x,m)],$$
$$VF \leftarrow \{\lambda x.l(x,y) \mid y \in \mathit{wff}_e\}\}$$

La phrase (2.24b) se voit ainsi attribuée la représentation sémantique résolue :

$$R(p) = \forall P[P \in \{\lambda x.l(x,y) \mid y \in \mathit{wff}_e\} \wedge P(p) \rightarrow P = \lambda x.l(x,m)]$$

Ainsi, le traitement de la désaccentuation basé sur l'unification prédit qu'un opérateur de focus désaccentué peut parfois hériter sa valeur de focus de la phrase source. Il ne prédit pas en revanche, que la valeur de focus soit *systématiquement* héritée de la source en cas de désaccentuation du focus. Seulement qu'elle l'est parfois et plus précisément qu'elle l'est lorsque proposition source et proposition cible sont parallèles (Pour plus de détails sur ce point cf. [57]).

2.3 Travaux apparentés

Deux autres approches tentent d'expliquer les similitudes entre ellipse et désaccentuation par le biais d'un mécanisme interprétatif commun : la théorie du focus de Rooth et le traitement du parallélisme discursif de [87]. Nous examinons maintenant ces deux approches et les comparons avec l'approche unificationnelle.

2.3.1 La théorie du focus de Rooth

[164] présente une analyse des énoncés désaccentués qui se base sur la théorie du focus présentée dans [164]. La différence essentielle entre cette approche et l'approche unificationnelle réside dans la façon dont la valeur de focus est déterminée. Alors que l'analyse unificationnelle donne une définition *syntaxique* de cette valeur par le biais d'une formule du lambda calcul, l'approche présentée par Rooth en donne une définition *sémantique* c'est-à-dire, en termes d'objets dans un modèle. Ainsi par exemple, la valeur de focus de la seconde clause dans (2.26) sera (2.26a) dans l'approche unificationnelle et (2.26b) dans l'approche de Rooth (E est le domaine des individus).

(2.26) Jon saw Mary. No, PETER [saw Mary].
  a. $\{\lambda x.saw(x, mary) \mid x \in \mathit{wff}_e\}$
  b. $\{saw(x,mary) \mid x \in E\}$

En général, la définition de la valeur de focus donnée par Rooth permet les mêmes prédictions que les contraintes unificationnelles. Cependant il y a une différence entre les deux approches qui émerge le plus visiblement dans les cas d'interaction entre ellipse et focus. Comme l'a montré [119], l'analyse de Rooth achoppe sur des exemples tels que (2.27).

(2.27) I only went to TANGLEWOOD because you did.

L'exemple (2.27) contraste la proposition *I went to Tanglewood because you went to Tanglewood* avec des propositions de la forme *I went to x because you went to x* – et non avec des propositions de la forme *I went to x because you went to y.* Malheureusement, l'analyse de Rooth ne peut rendre compte de cette observation qu'en postulant un mouvement obligatoire, au niveau de la forme logique, des syntagmes focalisés – en l'occurrence, un mouvement du GN *Tanglewood* . En d'autres termes, dans de tels cas, le focus doit obligatoirement être déplacé afin d'obtenir l'interprétation correcte. Ceci est malheureux puisque par ailleurs, on sait que le déplacement d'un focus n'est pas sujet aux contraintes régissant les déplacements. Ainsi par exemple, pour obtenir les interprétations correctes pour les exemples (2.28a–b), un groupe adjectival devrait être déplacé hors du GN et un GN extrait d'un ilôt, deux déplacements qui sont traditionnellement considérés comme illégitimes.

(2.28) a. I only saw more PINK edsels than you did.
  b. I only contacted the person who chairs the ZONING BOARD before you did.

Le même problème apparaît dans le contexte des énoncés désaccentués. Prenons par exemple (1.8) répété ici comme (2.29). Pour rendre compte du fait que la clause désaccentuée n'est validée que par des propositions de la forme *I'll R you if you want me to R you* – par opposition à *I'll* R_1 *you if you want me to* R_2 *you* – le verbe *KISS* doit être déplacé, ce qui encore une fois n'est pas un déplacement légitime.

(2.29) I'll [help you]^1 if you [want me to$_1$]^2 and I'll KISS [you even if you] DON'T [want me to kiss you].

2.3.2 Le parallélisme discursif de Hobbs et Kehler

Une autre analyse uniforme des énoncés désaccentués et de l'ellipse du GV est proposée dans [87] qui présente une théorie générale du parallélisme discursif et montre que cette théorie procure à la fois une analyse détaillée de l'interaction entre ellipse du GV et anaphore, et une couverture empirique générale des ellipses et de la référence et en particulier, des énoncés désaccentués.

Dans l'approche de Hobbs et Kehler (H&K), le parallélisme du discours est défini en termes d'une co-récursion entre deux propositions inferées. Plus précisément, deux segments discursifs sont dans une relation de parallélisme s'ils décrivent des propriétés similaires, c'est-à-dire si deux propositions peuvent en être inférées qui ont des prédicats identiques et dont les arguments sont soit coréférentiels soit similaires. En gros, deux arguments sont similaires si leurs propriétés sont conceptuellement similaires.

Comme l'algorithme d'UOS de Huet, l'algorithme utilisé par H&K pour déterminer le parallélisme du discours sont des algorithmes de co-récursion sur deux formules logiques. Les deux algorithmes diffèrent par le niveau de détail auquel ils sont définis et par leur interaction avec le savoir linguistique et encyclopédique.

Alors que l'algorithme d'UOS de Huet est un algorithme concret (c'est-à-dire complètement défini et donc opérationnel) qui détermine un processus de décomposition co-récursive et d'élimination de variables visant à transformer un ensemble d'équations en un ensemble de formes résolues, l'algorithme présenté par H&K est un algorithme abstrait qui laisse certains détails non spécifiés.

Par ailleurs, alors que l'UOS est une opération strictement syntaxique opérant sur des formules logiques, l'algorithme de H&K intègre à la fois un savoir linguistique (sous la forme de liens coréférentiels) et un savoir encyclopédique (utilisé pour prouver la similarité de deux entités). Comme H&K l'ont montré, l'intégration de l'information coréférentielle permet une analyse très fine de l'interaction entre GVE et anaphores nominales. En particulier, elle permet de modéliser les «puzzles éliminatifs de l'ellipse» (cf. ci-dessous). D'autre part, l'intégration d'un savoir encyclopédique permet une analyse fine et compositionnelle de la notion de similarité.

En résumé, l'un des algorithmes a une base mathématique et informatique bien comprise tandis que l'autre intègre plusieurs niveaux de connaissances permettant ainsi une analyse plus fine des données linguistiques. La question qui mériterait d'être approfondie mais que je laisserai ici sans réponse est la suivante : l'algorithme d'UOS de Huet peut il être modifié de façon à intégrer le savoir additionnel utilisé dans l'approche de H&K ? Et conjointement : la version entièrement spécifiée de l'algorithme de H&K serait elle foncièrement différente de cet algorithme d'UOS amendé ?

Les deux approches (H&K *versus* Unification) diffèrent également dans leurs prédictions empiriques aussi bien pour l'ellipse du GV que pour les énoncés désaccentués.

Pour l'ellipse du GV, H&K montrent que leur analyse rend compte du fait que certaines interprétations qui sont *prima facie* possibles ne sont de fait pas disponibles. Pour (2.30) par exemple, seules trois des quatre interprétations *a priori* possibles sont disponibles puisque (2.30b) n'est pas une interprétation possible de (2.30).

(2.30) John^1 says that he$_1$ likes his$_1$ mother and Peter does too.
- a. Peter^2 says that he$_1$ likes his$_1$ mother.
- b. $\star$ Peter^2 says that he$_1$ likes his$_2$ mother.
- c. Peter^2 says that he$_2$ likes his$_1$ mother.
- d. Peter^2 says that he$_2$ likes his$_2$ mother.

Alors que ni l'approche unificationnelle, ni celle de Rooth ne permet de rendre compte de cette observation, l'algorithme de reconstruction de H&K prédit cette interprétation manquante à partir de l'information coréférentielle utilisée.

Réciproquement, la théorie de H&K sur le parallélisme discursif n'a jusqu'à présent été appliquée ni aux interactions complexes entre ellipse, anaphore et quantification, ni à celles entre désaccentuation et anaphore – deux sortes d'interaction qui comme nous venons de le voir, sont largement couvertes par l'analyse unificationnelle.

2.4 Conclusion

Nous avons montré que l'analyse de l'ellipse présentée par DSP s'étend également aux énoncés désaccentués. Cette analyse fait des prédictions fortes quant à l'interaction de la désaccentuation avec les anaphores, prédictions qui sont confirmées par les données. En outre, parce qu'elle place une seule et même contrainte sur l'interprétation des deux phénomènes (l'ellipse et la désaccentuation), elle explique les similitudes sémantiques observées entre ces deux phénomènes comme en particulier, l'ambiguïté lâche/stricte.

Chapitre 3

UOS et interprétations lâches

Traditionnellement, les interprétations lâches sont traitées dans le cadre des théories de l'ellipse verbale. Ainsi par exemple, l'un des arguments forts avancé par DSP pour leur analyse de l'ellipse du GV est qu'elle prédit plutôt qu'elle ne stipule l'ambiguïté lâche/stricte qui résulte de l'interaction entre ellipses et pronoms.

Néanmoins, l'ambiguïté lâche/stricte est un phénomène très général qui n'est pas nécessairement lié à un contexte elliptique. Dans ce chapitre, nous montrons que l'analyse modifiée de DSP présentée au chapitre 1 fournit une théorie généralisée de l'interprétation lâche. L'idée clé est que la contrainte de redondance qui, comme nous l'avons vu au chapitre 2, permet une modélisation adéquate de l'interprétation des énoncés désaccentués, est en fait une contrainte d'ordre plus général régissant l'interprétation des structures parallèles. Partant de l'hypothèse que les interprétations lâches sont induites par le parallélisme sémantique, nous montrons que cette contrainte fait les prédictions empiriques appropriées. Outre l'ambiguïté lâche/stricte des ellipses du GV et des énoncés désaccentués discutée dans les chapitres précédents, elle prédit en effet l'interprétation lâche des phrases dites de «paycheck» ainsi que celle des anaphores relationnelles, des anaphores évènementielles et des anaphores de propriété («one-anaphora»). Par ailleurs, nous montrons qu'elle permet une modélisation appropriée de l'interaction entre ambiguïté lâche/stricte, phénomènes de liage et phénomènes de quantification. Nous concluons le chapitre par une comparaison de l'approche unificationnelle avec des approches apparentées : l'approche basée sur la sémantique dynamique de [81] et celle basée sur la reconstruction syntaxique de [51].

3.1 Les données

Rappelons (cf. section 1.4) qu'une interprétation lâche met en jeu deux phrases : la phrase **source** et la phrase **cible** c'est-à-dire la phrase contenant une anaphore ou une ellipse (la **forme dépendante cible**). Lorsque l'antécédent de la forme dépendante cible contient lui aussi une forme dépendante (la **forme dépendante source**), une interprétation lâche est possible. Dans ce cas, l'interprétation de la forme dépendante cible diffère de l'interprétation de son antécédent de par l'interprétation de la forme dépendante source. Par exemple, dans :

(3.1) Jon^1 [washes his$_1$ car]^2. Peter does$_2$ too.

l'ellipse du GV *does* a, entre autres, l'interprétation lâche *washes Peter's car* (c'est même l'interprétation privilégiée). L' interprétation est lâche puisque dans la proposition source *Jon washes his car* la forme dépendante source *his* dénote *Jon* tandis que dans la proposition cible *Peter does too*, elle est ré-interprétée comme dénotant *Peter*.

Bien qu'il soit communément associé avec celui de l'ellipse verbale, le phénomène de l'identité lâche apparaît dans un grand nombre de configurations et en particulier : dans les énoncés désaccentués (exemple 3.2) ; dans les phrases de «paycheck» (exemple 3.3) ; dans les énoncés faisant intervenir une ellipse verbale comme forme dépendante source (exemples 3.4 et 3.5) ; dans les éléments impliquant des arguments implicites (exemples 3.6 et 3.7) ; et en connexion avec les anaphores événementielles (exemples 3.8 et 3.9) et les anaphores de propriété (*one*-anaphora, exemple 3.10)[27].

(3.2) Jon^1 took his$_1$ wife to the station. No, BILL [took his wife to the station]. *(Bill took Bill's wife to the station)*

(3.3) Jon^1 spent [his$_1$ paycheck]^2 but Peter saved it$_2$. *(Peter saved Peter's paycheck)*

(3.4) I'll [help you]^1 if you [want me to$_1$]^2. I'll kiss you even if you don't$_2$. *(I'll kiss you even if you don't want me to kiss you)*

(3.5) When Harry [drinks]^1, I always conceal [my belief that he shouldn't$_1$]^2. When he gambles, I can't conceal it$_2$. *(When he gambles, I can't conceal my belief that he shouldn't gamble)*

(3.6) Jon went to a local bar to watch the Super-bowl, and Bob did too. *(Bob went to a bar local to Bob)*

(3.7) George drove to the nearest hospital, and Fred did too. *(Fred drove to the hospital nearest to Fred)*

(3.8) Jon^1 got shot by his$_1$ father. That happened to Bob too. *(Bob got shot by Bob's father)*

(3.9) Jon^1 kissed his$_1$ wife, and Bill followed his exemple (Dahl, 1972) *(Bill kissed Bill's wife)*

(3.10) Although Jon^1 bought a picture of his$_1$ son, Bill snapped one himself. *(Bill snapped a picture of Bill's son)*

Une interprétation lâche peut donc résulter de diverses combinaisons de formes dépendantes sources et de formes dépendantes cibles.

Quelles sont les contraintes régissant l'interprétation lâche ? Le parallélisme semble ici jouer une rôle fondamental au sens où une interprétation lâche ne paraît possible que lorsque l'antécédent de la forme dépendante source a une contrepartie dans la proposition cible (cf. [39]). Ainsi, dans l'exemple (3.11), *Jon* a un élément parallèle (*Bill*) dans la proposition source et l'énoncé a une interprétation lâche. Cette interprétation lâche est en revanche absente pour (3.12) où *Bill* n'a pas d'élément parallèle dans la proposition source.

(3.11) The policeman who arrested Bill^1 [forgot to read him$_1$ his$_1$ rights]^2 and so did$_2$ the policeman who arrested Jon.

(3.12) The policeman who arrested Bill^1 [forgot to read him$_1$ his$_1$ rights]^2 and so did$_2$ Peter.

L'ambiguïté lâche/stricte n'est pas pour autant systématique dans les structures parallèles car d'autres phénomènes linguistiques et extra-linguistiques peuvent bloquer l'une ou l'autre des deux interprétations. L'interaction avec la quantification par exemple exclut une interprétation stricte de l'énoncé (3.13). Dans l'exemple (3.14), c'est une interaction avec les contraintes de la théorie du liage qui exclut une interprétation stricte. Enfin, l'exemple (3.15) montre que l'ambiguïté lâche/stricte est sensible à la distinction entre pronom et nom propre : la forme dépendante source doit être un pronom (et non un nom propre) pour qu'une interprétation lâche soit possible.

[27]Le texte entre parenthèses représente l'interprétation pertinente c'est-à-dire l' interprétation lâche. Pour la plupart, les exemples sont tirés de [110].

(3.13) Every^1 man believes he$_1$ is a fool. No, PETER believes he is a fool. (lâche)

(3.14) Mary persuaded Jon^1 to shave himself$_1$. No, Mary persuaded PETER to shave himself. (lâche)

(3.15) Jon's mother loves Jon and Bill's mother does too (strict)

Une théorie des interprétations lâches doit donc être suffisament générale pour englober les diverses configurations illustrées par les exemples (3.2-3.10) ; elle doit rendre compte de la contrainte de parallélisme illustrée par les exemples (3.11) et (3.12) ; et elle doit correctement modéliser les interactions entre ambiguïté lâche/stricte, quantification, théorie du liage et catégorisation syntaxique (exemples 3.13-3.15). Dans le reste de ce chapitre, nous verrons que la contrainte de redondance présentée dans les chapitres précédents implémente une théorie des interprétations lâche conforme à ces desiderata. Un élément clé de cette analyse est l'utilisation de l'UOS comme outil de résolution pour les équations posées. C'est l'utilisation de ce mécanisme particulier qui permet de faire des prédictions appropriées au plan linguistique au sens où les solutions multiples générées par l'UOS reflètent (en partie) l'ambiguïté lâche/stricte de la langue naturelle.

3.2 Les pronoms «paycheck»

Dans l'énoncé (3.16) ci-dessous, le pronom *it* apparaissant dans la seconde clause a une interprétation lâche et peut être interprété comme signifiant *Peter's paycheck*.

(3.16) Jon^1 spent [his$_1$ paycheck]^2 but Peter saved it$_2$.

Dans la littérature, ces pronoms sont connus sous le nom de *pronoms paycheck* et sont généralement traités comme introduisant un défini dont la restriction est fixée pragmatiquement. Ainsi [30] par exemple propose de représenter un pronom paycheck par le lambda–terme :

$$\text{Pro} \quad \rightsquigarrow \lambda Q.\exists x[P(x) \wedge \forall y[P(y) \rightarrow y = x] \wedge Q(x)] \quad \text{avec } P, Q \in \mathit{wff}_{(e,t)}$$

où la restriction du défini est représentée par une variable libre P de type $e \rightarrow t$. Si l'on prend en outre pour hypothèse que, comme les ellipses du GV, les pronoms paycheck sont soumis à la contrainte de redondance régissant l'interprétation des structures parallèles, les contraintes imposées sur les représentations sont alors les suivantes[28] :

$$\begin{array}{ll} \exists! x[pc\_of(x,j) \wedge sp(j,x)] & = A(j, sp) \\ \exists! x[P \wedge sa(p,x)] & = A(p, sa) \end{array}$$

Etant données ces équations, deux résolutions sont possibles pour les variables P et A :

$$\begin{array}{ll} \textit{Lâche} & \{P \leftarrow \lambda y.pc\_of(y,p), A \leftarrow \lambda y.\lambda O.\exists! x[pc\_of(x,y) \wedge O(y,x)]\}\} \\ \textit{Stricte} & \{P \leftarrow \lambda y.pc\_of(y,j), A \leftarrow \lambda y.\lambda O.\exists! x[pc\_of(x,j) \wedge O(y,x)]\} \end{array}$$

La proposition cible peut donc correctement être associée soit à une interprétation stricte soit à une interprétation lâche. Toute autre résolution du pronom paycheck est en outre exclue par les contraintes équationnelles.

[28] Nous abrégeons $\lambda Q.\exists x[P(x) \wedge \forall y[P(y) \rightarrow y = x] \wedge Q(x)]$ par $\lambda Q.\exists! x[P(x) \wedge Q(x)]$.

	Phrase Source	Phrase Cible
Configuration	$\mathrm{NP}^j, \ldots, [_{VP}, \ldots, \boxed{\mathrm{PRO}}_j, \ldots,]^i$	$, \ldots, \boxed{\mathrm{VPE}}_i, \ldots,$
Exemple	*Jon*$_j$ *[*$_{VP}$ *washes* $\boxed{\textit{his}}_j$ *car]*^{i}	*Peter* $\boxed{\textit{does}}_i$ *too.*
Configuration	$\mathrm{VP}^j, \ldots, [_{VP}, \ldots, \boxed{\mathrm{VPE}}_j, \ldots,]^i$	$, \ldots, \boxed{\mathrm{VPE}}_i, \ldots,$
Exemple	*I'll [help you]*^{j}*, if you [*$_{VP}$ *want me* $\boxed{\textit{to}}_j$*]*^{i}	*I'll kiss you even if you* $\boxed{\textit{don't}}_i$*.*
Configuration	$\mathrm{NP}^j, \ldots, [_{NP}, \ldots, \boxed{\mathrm{PRO}}_j, \ldots,]^i$	$, \ldots, \boxed{\mathrm{PRO}}_i, \ldots,$
Exemple	*Jon*$_j$ *spent [*$_{NP}$ $\boxed{\textit{his}}_j$ *paycheck]*^{i}	*but Peter saved* $\boxed{\textit{it}}_i$*.*
Configuration	$\mathrm{VP}^j, \ldots, [_{NP}, \ldots, \boxed{\mathrm{VPE}}_j, \ldots,]^i$	$, \ldots, \boxed{\mathrm{PRO}}_i, \ldots,$
Exemple	*When Harry [drinks]*$_j$*, I conceal* *[*$_{NP}$ *my belief that he* $\boxed{\textit{shouldn't}}_j$*]*^{i}	*When he gambles,* *I can't conceal* $\boxed{\textit{it}}_i$*.*

TAB. 3.1 – Ellipses, Pronoms et Interprétation lâche

3.3 L'ellipse comme élément dépendant source

Jusqu'à présent, nous avons uniquement considéré les cas d'interprétation lâche dans lesquels la forme dépendante source est un pronom. Néanmoins, comme l'exemple (3.17) ci-dessous l'illustre, la forme dépendante source peut aussi être une ellipse du GV.

(3.17) I'll [help you]^1 if you [want me to$_1$]^2. I'll kiss you even if you don't$_2$.

(3.18) When Harry [drinks]^1, I always conceal [my belief that he shouldn't$_1$]^2. When he gambles, I can't conceal it$_2$.

Dans l'exemple (3.17), la forme dépendante cible est une ellipse du GV dont l'antécédent contient une ellipse du GV – la forme dépendante source est donc une ellipse du GV. De même, en (3.18), la forme dépendante cible est un pronom avec une ellipse du GV comme forme dépendante source. Les deux exemples ont une interprétation lâche :

(3.19) I'll [help you]^1 if you [want me to$_1$]^2. I'll kiss you even if you don't$_2$ (*you don't want me to kiss you*)

(3.20) When Harry [drinks]^1, I always conceal [my belief that he shouldn't$_1$]^2. When he gambles, I can't conceal it$_2$. (*I can't conceal the belief that he shouldn't gamble*)

Ces exemples montrent qu'une interprétation lâche est possible que les formes dépendantes source et cible soient des ellipses verbales ou des anaphores pronominales. Comme le tableau 3.18 l'indique, on a donc en tout quatre combinaisons possibles dont deux ont déjà été discutées. Nous montrons maintenant comment l'analyse unificationnelle permet de rendre compte des deux cas restant, à commencer par l'exemple (3.19).

Etant donné les éléments parallèles *help* et *kiss*, la contrainte de redondance impose les équations suivantes :

$$h(i, you) \leftarrow wt(you, h(i, you)) = A(h)$$
$$k(i, you) \leftarrow P(you)) = A(k)$$

Etant données ces équations, une résolution lâche de l'ellipse représentée par P est donnée par la substitution unifiante :

$$\{P \leftarrow \lambda x.wt(i, k(i, x)),\ A \leftarrow \lambda R.(R(i, you) \leftarrow wt(you, R(i, you)))\}$$

La contrainte de redondance est donc compatible avec une interprétation lâche des énoncés contenant une ellipse comme élément dépendant source. De même, l'énoncé (3.20) sera analysé de la façon suivante. Les éléments parallèles sont *drinks* et *gambles* si bien que la contrainte de redondance impose les équations suivantes :

$$dk(h) \rightarrow \exists x[bel(x, sn(dk(h))) \wedge hide(i, x)] = A(dk)$$
$$gb(h) \rightarrow \exists x[P(x) \wedge hide(i, x)] = A(gb)$$

Une interprétation lâche du pronom paycheck donne les équations :

$$dk(h) \rightarrow \exists x[bel(x, sn(dk(h))) \wedge hide(i, x)] = A(dk)$$
$$gb(h) \rightarrow \exists x[bel(x, sn(gb(h) \wedge hide(i, x)] = A(gb)$$

qui ont pour solution la substitution :

$$\{A \leftarrow \lambda R.(R(h) \rightarrow \exists x[bel(x, sn(R(h))) \wedge hide(i, x)])$$

Dans les deux cas, une interprétation lâche de la forme dépendante cible donne lieu à un système équationnel solvable qui satisfait la contrainte de redondance.

3.4 Autres constructions lâches

Arguments implicites. Le type de construction illustré par l'énoncé (3.21) ci-dessous ne pose pas de problème particulier dès lors que les arguments implicites sont inclus dans la représentation sémantique[29].

(3.21) First, Jon went to a local bar to watch the Super-bowl, and then BOB went to a local bar to watch the Super-bowl too. *(Bob went to a bar local to Bob/Jon)*

Ainsi par exemple, si l'adjectif *local* est associé à la représentation sémantique :

$$\lambda P.\lambda x.local(x, Y) \wedge P(x)$$

où Y représente une anaphore[30] dont la valeur est déterminée par le processus de résolution anaphorique. Alors ne sont possibles que deux résolutions qui satisfont les équations posées par la contrainte de redondance :

$$\begin{array}{ll} \textit{Résolution stricte} & Y = j \\ \textit{Résolution lâche} & Y = b \end{array}$$

[29]Exemples tirés de [110].

[30]De la même manière, un pronom dans la grammaire de Montague a pour représentation le lambda-terme $\lambda P.P(x)$ où x est une variable liée par le processus de résolution des anaphores.

La première résolution donne les équations :

$$\exists x[bar(x) \wedge local(x,j) \wedge went(j,x,watch(j,sb))] = A(j)$$
$$\exists x[bar(x) \wedge local(x,j) \wedge went(j,x,watch(j,sb))] = A(b)$$

qui ont pour solution :

$$\{A \leftarrow \lambda z.\exists x[bar(x) \wedge loc(x,j) \wedge went(z,x,watch(z,sb))]\}$$

La deuxième résolution induit les équations :

$$\exists x[bar(x) \wedge local(x,j) \wedge went(j,x,watch(j,sb))] = A(j)$$
$$\exists x[bar(x) \wedge local(x,b) \wedge went(j,x,watch(j,sb))] = A(b)$$

qui ont pour solution :

$$\{A \leftarrow \lambda z.\exists x[bar(x) \wedge loc(x,z) \wedge went(z,x,watch(z,sb))]\}$$

Une autre construction recensée par [110] est l'anaphore sur les éventualités qui peut être soit pronominale (3.22), soit définie (3.23).

(3.22) Jon got shot by his father. That happened to Bob too. (*Bob got shot by Bob/Jon's father*)

(3.23) Jon shot his father, and Bill followed his exemple (Dahl, 1972) (*Bill shot Jon/Bill's father*)

Sans décider de la représentation sémantique exacte qu'il est justifié d'attribuer aux expressions *that happened* et *followed x's exemple*, il semble raisonnable de traiter ces expressions comme des anaphores, c'est-à-dire comme des expressions dont la signification est déterminée par le contexte. Dans cette perspective, la seconde clause de (3.22) et (3.23) peut être représentée par $R(b)$ où R est une variable dont la valeur est déterminée par le processus de résolution anaphorique. Deux résolutions satisfaisant la contrainte de redondance sont alors possibles :

$$\begin{array}{ll} \textit{Résolution lâche} & R = \lambda z.shot(f(z),z) \\ \textit{Résolution stricte} & R = \lambda z.shot(f(j),z) \end{array}$$

La première résolution induit la contrainte de redondance :

$$shot(f(j),j) = A(j)$$
$$shot(f(b),b) = A(b)$$

Ces équations sont solvables ($\{A \leftarrow \lambda z.shot(f(z),z)\}$) et l'interprétation obtenue est l'interprétation lâche. Par contraste, la deuxième résolution donne lieu aux contraintes équationnelles :

$$shot(f(j),j) = A(j)$$
$$shot(f(j),b) = A(b)$$

qui ont pour solution $\{A \leftarrow \lambda z.shot(f(j),z)\}$ et justifient une interprétation stricte.

L'anaphore adjectivale permet elle aussi une ambiguïté lâche/stricte :

(3.24) Although Jon bought a picture of his son, Bill snapped one himself.

On peut modéliser ce fait en traitant l'anaphore adjectivale comme les pronoms paycheck, c'est-à-dire comme des définis dont la restriction est déterminée par un processus pragmatique. Dans cette perspective, les équations pour (3.24) seront :

$$\exists x[pic\_of(x,j'sson) \wedge bo(j,x)] = A(j,bo)$$
$$\exists x[P(x) \wedge sna(b,x)] = A(b,sna)$$

L'interprétation lâche est donnée par la résolution :

$$P = \lambda y.\exists x[pic\_of(x, y'sson)]$$

et l'interprétation stricte par la résolution :

$$P = \lambda y.\exists x[pic\_of(x, j'sson)]$$

3.5 Interaction avec syntaxe et quantification

Comme nous venons de le voir, l'analyse unificationnelle prédit une ambiguïté lâche/stricte systématique pour toute anaphore apparaissant dans une structure parallèle. Néanmoins, des interactions avec des phénomènes linguistiques indépendants peuvent avoir pour effet d'éliminer l'une ou l'autre de ces interprétations. Dans cette section, nous examinons quelques-unes de ces interactions et montrons comment l'unification colorée d'ordre supérieur (UCOS, cf. section 1.5.1) permet d'en rendre compte.

Reconstruction sémantique et syntaxe

Comme l'exemple (3.15) répété ici en (3.26) l'illustre, l'information catégorielle peut influencer l'ambiguïté lâche/stricte. Ainsi, bien que (3.25) ait deux interprétations, l'une stricte et l'autre lâche, (3.26) n'a qu'une interprétation stricte.

(3.25) Jon^1's mother loves him$_1$ and Bill's mother does too (lâche/strict)

(3.26) Jon's mother loves Jon and Bill's mother does too (stricte)

Pour l'analyse unificationnelle standard (c'est-à-dire non colorée), cet exemple pose problème puisque (3.25) et (3.26) auront une représentation sémantique identique.

Dans le cadre de l'unification colorée en revanche, les couleurs peuvent être utilisées pour créer une interface entre le processus de construction sémantique et les autres modules linguistiques en particulier, le module syntaxique. Au chapitre 1, nous avons vu que pour éviter la sur-génération, on pouvait introduire la restriction suivante :

> **ROP 2.** Les occurrences de symboles représentant des éléments contrastifs (c'est-à-dire des éléments dont l'élément parallèle est (i) réalisé phonétiquement et (ii) sémantiquement distinct) ne peuvent apparaître ni dans la valeur attribuée à R, la représentation sémantique de l'ellipse verbale, ni dans la valeur attribuée à C, la sémantique partagée par la proposition cible et par la proposition source.

Appliquée à l'exemple (3.25) ci-dessus, cette restriction exige que les équations implémentant la contrainte de redondance soient les suivantes :

$$l(m(j_{\mathsf{c}}), j) = C_{\mathsf{nc}}(j_{\mathsf{c}})$$
$$R_{\mathsf{nc}}(m(b_{\mathsf{c}})) = C_{\mathsf{nc}}(b_{\mathsf{c}})$$

Pour ce système d'équations, l'UCOS donne, comme il se doit, deux solutions :

$$\{C_{\mathsf{nc}} \leftarrow \lambda x.l(m(x), x), R_{\mathsf{nc}} \leftarrow \lambda x.l(x, c(x))\}$$
$$\{C_{\mathsf{nc}} \leftarrow \lambda x.l(m(x), x), R_{\mathsf{nc}} \leftarrow \lambda x.l(x, c(j_{\mathsf{nc}}))\}$$

L'absence d'interprétation lâche pour l'exemple (3.26) est capturée de la façon suivante. Alors que comme nous venons de le voir, la couleur d'un pronom est laissée indéterminée (sa représentation sémantique est un Lambda terme dont la couleur est une variable), la couleur d'un nom propre non contrastif (c'est-à-dire qui n'a pas d'élément parallèle explicitement réalisé) sera celle des éléments non contrastifs c'est-à-dire nc. Etant donné cette contrainte, les équations pour l'exemple (3.26) sont :

$$\begin{array}{l} l(m(j_{\mathsf{c}}), j_{\mathsf{nc}}) = C_{\mathsf{nc}}(j_{\mathsf{c}}) \\ R_{\mathsf{nc}}(m(b_{\mathsf{c}})) = C_{\mathsf{nc}}(b_{\mathsf{c}}) \end{array}$$

pour lesquelles, l'UCOS donne une solution unique à savoir la substitution permettant une interprétation stricte de l'énoncé :

$$\{C_{\mathsf{nc}} \leftarrow \lambda x.l(m(x), j_{\mathsf{nc}}), R_{\mathsf{nc}} \leftarrow \lambda x.l(x, c(j_{\mathsf{nc}}))\}$$

Contraintes de portée

Les phénomènes de portée peuvent eux-aussi avoir un effet sur l'ambiguïté lâche/stricte.

(3.27) Every^1 man believes he$_1$ is a fool. No, PETER believes he is a fool. (*interprétation lâche uniquement*)

(3.28) Jon lost a^1 book and never got it$_1$ back. No, Peter lost a PEN and never got it back (*interprétation lâche uniquement*)

Dans (3.27) et (3.28), le pronom source a un antécédent quantifié. De plus, la portée de cet antécédent est restreinte à la proposition source si bien qu'elle n'inclut pas la proposition cible. Il s'ensuit que seule l'interprétation lâche est possible. L'interprétation stricte est exclue parce qu'elle mettrait en jeu une variable libre : la variable liée par l'antécédent quantificateur et apparaissant dans la représentation cible. Dans notre analyse, cette restriction interprétative résulte de contraintes indépendamment justifiées sur les substitutions qui imposent qu'une variable libre ne puisse devenir liée et vice-versa. Ainsi pour (3.27), la contrainte de redondance exige que les équations suivantes soient solvables (comme dans DSP, lorqu'un nom propre a pour élément parallèle un GN quantifié, nous utilisons une représentation avec levée des types pour les noms propres afin de préserver le parallélisme des types) :

$$\begin{array}{l} \forall x[m(x) \rightarrow bel(x, f(x))] = A(\lambda P.\forall x[m(x) \rightarrow P(x)]) \\ bel(p, f(y)) = A(\lambda P.P(p)) \end{array}$$

Pour obtenir l'interprétation stricte, la seconde équation doit être résolue comme suit :

$$\{A \leftarrow \lambda Q.Q(\lambda z.bel(z, f(x)), y \leftarrow x\}$$

Si l'on applique cette substitution à la première équation, on obtient :

$$\begin{array}{ll} A(\lambda P.\forall x[b(x) \rightarrow P(x)]) & = \lambda Q.Q(\lambda z.bel(z, f(x))(\lambda P.\forall x[b(x) \rightarrow P(x)]) \\ & = \lambda P.\forall x[b(x) \rightarrow P(x)](\lambda z.bel(z, f(x))) \end{array}$$

Or à ce stade, les restrictions sur les substitutions bien formées exigent que x soit renommé. Par conséquent, la variable x ne peut jamais être liée par le quantificateur source *every man*.

Contraintes de liage

Enfin l'exemple suivant illustre une interaction entre ambiguïté lâche/stricte et contraintes de liage.

(3.29) Mary persuaded Jon[1] to shave himself$_1$. No, Mary persuaded PETER to shave himself. (*lâche seulement*)

Ici seule l'interprétation lâche est possible. Une explication simple de ce fait est que l'interprétation stricte (*Peter shaved Jon*) résulterait en une structure qui viole la condition A de la théorie du liage selon laquelle tout pronom réflexif doit être lié dans sa catégorie gouvernante. Rappelons à ce propos que la contrainte de redondance proposée ici est une contrainte ayant un effet filtrant sur les résolutions possibles des anaphores et des ellipses. Comme en l'occurrence la seule résolution permise pour la seconde occurrence de *himself* par la théorie du liage est *Peter*, les équations implémentant la contrainte de redondance sont :

$$pe(m, sh(j, j)) = A(j)$$
$$pe(m, sh(p, p)) = A(p)$$

pour lesquelles la seule solution possible est $\{A \leftarrow \lambda x.pe(m, sh(x, x))\}$. En particulier, la solution qui donnerait une interprétation stricte à savoir, $\{A \leftarrow \lambda x.pe(m, sh(x, j))\}$ n'est pas une substitution unifiante puisqu'elle ne peut résoudre simultanément les deux équations.

3.6 Comparaison avec les approches apparentées

Dans ce paragraphe, nous comparons l'approche unificationnelle avec deux approches alternatives : la théorie de l'ellipse de Hardt qui repose sur l'utilisation d'une sémantique dynamique ; et l'approche basée sur les formes logiques de Fiengo and May, formulée dans le cadre de la théorie du gouvernement et du liage.

Une approche basée sur la sémantique dynamique. Dans l'approche de Hardt, une anaphore dénote une fonction d'un ensemble de contextes vers un ensemble d'objets sémantiques du type approprié. En particulier, une ellipse dénote une fonction d'un ensemble de contextes vers un ensemble de propriétés, c'est-à-dire d'objets de type $e \rightarrow t$. Comme le montre [55, 81], cette analyse prédit l'ambiguïté lâche/stricte : si l'antécédent d'une ellipse contient un pronom, la dénotation de ce pronom dans le contexte source peut différer de la dénotation dans le contexte cible.

Il y a au moins deux grandes différences entre notre analyse et celle de Hardt. D'une part, Hardt n'impose pas de contrainte de parallélisme sur l'identité lâche. Cela a des avantages et des inconvénients. Un avantage est que l'approche peut rendre compte d'exemples tels que (3.30) sans avoir recours à l'inférence.

(3.30) The policeman who arrested Jon forgot to read him his rights and so did the one PETER got collared by.

Comme nous l'avons vu en section 2.1.2, dans l'approche unificationnelle, de tels cas demandent qu'une proposition soit inférée à partir du texte qui servira de source pour établir la contrainte de redondance. Dans l'approche basée sur la sémantique dynamique en revanche cette étape est superflue et l'interprétation lâche de *him* and *his* résulte du fait que le contexte cible contient un nouvel antécédent pour lier le pronom à savoir, *PETER*.

Par contre, un inconvénient de l'approche de Hardt est qu'elle place peu de restrictions sur les interprétations si bien que par exemple, elle prédit pour des exemples tels que (3.12) une interprétation lâche qui n'existe pas.

La seconde différence a trait à la directionnalité. La sémantique dynamique est fondamentalement directionnelle au sens où au sein d'une représentation sémantique, le terme représentant une anaphore doit être précédé par le terme représentant son antécédent. En conséquence, l'approche dynamique prédit correctement que (3.31) n'a pas d'interprétation lâche, c'est-à-dire n'a pas l'interprétation *this year, I voted for Harry* (exemple de [81]). Par contraste, l'approche unificationnelle prédit que la clause elliptique est ambiguë entre une interprétation stricte et une interprétation lâche.

(3.31) Tom^1 is always causing me problems. Last year, I didn't vote for him_1 and Tom got mad at me. This year, I did, and HARRY got mad at me.

L'avantage est mitigé cependant au regard de l'exemple suivant :

(3.32) Tom is always causing me problems. Last year, he got mad at me because I didn't vote for him and THIS year, HARRY got mad at me because I did.

Dans ce cas, *Harry* précéde l'ellipse si bien que l'approche dynamique prédit une interprétation lâche. Pourtant, tout comme en (3.31), cette interprétation n'est pas disponible. En bref, il n'est pas clair que l'exemple (3.31) soit une argument très décisif en faveur de l'approche dynamique. Comme le montre l'exemple (3.32) l'absence d'interprétation lâche est due non pas à l'ordre linéaire mais à un autre facteur, peut être la sémantique de la relation de contraste établie entre proposition cible et proposition source ou encore, le fait que *Tom* soit le topique du discours.

L'approche de Fiengo et May. Dans l'approche présentée dans [51] (FM), l'interprétation d'une ellipse entraîne un processus de reconstruction syntaxique et les différents types de GNs sont représentés différemment. Par exemple, un nom propre est toujours représenté par une occurrence de type α tandis qu'un pronom peut être représenté soit par une occurrence de type β soit par une occurrence de type α. Le processus de reconstruction prend en compte cette distinction de types si bien que des contraintes sur la reconstruction peuvent être formulées qui ont pour objet de capturer les interactions entre théorie du liage et reconstruction elliptique. En particulier, l'approche de Fiengo et May prédit la différence entre noms propres et pronoms discutée en section 3.5. Elle rend aussi compte de ce que l'exemple (3.33) n'a pas l'interprétation *Peter said Max saw Peter's mother.*

(3.33) Max^1 said he_1 saw his_1 mother and Peter^2 did too

L'approche unificationnelle présentée ici ne peut expliquer cette interprétation manquante pour la simple raison que les pronoms co-indexés ont des représentations identiques et sont donc traités de façon identique.

3.7 Conclusion

Nous avons présenté un traitement uniforme de l'interprétation lâche qui est suffisamment général pour couvrir les cas d'ambiguïtés lâche/stricte observés dans la littérature. Ce traitement intègre une contrainte de redondance qui restreint les interprétations lâches aux cas où l'antécédent de la forme dépendante source a un élément parallèle dans la proposition cible.

Deux grandes lignes de recherche restent à explorer.

D'une part, il reste à trouver comment déterminer les éléments parallèles. Une faiblesse de l'approche présentée ici est qu'elle prend ces éléments comme donnés. Pour résoudre ce problème, deux approches sont possibles. Soit les éléments parallèles sont déterminés par une contrainte résultant de la sémantique du parallélisme et du contraste (cf. e.g. [89, 5]), soit ils sont définis au moyen d'un mécanisme de comparaison sur les représentations sémantiques tels que l'union prioritaire (cf. e.g. [154, 79]). Comme le montre [62], il est nécessaire d'allier la précision et le pouvoir prédictif d'une approche basée sur un mécanisme de comparaison avec la puissance de raisonnement permise par l'accès aux connaissances linguistiques et encyclopédiques. Une telle approche est proposée par [87] – nous avons vu section 2.3.2 comment cette approche diffère de l'approche unificationnelle proposée ici.

D'autre part, il reste à montrer comment l'analyse et en particulier, l'utilisation de l'UCOS, peut être modifiée pour rendre compte de l'interaction entre ambiguïté lâche/stricte et anaphore pronominale. Comme nous l'avons vu dans la section précédente, l'analyse unificationnelle échoue en effet à prédire les «puzzles éliminatifs de l'ellipse». Elle ne capture pas le fait que dans certains textes contenant n éléments coréférentiels dans la source, il y a moins de 2^n interprétations qui sont en fait disponibles (cf. [36, 51]). Pour de tels cas et plus généralement, pour un traitement exhaustif de l'interaction entre processus de construction sémantique et les autres processus linguistiques, une approche plus sophistiquée est nécessaire.

Chapitre 4

Conclusion

Certaines constructions linguistiques ont une interprétation qui est régie par le parallélisme entre la phrase les contenant (phrase cible) et celle contenant l'élément permettant de reconstruire leur interprétation (phrase source).

Pour ces constructions, l'interprétation fait intervenir un processus de reconstruction qui peut se baser sur différents niveaux de l'analyse linguistique. Ainsi par exemple, [121] argumente pour une reconstruction *phonologique* de certains énoncés désaccentués, [51] pour une reconstruction de l'ellipse verbale basée sur la *syntaxe* et [168, 165, 39] pour une reconstruction *sémantique*. Au niveau sémantique, deux grands types d'approches peuvent à leur tour être identifiés : les approches ([168, 39]) basées sur la *syntaxe de la sémantique* c'est-à-dire sur les représentations sémantiques ; et celles basées sur une évaluation dans un modèle [165].

Comme les trois chapitres précédents l'ont montré, l'analyse unificationnelle, en se basant sur la syntaxe de la sémantique, fournit un niveau d'analyse approprié pour une analyse uniforme de l'effet du parallélisme sur l'interprétation : de l'ellipse verbale, des énoncés désaccentués et des anaphores lâches. Les approches basées sur d'autres niveaux d'analyse en revanche ne semblent pas pouvoir se généraliser de la même façon à l'ensemble de ces phénomènes. Ainsi l'analyse basée sur le niveau phonologique de Krifka ne permet ni une analyse uniforme des énoncés désaccentués, ni une généralisation à l'ellipse verbale puisqu'elle exige une stricte équivalence phonologique entre un énoncé désaccentué et sa phrase source[31]. L'analyse basée sur la syntaxe de Fiengo and May exige que la reconstruction du sens d'une ellipse se fasse sur la base de la forme logique associée avec la phrase source. Proche de l'approche unificationnelle de par sa comparaison de représentations sémantiques, cette approche en diffère du fait de sa base syntaxique : or comme nous l'avons vu au chapitre 2, la proposition source n'est pas nécessairement la représentation de la phrase source – elle peut en être inférée. Enfin comme nous l'avons vu en particulier en section 2.3, l'analyse purement sémantique de Rooth ne permet pas de rendre compte de l'interaction entre coréférence nominale et ellipse du fait en particulier de l'impossibilité de nommer les objets d'un modèle.

L'analyse unificationnelle n'en est pas pour autant exempte de difficultés et l'utilisation réelle de l'UOS comme outil de base pour l'interprétation des énoncés parallèles passe par la levée de plusieurs verrous.

Premièrement, nous avons vu que l'analyse unificationnelle repose sur l'identification préalable des éléments parallèles cibles et sources. Il est clair que cette identification est affectée par des facteurs syntaxiques (le sujet de l'ellipse est généralement un élément parallèle cible) et prosodiques (les éléments proéminents des énoncés désaccentués sont généralement des éléments

[31]Pour une comparaison détaillée des deux approches, voir [61].

parallèles cibles). Mais les contraintes exactes permettant de déterminer les éléments parallèles restent à déterminer. Pour une analyse exhaustive de cette notion, une analyse sur corpus est incontournable.

Deuxièmement, comme nous l'avons vu au chapitre 2, section 2.3.2, l'analyse unificationnelle ne rend parfaitement compte ni de l'interaction entre parallélisme, ellipse et anaphore nominale, ni de celle entre parallélisme et connaissances encyclopédiques. Nous avons en outre vu que pour remédier à ces problèmes, il est nécessaire de pouvoir prendre en compte pendant la reconstruction sémantique (i) l'information sur les liens coréférentiels et (ii) certaines connaissances encyclopédiques (afin de pouvoir déterminer les éléments en relation de contraste sémantique). Ici l'adaptation à des fins linguistiques d'une forme d'unification sensible aux types ontologiques (sortes) est pertinente [115], ainsi que l' exploration d'une forme plus sophistiquée d'unification colorée où les couleurs pourraient être des combinaisons booléennes.

Troisièmement, le fragment logique utilisé par l'analyse linguistique doit être identifié. Quel est l'ordre supérieur maximal des équations utilisées ? Peut-on toujours réduire ces équations à des équations de filtrage (plutôt que d'unification) ? Quels sont les types des variables libres utilisées ? Une réponse à ces questions permettrait de mieux cerner les propriétés théoriques et formelles des problèmes d'unification soulevés par les données linguistiques et ce faisant, d'optimiser les algorithmes existant.

Troisième partie

Inférence logique

L'inférence sémantique joue un rôle essentiel dans le traitement de la langue naturelle et plus particulièrement, dans de nombreuses sous-tâches de ce traitement. Comme l'illustre l'exemple (4.1), elle peut par exemple être mise en jeu dans la tâche de résolution des pronoms.

(4.1) Jean a lancé une pierre contre la fenêtre et elle s'est cassée.

C'est en effet sur une base purement épistémique qu'un locuteur francophone interprétera le pronom *elle* dans cet exemple comme référant à la fenêtre plutôt qu'à la pierre. Parce qu'il sait qu'une fenêtre est plus fragile qu'une pierre et que le contact soudain d'un objet fragile avec un objet moins fragile peut entraîner la destruction de l'objet le plus fragile, le récepteur de l'énoncé (4.1) peut en déduire que l'interprétation privilégiée du pronom *elle* est ici celle qui réfère à l'objet le plus fragile à savoir, la fenêtre.

Plus généralement, l'inférence sémantique peut jouer un rôle déterminant dans l'identification de la portée d'un opérateur (quantificateurs, modalités, négation, adverbes etc.) ; dans la résolution des ellipses et des présuppositions [14, 17] ou encore dans la détermination des implicatures et implications d'un énoncé. Dans une perspective de génération, [183] montre que l'inférence sémantique est également nécessaire pour la production de textes naturels et en particulier de textes non redondants.

De fait, l'inférence sémantique est un élément tellement central du traitement de la langue qu'une grande partie des travaux initiaux dans le domaine du traitement automatique des langues (TAL) s'est concentrée sur le développement de modèles *intelligents*, c'est-à-dire de modèles capables de reproduire les raisonnements sémantiques nécessaires à la compréhension et à la production de la langue naturelle. Ainsi, le système SHRDLU de Winograd ([197]) visait à simuler un robot «comprenant» la langue naturelle. Son fonctionnement était basé sur une interprétation par la preuve, la preuve construite pour interpréter la requête de l'utilisateur étant ensuite utilisée pour construire une réponse appropriée. Un autre exemple célèbre est celui de la thèse de Charniak [24] sur la compréhension d'histoires pour enfants, qui repose sur l'hypothèse que les histoires «sont immédiatement traduites dans une représentation interne appropriée pour la déduction» (Charniak, 1973 :337). Ce système utilisait inférence et connaissances encyclopédiques afin de pouvoir déduire des faits non explicités par l'histoire et répondre à des questions sur ce savoir implicite.

Dans le domaine du dialogue, les recherches des années 80 ([129]) ont montré l'utilité de tirer des inférences basées sur le profil de l'utilisateur (ses croyances, ses objectifs) pour comprendre des questions (souvent des actes de paroles indirects et fragmentaires) et y répondre de façon coopérative par exemple en signalant à l'utilisateur qu'une de ses présuppositions n'est pas satisfaite.

Enfin dans le domaine de la génération, la thèse de ([4])fournit une étude détaillée du rôle de l'inférence dans la production d'énoncés satisfaisant les besoins de l'utilisateur tout en tenant compte de ses capacités.

Malheureusement, ces recherches finirent par aboutir à un résultat largement négatif : d'une part, la puissance de calcul des prouveurs et des ordinateurs de l'époque ne permettait pas un traitement efficace des requêtes d'inférence soulevées par les données linguistiques ; et d'autre part, l'identification, la représentation et la structuration des connaissances nécessaires au raisonnement sémantique n'avaient pas une base théorique suffisament riche pour être traitées à un niveau de généralité satisfaisant. Pour ces raisons, la recherche en linguistique informatique s'est concentrée dans les années suivantes sur des problèmes plus accessibles tels que la production de grammaires informatiques, d'analyseurs morphologiques et syntaxiques ou encore le développement de générateurs de texte.

Néanmoins les recherches sur l'inférence dans le TAL continuent et certains travaux ont récemment mis en évidence l'intérêt qu'il peut y avoir à reconsidérer cette problématique à la lumière des outils contemporains. Ainsi, dans le domaine de la génération, [183, 185, 66] montre comment intégrer des raisonneurs dans des systèmes de micro-planning pour permettre la production de textes non redondants tandis que dans le domaine de l'analyse, des raisonneurs pour la logique du premier ordre commencent à être utilisés pour des problèmes tels que la résolution des anaphores et la projection des présuppositions ([13, 17]).

De façon plus générale, deux facteurs donnent à penser qu'une modélisation informatique des raisonnements sous-tendant certaines des tâches du traitement des langues est au moins partiellement possible : la disponibilité de larges ressources lexicales telles que WordNet, FrameNet ou VerbNet, qui fournit un répertoire étendu des connaissances lexicales impliquées dans ce traitement ; et celle de raisonneurs automatiques de plus en plus performants (un prouveur pour les logiques de description tel que Racer par exemple, donne des réponses quasi immédiates à des requêtes d'inférence évaluées sur une ontologie contenant plusieurs milliers de concepts).

Dans les chapitres qui suivent, nous présentons des débuts d'investigation dans cette direction. Le chapitre 5 introduit les notions de base nécessaires par la suite. Les deux chapitres suivants se concentrent ensuite sur deux applications spécifiques à la langue naturelle : le problème de la désambiguïsation sémantique (chapitre 6) et la question de l'utilisation pour des tâches linguistiques du raisonnement taxonomique supporté par les logiques des descriptions (chapitre 7). Plus précisément, nous examinons dans ce chapitre l'utilisation qui peut être faite de ce type de raisonnement pour la tâche de lexicalisation en génération d'une part, et pour l'amélioration des systèmes de question/réponse d'autre part.

Chapitre 5

Prouveurs automatiques et générateurs de modèles

Ce chapitre présente les notions de base nécessaires à la compréhension des deux chapitres qui suivent. Nous commençons par aborder la question du type des logiques nécessaires ou utiles pour le traitement de la langue naturelle et argumentons pour une approche qui, dans un premier temps au moins, se base sur des logiques peu expressives par rapport aux besoins de la langue naturelle mais utilisables étant donné les ressources actuelles. Nous présentons ensuite les deux outils d'inférence qui seront utilisés dans le reste du chapitre : les générateurs de modèles d'une part, et les prouveurs de théorèmes d'autre part.

5.1 Quelle logique ?

La modélisation du raisonnement sémantique passe par l'utilisation d'une logique. Mais il existe bien sûr de nombreuses logiques.

Une logique se distingue d'une part, par sa syntaxe. Par exemple, une logique d'ordre supérieur permet une quantification généralisée tandis qu'une logique du premier ordre restreint la quantification aux individus du domaine de base. Une logique temporelle permettra de référer à des mondes futurs ou passés et plus généralement, une logique modale permettra de référer à des mondes possibles, probables, nécessaires, etc.

Pour une syntaxe donnée, la logique utilisée peut être, selon les règles structurelles adoptées, linéaire, intuitioniste ou classique (cf. chapitre 1).

La question se pose donc du choix de la logique appropriée au traitement de la langue naturelle. Nous ne reviendrons pas ici sur la question du choix de la logique – celle ci a déjà été abordée dans le chapitre 1. Nous considérons en revanche la question de l'expressivité et indirectement du choix de la syntaxe adoptée. Dans son travail sur le développement d'une sémantique compositionnelle pour la langue naturelle, [139, 140] argumente pour l'utilisation du Lambda-calcul typé intensionnel d'ordre supérieur. Dans le même ordre d'idée, [9] préconise l'utilisation d'une logique intensionnelle et [27] celle de la théorie des propriétés.

En d'autres termes, le traitement des inférences nécessaires au traitement de la langue naturelle requiert l'utilisation d'une logique extrêmement expressive. Malheureusement, plus une logique est expressive, moins ses propriétés mathématiques et informatiques sont bonnes : si la logique des propositions est NP-complète, la logique des prédicats n'est pas récursivement énumérable tandis que la théorie des propriétés et la logique intensionnelle sont incomplètes.

Face à ces observations, deux attitudes sont possibles.

On peut accepter le coût inhérent à la tâche et utiliser des logiques expressives qui permettent de coller au mieux à la sémantique de la langue naturelle. C'est l'attitude adoptée dans la théorie des situations par exemple ainsi que par Allan Ramsay [159] dans son approche computationnelle basée sur la génération de modèles.

Une deuxième possibilité consiste à préférer des logiques moins expressives et à se contenter d'une approximation des inférences sous-tendant le traitement de la langue naturelle. C'est l'approche poursuivie par exemple par [14] où la logique des prédicats est utilisée pour modéliser la résolution des présuppositions.

L'approche poursuivie ici est du deuxième type, ce choix étant motivé par les objectifs de recherche poursuivis. En effet, notre but n'est pas ici de modéliser les raisonnements sous tendant le traitement de la langue naturelle de façon exacte et exhaustive – si c'est là un but louable, il nous semble pour l'instant trop ambitieux – mais plutôt de développer des outils d'inférence permettant le traitement automatique en temps réel de certaines des sous-tâches d'inférence soulevées par les données et les applications linguistiques. Nous visons ainsi à identifier un certain nombre de tâches d'inférence sous-tendant le traitement de la langue ; à décrire chacune de ces tâches de façon systématique (à partir d'études de corpus et d'ensembles de tests) et à tester et optimiser les prouveurs existants par rapport à ces tâches. En pratique donc, nous utilisons deux types de logiques :

- la logique des prédicats et
- la logique des descriptions.

Le choix de l'une ou l'autre de ces deux logiques dépendra de la tâche considérée. Ainsi, nous utilisons la logique des prédicats pour la désambiguisation en particulier, celle du réciproque *les uns les autres* parce que le traitement formel adopté exige une quantification universelle non restreinte sur les individus (ce qui n'est pas le cas pour les logiques de descriptions qui ne permettent pas d'exprimer un message tel que *Jean aime toutes les femmes*). En revanche, nous utilisons les logiques des descriptions pour les autres applications considérées, soit parce qu'elles leur sont particulièrement appropriées du fait de la nature même du problème (c'est le cas par exemple pour la tâche de lexicalisation en génération) soit parce qu'elle permet d' approximer une tâche qui doit être exécutée de façon aussi performante que possible – c'est le cas par exemple pour les systèmes de question/réponse où les masses de données à traiter sont énormes et où l'efficacité devient du coup, un facteur dominant.

Enfin une logique donnée peut être spécifiée par différents cadres formels de règles d'inférence : système de déduction naturelle, système d'axiomes à la Hilbert, calcul des séquents, réseaux de démonstrations, etc. Le choix de ce cadre est dicté par l'application visée. Par exemple, dans les grammaires dites «catégorielles» où une analyse syntaxique est une démonstration, il convient d'avoir des structures de preuves reflétant la structure des constituants. Il est également souhaitable qu'il n'y ait pas plusieurs analyses formelles pour un constituant donné. Pour cette application, la déduction naturelle et les réseaux de démonstrations seront donc plus adaptés que les déductions à la Hilbert qui n'ont pas une claire structure d'arbre ou que le calcul des séquents qui distinguent des démonstrations équivalentes.

Pour les applications sémantiques qui nous concernent, le cadre d'inférence doit permettre soit de vérifier la cohérence d'une formule soit de générer un (ou plusieurs) modèle(s) la satisfaisant. En outre, ces processus de vérification et de génération doivent pouvoir être implémentés et évalués. Pour la vérification, nous utilisons des systèmes existants et disponibles sur le web qui sont basés soit sur les tableaux soit sur la résolution. Pour la génération, les systèmes basés sur les tableaux sont les plus appropriés puisqu'ils permettent de construire un modéle par la simple lecture d'une branche non fermée.

5.2 Les générateurs de modèles

Un générateur de modèles permet de déterminer la satisfiabilité d'une théorie logique, c'est-à-dire d'un ensemble de formules logiques. Comme le nom l'indique, un générateur construit des modèles : si le générateur peut construire un modèle satisfaisant la théorie considérée, cette théorie est par définition satisfiable.

Modèles et modèles de Herbrand

Comme nous l'avons vu au chapitre 1, un **modèle** est une structure mathématique qui fixe l'interprétation des symboles d'une théorie logique. Pour un langage du premier ordre L, un modèle est une paire $\langle I, D \rangle$ telle que D est un ensemble non vide d'entités (**l'univers** ou **domaine** des individus) et I est une fonction d'interprétation qui assigne aux constantes de L des éléments de D et aux symboles relationnels de L des relations sur D d'arités appropriées.

Une classe importante de modèles concerne les **modèles de Herbrand** dans lesquels les constantes et autres termes sont interprétés comme eux-même (pour tous les termes fermés : $I(t) = t$). Dans ces modèles, l'univers est un ensemble de termes construit à partir de la signature de la spécification interprétée. Dans une logique classique du premier ordre, un modèle de Herbrand M et son univers D sont totalement spécifiés par l'ensemble *Ground(M)* de tous les littéraux (termes ou négations de termes) clos (c'est-à-dire sans variables) satisfaits par M. Dans ce qui suit, un modèle de Herbrand M est assimilé à sa représentation comme ensemble de littéraux clos. En outre, comme cette représentation est parfois fastidieuse, les modèles de Herbrand ne seront en fait représentés que par leur partie positive, c'est-à-dire par l'ensemble des littéraux positifs $\textsc{Pos}(M)$. La fonction d'interprétation I qui est dérivée de M interprète tout littéral P comme faux si P n'est pas dans M.

Les modèles de Herbrand sont importants parce que toute théorie satisfiable du premier ordre a au moins un modèle de Herbrand. Les raisonneurs automatiques utilisent cette propriété pour prouver l'insatisfiabilité. Pour prouver qu'une théorie est insatisfiable, il suffit de prouver qu'elle n'a pas de modèle Herbrand. C'est en général plus simple que de démontrer l'inexistence de modèles arbitraires.

Décidabilité

La logique du premier ordre est indécidable, c'est-à-dire qu'il est impossible de définir un algorithme qui puisse déterminer si oui ou non une formule de cette logique est valide. De même, il est impossible de définir un algorithme qui puisse construire pour *toute* théorie logique satisfiable un modèle satisfaisant cette théorie.

Néanmoins, il est possible de définir une méthode complète pour la satisfiabilité finie : lorsqu'une spécification (c'est-à-dire un ensemble fini de formules de la logique du premier ordre) est soit insatisfiable soit satisfiable par un modèle fini, cette méthode retourne le résultat approprié (pas de modèle ou tous les modèles satisfaisant la spécification suivant le cas).

Dans ce qui suit, seuls les modèles finis seront considérés même s'il est clair que certaines phrases telles par exemple *Chaque personne a un parent qui est une personne* ont des modèles infinis. En pratique, un générateur de modèles pourra être amené à fonctionner sur de tels exemples en fixant une limite supérieure pour la taille de l'univers.

Minimalité

Les générateurs de modèle sont des outils utiles pour implémenter l'idée d'un principe de minimalité parce qu'ils permettent une définition précise et opérationnelle du concept d'interprétation minimale. En effet, une formule logique du premier ordre est satisfiable non pas par un, mais par plusieurs modèles. Par exemple, la formule $\exists x.p(x)$ est satisfaite par le modèle $\{p(a)\}$ mais aussi par les modèles $\{p(a), p(b)\}, \{p(a), p(b), q(a)\}$ etc.

Dans des applications spécifiques, les générateurs de modèles implémentent différentes formes de minimalité qu'ils utilisent pour restreindre l'ensemble des modèles générés à l'ensemble des modèles satisfaisant le critère de minimalité choisi. On distingue en particulier : la minimalité ensembliste, la minimalité de domaine, la minimalité locale et la minimalité spécifique à un prédicat.

Minimalité ensembliste. La minimalité ensembliste [21] sélectionne parmi les modèles satisfaisant une spécification donnée, les modèles ayant le plus petit nombre de littéraux. Plus précisément, étant donné un ensemble de formules du premier ordre Φ et S l'ensemble des modèles de Herbrand satisfaisant Φ, un modèle $\langle$ *I,D* $\rangle$ dans S est *ensemble-minimal* ssi il n'existe pas d'autre modèle $\langle$ *I',D'* $\rangle$ dans S tel que $I' \subset I$.

La minimalité ensembliste est au coeur du raisonnement sur les informations incomplètes basé sur la circonscription [133]. Dans le raisonnement par circonscription, la relation de conséquence $\models_c$ est formalisée comme suit : $\Gamma \models_c \phi$ ssi ϕ est vraie dans tous les modèles ensemble-minimaux de Γ.

Minimalité de domaine. La minimalité de domaine minimise les individus plutôt que les faits. Etant donné un ensemble de formules du premier ordre Φ et S l'ensemble des modèles de Herbrand satisfaisant Φ, un modèle $\langle$*I,D*$\rangle$ dans S est domaine minimal ssi il n'existe pas d'autre modèle $\langle$*I',D'*$\rangle$ dans S tel que la cardinalité de *D'* est plus petite que la cardinalité de D.

La minimalité de domaine a été introduite par [85, 130] comme une forme alternative de circonscription visant à minimiser les individus plutôt que les faits.

Minimalité locale. La minimalité locale minimise à la fois les individus et les faits. Etant donné un ensemble de formules du premier ordre Φ et S l'ensemble des modèles de Herbrand domaine-minimaux satisfaisant Φ, un modèle $\langle$*I,D*$\rangle$ dans S est local-minimal ssi il n'existe pas d'autre modèle $\langle$*I',D'*$\rangle$ dans S tel que $I' \subset I$.

Minimalité spécifique à un prédicat. Etant donné p un symbole de prédicat et $p(M)$ le sous-ensemble des atomes de $\textsc{Pos}(M)$ dont le symbole prédicat tête est p, un modèle M pour la spécification logique Φ est p-minimal s'il n'existe pas de modèle M' satisfaisant Φ, tel que $\mid p(M') \mid < \mid p(M) \mid$.

Ce type de minimalité est utilisé dans les applications de diagnostic dont le but est de produire le ou les modèles minimisant la cardinalité d'un prédicat enregistrant les anomalies du système modélisé par la spécification logique donnée en entrée.

Énumération

Les modèles minimaux d'une spécification logique représentent généralement différentes solutions ou interprétations possibles. Il est donc important de pouvoir énumérer l'ensemble des modèles minimaux satisfaisant une spécification, c'est-à-dire d'avoir une méthode de génération complète. Comme nous l'avons vu plus haut, nous nous restreignons ici aux systèmes pour la satisfiabilité finie. Parmi les générateurs de modèles complets pour la satisfiabilité finie, on trouve en particulier :

– Le système FinFiMo, MM-Satchmo de [21], un prouveur de théorèmes générateur de modèles sans skolémisation (la skolémisation est problématique car elle peut transformer une spécification satisfiable par un modèle fini en une spécification qui n'a pour modèle satisfaisant que des modèles infinis).
  `http://www.pms.informatik.uni-muenchen.de/software/finfimo`
– Le générateur de modèles Finder [175] qui applique des procédures de décision propositionnelles à des spécifications de premier ordre traduites dans la logique des propositions à partir d'un univers donné.
  `http://arp.anu.edu.au/\~{}jks/jks.html`
– SEM – J. Zhang and H. Zhang
  `http://www.cs.uiowa.edu/\~{}hzhang/sem.html`
– Paradox - K. Claessen, N. Soerensson `http://www.cs.chalmers.se/~koen/paradox/`
– Le générateur de modèles KimBa [118] basé sur les contraintes et adapté aux exigences de la langue naturelle. Ce système accepte en particulier des spécifications d'ordre supérieur permettant ainsi une représentation du sens de la langue naturelle compatible avec celle proposée dans la littérature sur la sémantique formelle de la langue naturelle.

Les propositions présentées dans le chapitre 6 ont été implémentées à l'aide du générateur de modèles KimBa.

5.3 Les prouveurs de théorèmes

Un prouveur de théorèmes vise à déterminer la validité d'une formule dans une logique donnée.

Logique des prédicats. Si l'ensemble des validités de la logique des prédicats est récursivement énumérable, l'ensemble de ses formules non valides ne l'est pas, ce qui rend la logique des prédicats non récursive (ou semi-décidable). En d'autres termes, il est impossible de définir un algorithme qui puisse décider si oui ou non, une formule donnée est une formule valide de la logique des prédicats. En revanche, il est possible de définir un algorithme qui, si une formule est valide, sera capable de déterminer la validité de cette formule et de terminer. Enfin, certains fragments de la logique des prédicats sont récursifs tels par exemple, le fragment gardé qui est dans la classe de complexité ExpTime.

Divers prouveurs sont actuellement disponibles sur le web tels en particulier, Bliksem [42], Otter [137] et Spass [195]. Comme la plupart des prouveurs existants, ces prouveurs ont été optimisés sur des problèmes mathématiques. Néanmoins, ils ont fait l'objet d'une première application au domaine linguistique [14, 17] dont les résultats sont encourageants. En particulier, [17] utilise les prouveurs Bliksem, otter, mace et Spass en combinaison avec le générateur de modèles Mace [136] pour vérifier la consistance de représentations sémantiques «résolues» (c'est-à-dire de représentations sémantiques où les antécédents des pronoms et des descriptions définies sont identifiés). Il conclut que pour cette tâche spécifique, Spass et Otter performent au mieux permettant de trouver une réponse à 66% des 283 problèmes donnés, ceci en l'espace de 30 secondes avec des temps CPU allant de 2.5 à 29.9 secondes (13 secondes en moyenne).

Logique des descriptions. [19] montre qu'il y a un compromis à faire entre l'expressivité d'un langage de représentation et la facilité avec laquelle il est possible de raisonner sur les expressions de ce langage. En clair, plus le langage est expressif, plus le raisonnement est difficile. Cette observation a donné lieu au développement de logiques moins expressives mais plus «efficaces» et en particulier, à la famille des logiques de descriptions. Ces logiques sont pour la plupart décidables[32], les moins expressives étant même dans la classe de complexité PTime.

Si les logiques de description (LDs) sont encore moins aptes à représenter le sens des expressions de la langue naturelle de façon précise que la logique des prédicats, elles permettent un raisonnement taxonomique particulièrement efficace. Pour cette raison, il n'est pas inintéressant d'approximer le sens de la langue naturelle dans le cadre des logiques des descriptions lorsque l'application (linguistique ou commerciale) considérée requiert un raisonnement taxonomique important – c'est le cas par exemple pour la tâche de lexicalisation en génération [59]) et, comme nous le verrons plus en détail dans le chapitre 7, pour le raisonnement lexical impliqué dans le traitement des paires Question/Réponse.

Comme pour la logique des prédicats, des prouveurs pour les logiques des descriptions sont disponibles sur le web dont en particulier, FaCT [91, 92], Dlp [147] et Racer [80].

Fact a deux versions, l'une supportant la logique $\mathcal{ALCH}_{R+}$, l'autre la logique $\mathcal{SHIQ}$. Bien que ces logiques soient très expressives pour des LDs, le prouveur est correct et complet. Il est également optimisé sur les cas moyens et démontre qu'en pratique, expressivité ne rime pas nécessairement avec inefficacité. Dlp est une version optimisée de Fact appliquée à la logique $\mathcal{ALCN}$. Enfin Racer est le seul de ces trois raisonneurs qui permette outre le raisonnement terminologique, le raisonnement assertionnel. Racer supporte la logique $\mathcal{SHIQ}$ et intègre toutes les techniques d'optimisation utilisées par Fact.

Une utilisation linguistique de Racer est décrite dans [116] qui présente un système utilisant l'inférence pour différents modules d'un jeu informatique comme par exemple, pour résoudre des expressions anaphoriques, pour confronter des opérateurs de planification avec l'état du monde ou encore pour générer des descriptions définies. Là encore les premiers résultats sont encourageants puisque la plupart des requêtes obtiennent une réponse en moins de 10 millisecondes avec seulement quelques requêtes prenant plus de 100ms. Les requêtes les plus coûteuses prennent environ 500ms et le temps moyen utilisé par les requêtes d'inférence pour chaque entrée de l'utilisateur est de 546ms avec une moyenne de 39 requêtes d'inférence par entrée[33].

[32] Certaines logiques de descriptions particulièrement expressives sont indécidables telle que par exemple, la logique $\mathcal{FL}^-$ étendue à la composition de rôles et l'égalité entre rôles.

[33] Malheureusement [116] ne donne aucune indication sur la taille de la base de connaissance utilisée par leur jeu.

Chapitre 6

Désambiguïsation

Ce chapitre explore l'utilisation de raisonneurs automatiques (prouveurs de théorèmes et générateurs de modèles) pour la résolution d'ambiguïtés discursives. Nous montrons d'abord que l'utilisation conjointe d'un prouveur de théorèmes et d'un générateur de modèles peut permettre de contribuer à la la désambiguïsation d'un certain nombre d'anaphores nominales et relationnelles (noms composés, métonymies et descriptions définies). Nous montrons ensuite que la génération de modèles permet un traitement informatique naturel de l'interprétation des phrases faisant intervenir le réciproque *each other* (*l'un l'autre*).

6.1 Anaphores nominales et relationnelles

L'inférence sémantique peut être nécessaire pour résoudre les ambiguïtés intervenant dans la résolution des noms composés, des métonymies logiques et des descriptions définies. Dans ce qui suit, nous commençons par présenter l'approche décrite dans [14] et en particulier, les critères de cohérence et d'informativité qu'elle utilise. Nous argumentons ensuite en faveur d'un critère additionnel de désambiguïsation à savoir le critère de minimalité. Enfin, nous confrontons le modèle résultant aux données linguistiques.

6.1.1 Un premier traitement inférentiel de la désambiguisation

[190] présente une théorie des présuppositions qui met en jeu des tests d'ordre pragmatique : pour être acceptable, une présupposition doit être à la fois *informative* et *cohérente* par rapport au contexte considéré. Le travail présenté dans [14] montre comment ces deux critères peuvent être implantés à l'aide de raisonneurs automatiques.

Cohérence. Comme nous l'avons vu dans le chapitre précédent, la cohérence logique et la validité d'une formule de la logique de premier ordre sont indécidables : étant donné une formule ϕ, il n'existe pas de méthode qui permette de déterminer si ϕ est ou non cohérent. Pour remédier à ce problème, [14] proposent d'utiliser conjointement un prouveur de théorème et un générateur de modèles. Etant donné ϕ la représentation logique du texte à désambiguïser et Γ celle de son contexte et du savoir pertinent[34], le test de cohérence cherchera

1. à prouver que $\neg(\Gamma \wedge \phi)$ est valide et

[34] Dans l'approche de Blackburn et Bos, posé et présupposé ne sont pas distingués formellement. Comme dans la théorie de [190], un présupposé est un élément anaphorique qui peut être soit lié par le contexte soit accommodé. Dans les deux cas, la représentation résultante ne diffère pas formellement de la représentation du posé.

2. à construire un modèle pour $\Gamma \wedge \phi$

L'utilisation conjointe d'un prouveur et d'un générateur permet de maximiser les chances de terminaison. De fait, [14] montrent qu'en pratique, l'utilisation d'un prouveur de théorèmes sophistiqué (e.g. BLIKSEM [42], SPASS [195]) suffit à obtenir une réponse *rapidement* et dans *pratiquement tous les cas considérés*, c'est-à-dire pour les problèmes d'inférence soulevés par le traitement des présuppositions.

Informativité. Comme [14] le montre, le test d'informativité peut être réduit à un problème de cohérence logique : étant donné un contexte linguistique et extra-linguistique Γ, la proposition ϕ est informative par rapport à Γ, si et seulement si, Γ n'implique pas ϕ ou de façon équivalente, si $(\Gamma \rightarrow \phi)$ n'est pas valide.

6.1.2 Un ajout : le critère de minimalité

Dans ce qui suit, nous montrons que l'approche de [14] peut être étendue au problème de l'interprétation des anaphores nominales et relationnelles. Pour rendre compte des données observées, nous introduisons cependant un troisième critère, celui de la minimalité, critère qui vise à privilégier les interprétations faisant intervenir un nombre minimal d'individus et/ou de faits.

Dans le domaine de la linguistique informatique, la minimalité a été identifiée de façon récurrente comme étant l'un des facteurs permettant la désambiguïsation sémantique. Ainsi [34] et [100] utilisent la minimalité pour choisir entre les interprétations concurrentes d'un énoncé tandis que [88] utilise la factorisation pour rendre compte du fait que les propositions impliquées par une énoncé peuvent être redondantes les unes par rapport aux autres. Cette factorisation unifie les hypothèses identiques d'un énoncé permettant ainsi de restreindre le nombre d'hypothèses à partir desquelles la signification globale d'un énoncé peut être calculée. En pratique, la factorisation permet de réduire le nombre d'hypothèses nécessaires pour vérifier la vérité d'un énoncé.

Intuitivement, la minimalité rappelle l'heuristique de Sacerdotti [167] : «Use existing objects !» (*Utilisez les objets existants !*) mais appliquée au contexte linguistique. Dans ce contexte, le terme «objet» réfère aux objets sémantiques évoqués par le discours, c'est-à-dire aux individus, aux éventualités et aux propositions tandis que le terme «existant» désigne le fait d'avoir été évoqué soit directement, soit indirectement par le discours et/ou la situation discursive. De fait, la minimalité sert la cohésion du discours en privilégiant les interprétations qui continuent à parler des objets précédemment introduits au détriment de celles qui introduisent de nouveaux objets.

Dans ce qui suit, nous verrons que le test de minimalité peut être implanté à l'aide d'un générateur de modèles : étant donné deux interprétations possibles pour un énoncé, l'interprétation privilégiée sera celle dont le modéle est minimal. Le choix d'une forme de minimalité donnée parmi les formes de minimalités connues (cf. chapitre 6) dépendra de la construction linguistique considérée.

Il faut cependant noter que le critère de minimalité n'est ni suffisant, ni parfait.

Qu'il ne soit pas suffisant est évident : le critère de minimalité n'est que l'un des nombreux critères intervenant dans la désambiguïsation d'un énoncé. Il est clair en particulier qu'une interprétation incohérente ou redondante (non informative), même si elle est minimale, peut difficilement être l'interprétation visée. En outre, des contraintes additionnelles provenant d'autres niveaux linguistiques (syntaxe, prosodie etc.) peuvent elles aussi interagir avec le critère de minimalité pour déterminer l'interprétation d'un énoncé. Par exemple, supposons que l'énoncé (6.1a) ait pour représentation sémantique (6.1b) :

(6.1) a. John looks at his friend.
b. $\exists x(lk(j,x) \wedge frd(x) \wedge of(x,j))$
c. $M_1 = \{lk(j,j), frd(j), of(j,j)\}$
d. $M_2 = \{lk(j,a), frd(a), of(a,j)\}$

M_1 et M_2 sont tout deux des modèles satisfaisant (6.1b). Mais le modèle localement minimal[35] reflète une interprétation où Jean se regarde, ce qui n'est pas l'interprétation préférée de (6.1a). Ici une contrainte pragmatique entre en jeu qui est la contrainte de manière de Grice exigeant du locuteur qu'il soit clair (Be perspicuous !) [77]. Plus précisément, si le locuteur avait voulu que les deux syntagmes nominaux coréférent, il aurait dû, pour respecter cette contrainte, utiliser une construction qui implique cette identification comme par exemple, *Jean se regarde.*

Dans tous les cas où le savoir prosodique, syntaxique, pragmatique, encyclopédique etc. nécessaire pour résoudre les ambiguïtés contenues dans un énoncé peut être identifié et inclus dans la spécification donnée en entrée au générateur de modèles, tout modèle incohérent avec cette spécification sera automatiquement exclu. Par exemple pour le texte (6.1a) ci-dessus, le modèle M_1 sera exclu des interprétations possibles dès lors que la contrainte $x \neq j$ est incluse dans la spécification donnée en entrée.

Malheureusement, il est des cas où l'interprétation visée viole une contrainte linguistique. Ainsi par exemple, on pourra énoncer le discours (6.2) pour signifier que Jean s'admire.

(6.2) Jean admire sa personne préférée.

En bref, la minimalité, même lorsqu'elle est combinée de façon appropriée avec les autres facteurs de Désambiguïsation, ne saurait fournir un test parfait pour déterminer l'interprétation privilégiée dans un contexte donné d'un énoncé donné. Il reste que s'il est possible d'identifier les situations où la minimalité est un bon critère de désambiguïsation (c'est-à-dire où son taux de faux positifs et de faux négatifs est bas), il fournit un test utile pour précisément ces situations.

Le critère de minimalité est également imparfait en ce qu'il est parfois impossible de déterminer une interprétation minimale unique. Rappelons que le critère de minimalité est appliqué à un ensemble de modèles minimaux satisfaisant un ensemble de représentations sémantiques. Or dans certains cas, cet ensemble peut contenir des modèles qui ne sont pas comparables (typiquement parce qu'ils ne sont pas dans une relation de sous ensemble). Lorsque ces modèles concurrents explicitent des interprétations également plausibles de l'énoncé considéré, il est normal de ne pas pouvoir les ordonner. Ce n'est pas le cas en revanche, lorsque des modèles incomparables ou de taille égale reflètent des interprétations ayant des degrés d'acceptabilité différents. Dans une telle situation, il faudrait sans doute inclure des facteurs de choix additionnels tels que par exemple, la préférence pour une certaine source de savoir (connaissance encyclopédique *vs* contexte linguistique), pour un certain type de résolution (lien anaphorique *vs* accommodation) ou avoir recours à des notions telles que la saillance et la probabilité. Soulignons à ce propos, que le but n'est pas ici de résoudre le problème de la désambiguïsation du discours dans son intégralité mais plutôt de montrer que le critère de minimalité peut être utilisé pour éliminer rapidement certaines des interprétations *a priori* possibles pour un énoncé donné.

6.1.3 Approche constructive *vs.* approche filtrante

Il y a deux grandes façons d'intégrer inférence et analyse : soit l'inférence est utilisée pour *résoudre* des interprétations sous-spécifiées produites par l'analyse, soit elle est utilisée pour *filtrer* les interprétations indésirables.

[35]Rappelons (cf. chapitre 6, section 5.2) que la minimalité locale minimise à la fois les individus et les faits.

Dans le premier type d'approche – l'approche «constructive» – une formule logique contenant des variables voit la valeur de ces variables déterminée par le processus d'interprétation qui est aussi un processus d'inférence (comme par exemple, l'abduction). Ainsi une représentation plus spécifique est *construite* à partir d'une représentation sous-spécifiée. Cette approche est l'approche adoptée par [88, 63] et à un degré moindre par [126]. Ainsi, [88] utilise l'abduction pour prouver une représentation sémantique sous-spécifiée. Lorsque le processus de preuve exige de faire des hypothèses sur la valeur des variables impliquées, ces hypothèses deviennent partie intégrante de l'interprétation permettant ainsi de mieux spécifier le sens de l'énoncé. De façon similaire, [126] utilise l'inférence par défaut pour calculer l'interprétation d'un discours, le contenu défaisable inféré permettant lui aussi d'enrichir une représentation sémantique initiale généralement sous-spécifiée.

Comme il a souvent été remarqué, l'utilisation de l'abduction pour l'interprétation de la langue naturelle présente beaucoup de points communs avec la génération de modèles telle qu'elle est mise en oeuvre dans [63, 10, 159]. Lorsqu'un générateur de modèles est utilisé pour l'interprétation de la langue naturelle, l'interprétation d'une formule ϕ représentant le sens d'un texte est un des modèles satisfaisant ϕ. Comme pour l'abduction, ce modèle peut expliciter des informations qui ne sont qu'implicites dans l'énoncé considéré [63].

L'avantage de cette première approche est qu'elle permet une intégration directe du processus de raisonnement sémantique dans le processus d'interprétation. Dans le cas de la génération de modèles, on a en particulier un test de cohérence «pour rien» puisque seuls les modèles cohérents seront construits. Une fois que l'ensemble des interprétations possibles est produit soit par abduction, soit par génération de modèles, il est possible de définir une métrique permettant d'ordonner ces interprétations concurrentes par ordre de préférence. On a ainsi une prise directe sur l'ordonnancement des interprétations concurrentes.

Le désavantage est que l'ambiguïté est résolue sur une base purement sémantique. Ceci va à l'encontre par exemple du traitement de van der Sandt pour la résolution des présuppositions, qui fait un usage essentiel de la structure des représentations sémantiques [190]. Il est bien sûr possible de prendre en compte cette structure et tout autre paramètre jugé pertinent mais cela nécessite un encodage explicite en terme de prédications additionnelles.

La seconde façon d'intégrer analyse et inférence consiste à utiliser l'inférence en complément d'algorithmes spécialisés pour la résolution des ambiguïtés. L'inférence sert alors à *filtrer* les interprétations indésirables. C'est en particulier l'approche adoptée dans [13] où des prouveurs sont utilisés pour éliminer les interprétations incohérentes ou non informatives produites par l'algorithme de projection des présuppositions.

L'inconvénient de cette seconde approche est qu'elle exige que toutes les ambiguïtés soient identifiées puis résolues. Les interdépendances entre les différentes expressions et différents types d'ambiguïté peuvent rendre ce processus extrêmement complexe. Si les interprétations produites sont filtrées séquentiellement par exemple (c'est-à-dire sans tenir compte de caractéristiques communes qui pourrait permettre de filtrer des sous ensembles entiers de possibilités), le processus d'interprétation sera particulièrement inefficace. Mais l'entrée de l'opération de filtrage n'est pas nécessairement l'ensemble des interprétations possibles. Par exemple, on pourrait imaginer ordonner les interprétations par ordre de minimalité (de la même façon que dans une recherche en largeur, les solutions les plus courtes sont produites en premier ou dans une recherche A*, les solutions «les meilleures») puis appliquer les tests de cohérence et d'informativité aux sous ensembles ordonnés par minimalité. Les interprétations privilégiées seraient alors les interprétations minimales satisfaisant la cohérence et l'informativité. Des tests additionnels (dont nous ne discuterons pas ici) peuvent ensuite être utilisés pour choisir entre les interprétations ayant soit la même minimalité, soit des minimalités incomparables.

Parce que l'approche de filtrage permet une spécification précise des facteurs pertinents et de leur mise en oeuvre, c'est l'approche que nous adopterons ici.

6.1.4 Données et analyse

Nous montrons maintenant comment les critères de cohérence, d'informativité et de minimalité peuvent contribuer à désambiguïser certaines anaphores relationnelles et nominales. Pour les anaphores nominales, nous nous limitons aux descriptions définies. Pour les anaphores relationnelles,[36] l'approche peut être appliquée aux groupes nominaux composés (6.3), aux adjectifs non-intersectifs (6.4) et aux métonymies logiques (6.5) mais nous nous limiterons ici à l'étude des noms composés et renvoyons le lecteur intéressé à [70] pour une discussion de la métonymie.

(6.3) The California student produced an excellent report.
Interprétation : The student **from** California produced an excellent report.

(6.4) The fast programmer worked very hard.
Interprétation : The fast **working** programmer worked very hard.

(6.5) Sarah began a book.
Interprétation : Sarah began **to read** a book.

Noms composés

Dans un nom composé, la relation binaire qui lie les référents des noms intervenant dans le composé est laissée implicite. Comme les exemples suivants l'illustrent, cette relation, si elle est souvent inférée sur la base des connaissances du monde, peut aussi être donnée par le contexte linguistique [127, 88].

(6.6) The California student produced an excellent report.

La relation la plus plausible qui puisse être déduite à partir des connaissances du monde entre un étudiant ou, plus généralement, une personne et un état est la relation *être originaire de*. D'où il résulte que l'interprétation la plus plausible du GN *the California student* dans l'exemple (6.6) est celle où l'étudiant est originaire de Californie. Néanmoins, [88, 127] montre qu'en contexte, d'autres relations sont non seulement possibles mais aussi plus plausibles :

(6.7) Each student was assigned a state to study. The California student produced an excellent report.

Dans l'exemple (6.7), le co-texte rend plausible une nouvelle relation entre *étudiant* et *Etat* à savoir celle d'*être affecté à l'étude de l'Etat*. De fait, c'est cette relation qui est utilisée pour reconstruire l'interprétation sous-spécifiée du composé si bien que l'interprétation privilégiée est dans ce cas non pas *l'étudiant originaire de Californie* mais plutôt *l'étudiant affecté à l'étude de l'Etat de Californie*.

Dans un contexte linguistique non nul, une relation rendue accessible par ce contexte peut donc prendre le pas sur les relations rendues plausibles par les connaissances du monde. Cependant, ceci n'est pas systématique et la résolution de la relation implicite dépend au final d'une interaction complexe entre connaissances et contextes, linguistiques et extra-linguistiques d'une part, cohérence, informativité et minimalité d'autre part. Nous montrons maintenant à l'aide d'exemples comment ces divers facteurs interagissent pour déterminer l'interprétation privilégiée.

[36] Le terme «anaphore relationnelle» est dû au fait que dans chacune de ces constructions, une relation est laissée implicite qui peut être reconstruite à partir du contexte linguistique et des connaissances du monde.

Minimalité. La différence entre (6.6) et (6.7) peut être expliquée par l'application du critère de minimalité. Lorsque le contexte linguistique est nul, l'interprétation du groupe nominal (GN) composé est déterminée exclusivement par les connaissances du monde. Par contre, lorsque le contexte linguistique met à disposition une relation plausible pour l'interprétation du GN, cette relation est utilisée de préférence à celle(s) rendue(s) plausible(s) par notre connaissance du monde parce qu'elle permet une interprétation plus «cohésive» de l'énoncé : la même relation est évoquée à deux reprises par l'énoncé plutôt que deux relations distinctes dans le cas où la relation est déterminée par les connaissances du monde.

Plus concrètement, supposons que les interprétations possibles de l'exemple (6.7) aient été construites par les algorithmes de résolution idoines. On aura alors les deux interprétations paraphrasées en (6.8) – la première utilisant une relation donnée par le contexte linguistique, la seconde une relation donnée par les connaissances du monde.

(6.8) a. *The student assigned to study California produced an excellent report*
b. *The student from California produced an excellent report.*

Pour chacune de ces interprétations, les modèles localement minimaux sont les modèles $\mathcal{M}_a$ et$\mathcal{M}_b$ respectivement[37].

(6.9) $\mathcal{M}_a = \{\mathit{student}(c_1), \mathit{assigned}(c_1, \mathit{study}(c_1, c_2)), \mathit{state}(c_2),$
$\mathit{produced}(c_1, c_3), \quad \mathit{excellent\_rep}(c_3), \mathit{cal}(c_2)\}$
$\mathcal{M}_b = \{\mathit{student}(c_1), \mathit{assigned}(c_1, \mathit{study}(c_1, c_2)), \mathit{state}(c_2),$
$\mathit{produced}(c_1, c_3), \mathit{excellent\_rep}(c_3), \mathit{cal}(c_2), \mathit{from}(c_1, c_2)\}$

De ces deux modèles, le plus petit, c'est-à-dire le modèle localement minimal est $\mathcal{M}_a$. Si comme il est proposé ci-dessus, le critère de minimalité suppose que le plus petit (et donc le plus simple) modèle satisfaisant la représentation du discours détermine l'interprétation privilégiée, alors *par minimalité*, (6.8a) peut être identifiée comme représentant l'interprétation privilégiée de (6.7).

Cohérence. L'exemple suivant illustre le rôle que peut jouer le critère de cohérence dans la désambiguïsation des noms composés.

(6.10) After each student was assigned a state to study, the California student, who had been assigned Ohio, started work immediately.

Comme dans l'exemple (6.7), deux interprétations sont *a priori* possibles pour la deuxième proposition :

(6.11) a. *The student from California, who had been assigned to study Ohio, started work immediately.*
b. *The student assigned to study California, who had been assigned to study Ohio, started work immediately.*

Si par implicature de quantité [77], *a state* est interprété comme signifiant *exactement un Etat*, l'interprétation (6.11b) aboutit à une contradiction si bien que l'interprétation préférentielle est en ce cas (6.11a).

Mais les choses ne sont pas toujours aussi simples. Considérons le texte 6.12 par exemple.

(6.12) Each student was assigned a state to study. The California student agreed to work on Ohio.

[37] Notons que $\mathcal{M}_b$ contient une information non impliquée par l'interprétation (6.8b) à savoir que l'étudiant originaire de Californie a été affecté à l'étude de la Californie. Ceci est un autre exemple de cas tels que 6.1 ci-dessus où la minimalité engendre des interprétations sur-spécifiées.

Il y a de nouveau deux possibilités pour résoudre la relation entre *California* et *student* :

(6.13) a. *The student from California agreed to work on Ohio.*
b. *The student assigned to study California agreed to work on Ohio.*

Aucune de ces interprétations n'est incohérence ou redondante et, par le même raisonnement que pour l'exemple (6.8), le principe de minimalité établira une préférence pour (6.13b). Malheureusement, c'est (6.13a) qui est ici l'interprétation privilégiée. Une raison plausible en est que l'interprétation (6.13b) est incohérente avec une présupposition du texte (6.12). En effet, si *x agreed to P* présuppose *x was asked to P*[38], alors les présuppositions des deux interprétations données en (6.13) sont :

(6.14) a. *The student from California was asked to work on Ohio.*
b. *The student assigned to study California was asked to work on Ohio.*

Si *x is assigned to study California* implique *x is asked to work on California* et, toujours dans l'hypothèse de l'application du critère de minimalité, alors la présupposition (6.14b) est incohérente. Comme le montre [190], une présupposition incohérente résulte en un discours peu acceptable ce qui explique l'inacceptabilité de l'interprétation (6.13b).

Informativité. Enfin l'informativité peut contribuer à déterminer l'interprétation des GNs composés.

(6.15) Each student was assigned a state to study. The California student was assigned California.
a. *The student from California was assigned California.*
b. *The student assigned to study California was assigned California.*

Bien que l'interprétation (6.15b) soit plus cohésive que l'interprétation (6.15a), elle n'est pas préférentielle parce qu'elle aboutit à un énoncé non-informatif.

Comme pour la cohérence, les présuppositions peuvent interagir avec le critère d'informativité pour déterminer l'interprétation privilégiée du discours considéré.

(6.16) Each student was assigned a state to study. The California student agreed to work on California.

Si comme auparavant, le verbe factif *x agreed to P* est admis comme présupposant *x was asked to P*, alors les présuppositions des deux interprétations de (6.16) sont :

(6.17) a. *The student from California was asked to work on California.*
b. *The student assigned to study California was asked to work on California.*

Parce qu'elle est non-informative, la présupposition (6.17b) exclut l'interprétation correspondante à savoir : *The student assigned to study California agreed to work on California.*

Descriptions définies

Depuis [186], une description définie est généralement analysée comme présupposant l'existence d'une entité unique satisfaisant cette description. Il est par ailleurs bien connu que cette entité peut «exister» de différentes manières à savoir : par son introduction explicite dans le contexte linguistique (anaphore simple, 6.18a) ; par inférence à partir du contexte linguistique (anaphore associative, 6.18b) ou même (6.18c) grâce à une stratégie de «réparation», l'accommodation [180], qui consiste à «réparer» le contexte de façon à satisfaire les exigences de la langue,

[38] Qu'il s'agisse bien d'un présupposition est confirmé par le fait que l'implication *x was asked to P* est préservé par la négation. En d'autres termes, *x was asked to P* est une présupposition de *x agreed to P* parce que *x was asked to P* est impliqué à la fois par *x agreed to P* et par *x didn't agree to P*.

en l'occurrence par l'introduction dans le contexte d'une entité supplémentaire satisfaisant la description *la femme*.

(6.18) a. Une femme dort. La femme rêve. (*anaphore simple*)
b. Une voiture explose. Le moteur était défectueux. (*anaphore associative*)
c. La femme rêve. (*accommodation*)

Dans les deux premiers cas, un raisonnement basé sur les connaissances peut contribuer à déterminer l'antécédent de la description définie. Par exemple, étant donnés les énoncés :

(6.19) a. Le mari de Mia aime Sally. La chanteuse est célibataire.
b. Le mari de Mia aime Sally. La chanteuse est mariée.

Les connaissances encyclopédiques nous amènent à conclure que *la chanteuse* en (6.19a) réfère à Sally plutôt qu'à Mia. La résolution inverse (*la chanteuse* réfère à Mia) est exclue car elle aboutirait à une **contradiction** : on ne peut être à la fois célibaire et mariée. En (6.19b), l'interprétation préférentielle est également celle où *la chanteuse* réfère à Sally mais cette fois le raisonnement suivi pour atteindre cette conclusion repose sur la notion d'**informativité** plutôt que de contradiction : si *la chanteuse* est Mia, la deuxième proposition exprime un fait déjà impliqué par la première à savoir, le fait que Mia est mariée.

De même, la résolution des anaphores associatives est contrainte par les connaissances encyclopédiques (exemple de [6]) :

(6.20) a. Jean vient d'arriver. Le chameau est dehors.
b. Jean vient d'arriver. Les puces sont dehors.

Dans le premier cas, l'interprétation préférentielle de la description définie *le chameau* est celle où ce chameau est le moyen de locomotion utilisé par Jean pour arriver. Informellement, les inférences suivantes sont faites :

1. Jean est arrivé en utilisant un certain mode de locomotion,
2. un chameau peut être un mode de locomotion et
3. afin de maximiser la **cohésion** du texte, le (référent du) chameau est identifié avec le (référent du) mode de locomotion de Jean.

Lors de l'interprétation de (6.20b) en revanche, ce raisonnement n'est pas fait puisque *les puces* ne sont pas (normalement) un moyen de locomotion. *Les puces* ne peuvent pas non plus être identifiées avec Jean (un être humain ne peut être un insecte). Comme la description définie ne peut être résolue ni au moyen d'un lien anaphorique, ni au moyen d'une anaphore associative, elle est «accommodée» et son référent ajouté au contexte existant.

Des cas plus complexes de résolution apparaîssent dès lors que les relations entre éventualités contraignent de façon indépendante la résolution du défini :

(6.21) Jean est arrivé à trois heures. Le chameau était dehors
a. mais il est parti il y a dix minutes.
b. et Jean l'a trouvé splendide.

La continuation (6.21a) entraîne une interprétation préférentielle où le chameau est de nouveau le moyen de locomotion de Jean. Par contraste, (6.21b) favorise une interprétation où le chameau n'est pas le mode de locomotion de Jean, mais plutôt un élément de la scène d'arrivée de Jean (le chameau était sur place lors de l'arrivée de Jean). Dans le premier cas, la description définie *le chameau* est résolue au moyen d'une anaphore associative, dans le second cas, elle est accommodée.

Le raisonnement qui conduit à ces conclusions divergentes peut être résumé de façon informelle comme suit. La première clause dénote un événement ev_1 et la seconde, un état et_2. Les deux clauses étant au passé, on peut conclure (cf. [102, 84]) que et_2 et ev_1 se recouvrent, c'est-à-dire qu'ils sont partiellement ou totalement simultanés. De plus l'événement dénoté par la première clause est un accomplissement se composant d'un état préparatoire etp_1 suivi d'un point de culmination cul_1 suivi d'un état résultant etr_1. En résumé, les relations temporelles suivantes sont spécifiées :

ev_1 o et_2	ev_1 et et_2 se recouvrent
etp_1)(cul_1)(etr_1	etp_1, cul_1 and etr_1 sont contigus

Par ailleurs, ev_1 (l'événement d'arrivée) implique une transition d'un état (etp_1) où Jean n'est pas sur le lieu d'arrivée, vers un état (etr_1) où Jean y est et donc indirectement, une transition d'un état où le moyen de locomotion de Jean n'est pas sur le lieu d'arrivée vers un état où il y est. En d'autres termes, durant l'état préparatoire etp_1, le moyen de locomotion de Jean n'est pas sur le lieu d'arrivée alors que pendant l'état résultant etr_1, il y est.

Maintenant considérons les relations temporelles possibles entre l'état préparatoire etp_1 et l'état et_2. Deux relations sont possibles : soit les états se recouvrent, soit ils ne se recouvrent pas.

Dans le premier cas de figure (etp_1 et et_2 se recouvrent), l'identification du chameau avec le moyen de locomotion de Jean entraînerait une contradiction puisqu'elle résulterait en deux états partiellement simultanés mais ayant des propriétés **contradictoires** (le moyen de locomotion de Jean, c'est-à-dire le chameau, est sur le lieu d'arrivée et n'y est pas). En conséquence, le chameau fait l'objet d'une accommodation – c'est l'interprétation apparaissant en (6.21b).

Par contre, si etp_1 et et_2 ne se recouvrent pas, le chameau peut être identifié avec le moyen de transport de Jean sans pour autant introduire une contradiction. Comme cette identification contribue à rendre le discours plus **cohésif** (une même entité est mentionnée plusieurs fois plutôt que des entités distinctes), cette interprétation associative de l'anaphore introduite par le défini est préférée à une interprétation mettant en jeu la stratégie d'accommodation – c'est l'interprétation intervenant en (6.21a).

6.1.5 Conclusion

Les exemples ci-dessus montrent que l'interprétation des anaphores relationnelles et nominales peut mettre en jeu un processus d'inférence basé sur les connaissances du monde ainsi que sur la situation linguistique et extra-linguistique. Trois facteurs contribuent à déterminer l'interprétation privilégiée des discours anaphoriques qui tous trois font intervenir l'inférence et peuvent être vérifiés par le biais de l'utilisation conjointe d'un prouveur de théorèmes et d'un générateur de modèles. L'interprétation résultant de la résolution est-elle :

1. *cohérence ?* – l'interprétation n'introduit pas de contradiction ;
2. *informative ?* – l'interprétation ajoute du sens au sens communiqué par le contexte existant ;
3. et *cohésive ?* – l'interprétation maximise le nombre de références faites aux entités mentionnées dans l'énoncé.

6.2 Le réciproque «each other»

La section précédente montre l'utilité de la notion de *minimalité locale* dans la résolution des anaphores : étant donné plusieurs interprétations informatives et cohérentes possibles, l'interprétation privilégiée d'un discours contenant une anaphore nominale ou relationnelle sera celle satisfaite par le modèle qui est localement minimal. Rappelons qu'un modèle localement minimal est un modèle satisfaisant la spécification avec un nombre minimal de faits et d'individus.

Comme l'indique la littérature, certaines constructions linguistiques semblent faire intervenir un principe qui va à l'encontre du principe de cohésion privilégiant les interprétations localement minimales. Ainsi, [38] montre que l'interprétation du réciproque «l'un l'autre» est sujette à un principe, appelons-le «Principe d'informativité», qui tend à privilégier les interprétations faisant intervenir un nombre maximal de faits. Nous montrons maintenant comment la génération de modèles peut être utilisée pour rendre compte de ce principe opposé au principe de minimalité, et plus particulièrement, comment il peut servir à lever l'ambiguïté des énoncés réciproques.

Dans la littérature linguistique (cf. [124, 38], le réciproque *l'un l'autre* est généralement vu comme dénotant un quantificateur dyadique prenant pour arguments un ensemble du premier d'ordre (l'ensemble antécédent) et une relation binaire (la relation de portée). Ainsi par exemple, la représentation sémantique de (6.22) sera (6.22a) où RCP représente le quantificateur dyadique, $\{j,g\}$ l'ensemble antécédent et $\lambda x \lambda y.lk(x,y)$ la relation de portée.

(6.22) Jennifer and Gabrielle are looking at each other
 a. $RCP(\{j,g\})(\{\lambda x\lambda y.lk(x,y)\})$
 b. $\forall x(x \in \{g,j\} \rightarrow \forall y(x \neq y \wedge y \in \{g,j\} \rightarrow lk(x,y)))$

Lorsque l'ensemble antécédent ne contient que deux éléments, chacun des éléments doit se trouver dans la relation de portée avec l'autre élément de l'ensemble antécédent, si bien que l'interprétation de (6.23) peut être représentée par la formule (6.23b)[39]. Lorsque l'ensemble est plus grand en revanche, la littérature sur les réciproques a identifié une variété d'interprétations possibles. Considérons les exemples suivants :

(6.23) The cousins like each other
 a. $\forall x(cousin(x) \rightarrow \forall y(x \neq y \wedge cousin(y) \rightarrow like(x,y)))$

(6.24) The pirates stared at each other
 a. $\forall x(pir(x) \rightarrow \exists y(x \neq y \wedge pir(y) \rightarrow stare(x,y)))$

(6.25) The children gave each other measles
 a. $\forall x(ch(x) \rightarrow \forall y(x \neq y \wedge ch(y) \rightarrow measles(x,y) \vee measles(y,x)))$

L'énoncé (6.23) n'est vrai que si pour chaque paire (x,y) dans l'ensemble antécédent dénoté par *les cousins*, *x like y*. En revanche, une interprétation similaire est impossible pour les énoncés (6.24) et (6.25) : une personne ne peut fixer qu'une seule autre personne à la fois et on ne peut attraper la rougeole qu'une seule fois. Plus généralement, [124, 38] montre clairement que les constructions réciproques peuvent avoir des conditions de vérités très différentes. Dans leur analyse sémantique des réciproques, [38], (DKKMP) utilise le répertoire d'interprétations suivant (où A représente l'ensemble antécédent et R la relation de portée) :

- Strong Reciprocity (SR) :$\forall x,y \in A(x \neq y \rightarrow R(x,y))$
- Intermediate Reciprocity (IR) : $\forall x,y \in A \exists z_1,\ldots,z_n \in A(x \neq y \rightarrow R(x,z_1) \wedge ldots \wedge R(z_n,y))$
- Oneway Weak Reciprocity (OWR) : $\forall x \in A \exists y \in A(x \neq y \wedge R(x,y))$
- Intermediate Alternative Reciprocity (IAR) : $\forall x,y \in A \exists z_1,\ldots,z_n \in A(x \neq y \rightarrow (R(x,z_1) \vee R(z_1,x)) \wedge \ldots \wedge (R(z_n,y) \vee R(y,z_n))$
- Inclusive Alternative Ordering (IAO) : $\forall x \in A \exists y \in A(x \neq y \wedge (R(x,y) \vee R(y,x))$

[39]Dans ce cas, le quantificateur RCP peut être représenté par le lambda terme $\lambda A \lambda R \forall x(x \in A \rightarrow \forall y(x \neq y \wedge y \in A \rightarrow R(x)(y)))$.

Ces interprétations sont dans une relation d'implication : SR implique IR, IR implique OWR et IAR, OWR et IAR impliquent IAO. S'appuyant sur cette observation, DKKMP postule le principe suivant pour déterminer l'interprétation d'un énoncé réciproque :

> **Strongest Meaning Hypothesis (SMH).** Si E est un énoncé réciproque et si Γ est une formule logique représentant le sens du co-texte, la situation discursive et les connaissances encyclopédiques pertinentes, alors les interprétations privilégiées de E sont les interprétations (i) qui sont les plus fortes d'un point de vue logique et (ii) qui sont cohérentes avec Γ.

La SMH sélectionne donc parmi l'ensemble des interprétations qui peuvent être celles d'un énoncé réciproque, l'interprétation la plus forte d'un point de vue logique qui est compatible avec le contexte. Elle ne dit pas en revanche, comment cette interprétation peut être calculée.

Pour ce faire, une première possibilité est d'adopter l'approche proposée par [13] pour le traitement des présuppositions. Etant donné un énoncé ambigu E, cette approche consiste à donner en parallèle à un démonstrateur de théorèmes et à un générateur de modèles, chacune des représentations possibles ϕ de cet énoncé. Si le démonstrateur de théorèmes prouve $\neg\phi$, ϕ est incohérent et l'interprétation peut être éliminée. Par contre, si le générateur de modèles génère un modèle satisfaisant ϕ , ϕ est une interprétation cohérente et donc possible, pour autant qu'elle satisfasse les autres critères de désambiguïsation. Cette approche peut être adaptée aux réciproques de la façon suivante.

1. Chaque construction réciproque a pour représentations sémantiques les cinq représentations correspondant aux interprétations identifiées par DKKMP (SR,IR,OWR,IAR,IAO),
2. les interprétations incohérentes sont éliminées par le biais du raisonnement automatique et
3. l'interprétation cohérente la plus forte (l'*interprétation maximale*) est choisie comme étant l'interprétation la plus plausible.

Le mérite de [13] est d'avoir montré qu'une telle approche fonctionne en pratique pour une classe de problèmes donnée, à savoir la projection des présuppositions. Néanmoins, comme la logique du premier ordre est indécidable, il reste possible que ni le démonstrateur de théorèmes, ni le générateur de modèles, ne produisent une réponse. D'un point de vue linguistique, l'approche est en outre peu attractive du fait qu'elle postule, plutôt qu'elle ne prédit, l'ambiguïté du réciproque *l'un l'autre*.

Une seconde possibilité – celle que nous explorerons ici et qui est expliquée plus en détail par [64] – est d'utiliser la génération de modèles pour générer la signification en contexte d'un énoncé réciproque donné. Les trois grands traits de cette approche sont les suivants :

1. L'interprétation des énoncés réciproques est basée sur l'interprétation IAO.
2. Chaque paire du domaine antécédent qui n'est pas dans la relation de portée est également dans une relation de coût.
3. Le modèle minimal par rapport à cette relation de coût donne l'interprétation de l'énoncé.

Considérons ces points un par un. Tout d'abord le rôle joué par l'interprétation IAO dont la représentation sémantique est répétée en (6.26) ci-dessous.

(6.26) $\mathrm{RCP}_{IAO} = \lambda P \lambda R \forall x(P(x) \rightarrow \exists y(P(x) \wedge x \neq y \wedge (R(x,y) \vee R(y,x))))$

Cette représentation reflète l'interprétation minimale possible pour un énoncé réciproque : elle est donc impliquée par toutes les interprétations possibles d'un énoncé réciproque. Cependant, elle ne rend pas compte de la préférence manifestée par ces énoncés pour une interprétation maximale. Pour mieux situer le problème, supposons que l'on ait l'énoncé (6.27a). Pour cet énoncé, le modèle minimal satisfaisant la représentation (6.27b) est (6.27c). Or ce modèle ne capture pas de façon acceptable l'interprétation de (6.27a) qui est bien capturée en revanche par le modèle (6.27d).

(6.27) a. Sue, Sarah et Sally s'aiment l'une l'autre.
b. $\mathrm{RCP}_{IAO}(\{sue, sarah, sally\})(\lambda x\lambda.aime(x,y))$
c. $\{aime(sue, sarah), aime(sarah, sally)\}$
d. $\{aime(sue, sarah), aime(sarah, sally), aime(sue, sally),$
$aime(sally, sarah), aime(sarah, sue), aime(sally, sue)\}$

Comme cet exemple l'illustre bien, l'interprétation privilégiée d'un énoncé réciproque est celle qui maximise la relation de portée, ici : (6.27d).

Pour capturer cette intuition, nous introduisons le prédicat spécial \$ et privilégions les modèles \$-minimaux. Rappelons (cf. Chapitre 5) que la minimalité spécifique à un prédicat (en l'occurence le prédicat \$) privilégie les modèles qui minimisent la cardinalité de ce prédicat. En sélectionnant les modèles \$-minimaux, nous privilégions donc les modèles où la cardinalité de la relation \$ est minimale. L'idée est la suivante. L'interprétation maximale (et privilégiée) d'un énoncé réciproque est une interprétation qui maximise la cardinalité de la relation de portée. Or la maximisation d'une relation est aussi la minimisation de son ensemble complément. Ainsi la minimisation de la cardinalité du prédicat \$ vise à maximiser celle de la relation de portée : la relation \$ est définie comme étant la relation complémentaire de la relation de portée.

Nous détaillons maintenant le fonctionnement de cette approche par le biais d'un exemple, l'exemple (6.27) ci-dessus. La représentation des énoncés réciproques repose sur la définition suivante :

(6.28) $\mathrm{RCP} = \lambda P\lambda R(RCP_{IAO}(P)(R) \wedge \forall x\forall y(P(x) \wedge P(y) \wedge x \neq y \wedge \neg R(x,y) \equiv \$(x,y)))$

Dans cette définition, la première partie associe aux énoncés réciproques, l'interprétation la plus faible possible, à savoir l'interprétation IAO. Comme cette interprétation est impliquée par toutes les interprétations possibles des énoncés réciproques, cette exigence est compatible avec le fait que l'énoncé puisse avoir une interprétation plus forte.

La deuxième partie impose que chaque paire d'éléments distincts *(x,y)* qui n'est pas dans la relation de portée est dans la relation \$ (intuitivement cette paire augmente le coût de l'interprétation).

Etant donné cette nouvelle représentation du quantificateur RCP, (6.27a) a pour modèles (6.29a) et (6.29b).

(6.29) a. *{aime(sue,sarah),aime(sarah,sally),\$(sue,sally),*
\$(sally,sarah),\$(sarah,sue),\$(sally,sue)}
b. *{aime(sue,sarah),aime(sarah,sally),aime(sue,sally),*
aime(sally,sarah),aime(sarah,sue),aime(sally,sue)}

Comme la cardinalité de \$ est quatre dans (6.29a) et zéro dans (6.23b), (6.23b) est le modèle \$-minimal et reflète (correctement) l'interprétation privilégiée de l'exemple (6.27a).

Pour l'exemple (6.24), une forte contrainte encyclopédique entre en jeu qui est qu'on ne peut fixer du regard qu'une seule personne à la fois. Les modèles \$-minimaux compatibles avec cette contrainte sont les modèles où chaque pirate fixe exactement un autre pirate. Ainsi dans un univers contenant cinq pirates, les interprétations privilégiées sont données par les modèles où la cardinalité de la relation de portée *x fixe y* est cinq. Les modèles qui suivent en sont des exemples (pour faciliter la lecture, les propositions \$ sont omises et remplacées par une indication du coût du modèle, c'est-à-dire par la cardinalité du complément de la relation de portée).

(6.30) *{fixe(a,b),fixe(b,c),fixe(c,d),fixe(d,e),fixe(e,b)}* $\$ = 15$

(6.31) *{fixe(a,b),fixe(b,c),fixe(c,d),fixe(d,e),fixe(e,a)}* $\$ = 15$

L'exemple (6.25) illustre un cas intermédiaire en ce qui concerne la force des conditions de vérités. Les connaissances encyclopédiques nous disent que le relation de portée *x donne la rougeole à y* est assymmétrique et que chaque individu peut être contaminé par au plus une personne. Etant donné un ensemble de cinq enfants, les modèles (6.32) et (6.33) sont tous deux des modèles acceptables pour (6.25), (6.32) représentant l'interprétation privilégiée.

(6.32) *{dr(a,b),dr(a,c),dr(b,d),dr(c,e)}* $\$ = 16$

(6.33) *{dr(a,b),dr(b,d),dr(c,e)}* $\$ = 17$

Ces exemples illustrent le fonctionnement de l'analyse proposée. Ils montrent comment, à partir d'une représentation unique, diverses interprétations peuvent être dérivées qui correspondent aux différentes interprétations possibles des énoncés réciproques.

Cette approche montre que la génération de modèles peut être utilisée pour définir une implantation presque fidèle du modèle linguistique présenté par [38] : elle fait, à quelques exceptions près, les même prédictions (cf. [64] pour une comparaison des deux approches).

Plus généralement, la génération de modèles et le concept de minimalité permettent de *construire*, de *comparer* et d'*ordonner* les interprétations concurrentes d'un énoncé. Dans le cas des réciproques, l'ordre est donné par la cardinalité de la relation de portée mais pour d'autres phénomènes, d'autres critères d'ordre ont été proposés dans la littérature. Ainsi nous avons vu au début de ce chapitre que la désambiguïsation d'anaphores nominales et relationnelles fait intervenir la minimalité locale. [10] en revanche, montre que la minimisation de prédicats peut être utilisée pour modéliser l'interprétation des anaphores pronominales.

6.3 Conclusion

En langue naturelle, la levée des ambiguïtés est sujette, outre les facteurs syntaxiques, à des facteurs sémantiques distincts et parfois contradictoires. Nous en avons identifié quatre :

- le principe de cohérence,
- le principe de cohésion (ou de minimalité),
- le principe d'informativité (ou de non redondance),
- le principe de maximalité.

Dans un système de traitement automatique, ces principes peuvent être vérifiés par le biais de prouveurs de théorèmes et de générateurs de modèles. Ainsi par exemple, le système Doris de Johan Bos utilise cohérence et informativité pour filtrer les résolutions d'anaphores nominales.

Néanmoins ces résultats restent préliminaires laissant en particulier deux grandes questions sans réponse :

- Quelles sont les interactions entre ces différents critères ?
- Quelles sont les performances des raisonneurs automatiques sur le type de problèmes soulevés par les données linguistiques ?

Le problème des interactions requiert une analyse linguistique approfondie des critères intervenant dans la résolution des différents types d'expressions ambiguës permises par la langue naturelle. En effet, différentes constructions peuvent être sensibles à différents critères. Par exemple, si l'interprétation des anaphores privilégie les interprétations minimales, celle du réciproque privilégie les interprétations maximales. Étant donné un énoncé incluant à la fois une anaphore relationnelle et un réciproque, il conviendra donc de calculer l'interprétation du réciproque avant de résoudre l'anaphore. De façon plus générale, il est nécessaire, à partir d'études linguistiques approfondies, d'identifier les différents facteurs de résolution entrant en jeu et de les ordonner de façon cohérente.

La question de la performance des raisonneurs automatiques sur des problèmes linguistiques passe quant à elle par une expérimentation systématique à partir de données linguistiques et suggère le développement d'une architecture logicielle supportant cette expérimentation. Cette architecture permettrait à partir d'une suite de tests répertoriant les problèmes d'inférences soulevés par les données linguistiques, de tester les raisonneurs existants, de comparer leur performance et enfin de les optimiser.

Chapitre 7

Raisonnement taxonomique

Peu expressives mais plus adaptées au traitement informatique que la logique des prédicats, les logiques des descriptions sont des candidates naturelles pour certaines des sous-tâches du traitement automatique des langues. Nous considérons ici deux de ces sous-tâches (la lexicalisation et le raisonnement lexical) et montrons en quoi les logiques de descriptions et les prouveurs afférant fournissent des outils particulièrement adaptés à leur traitement.

Les logiques de descriptions forment une famille de logiques avec différents niveaux d'expressivité et de complexité. La logique utilisée ici a la syntaxe suivante :

$$C, D \rightarrow A|\top|\bot|\neg A \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

La sémantique de cette logique est donnée ci-dessous avec Δ le domaine d'interprétation et I la fonction d'interprétation qui associe à chaque concept atomique A, un ensemble $A^I \subseteq \Delta$ et à chaque rôle atomique R, une relation binaire $R^I \subseteq \Delta \times \Delta$.

$$\begin{array}{rcl}
\top^I & = & \Delta \\
\bot^I & = & \emptyset \\
(\neg A)^I & = & \Delta \backslash A^I \\
(C \sqcap D)^I & = & C^I \cap D^I \\
(C \sqcup D)^I & = & C^I \cup D^I \\
(\forall R.C)^I & = & \{a \in \Delta \mid \forall b(a,b) \in R^I \rightarrow b \in C^I\} \\
(\exists R.C)^I & = & \{a \in \Delta \mid \exists b \in C^I \wedge (a,b) \in R^I\} \\
(R^-)^I & = & \{(b,a) \in \Delta \times \Delta \mid (a,b) \in R^I\}\}
\end{array}$$

7.1 La tâche de lexicalisation

Dans la génération de textes, la tâche de lexicalisation consiste à identifier, pour une entrée conceptuelle donnée (par exemple, un ensemble de faits décrivant une situation) un ensemble de mots permettant de verbaliser cette entrée [161].

7.1.1 Paraphrases

Une caractéristique essentielle de la tâche de lexicalisation est qu'elle doit permettre la génération de paraphrases non pas tant pour permettre la diversité des sorties (après tout, dans une application donnée, on se contente généralement d'une verbalisation unique) mais surtout pour maximiser les chances de succès. Supposons par exemple que l'on ait l'entrée conceptuelle suivante (formulée dans le cadre des logiques de descriptions comme un ensemble d'assertions)[40] :

(7.1) p *:Femme*, b *:Bmw*, e *:Conduire*, (e,p) *:agent*, (e,b) *:theme*, s *:Sortir*, (s,p) *:agent*

Les textes (7.2a-b) sont deux verbalisations possibles de ce contenu.

(7.2) a. La femme qui conduit la BMW est sortie.
b. La conductrice de la BMW est sortie.

Dans ce cas précis, les deux paraphrases ont un degré d'acceptabilité identique[41]. Cependant comme les exemples (7.3), (7.4) et (7.5) l'illustrent, ceci n'est pas toujours le cas (le double point d'interrogation précède les paraphrases peu acceptables).

(7.3) a. La femme qui conduit une BMW en fumant des cigares est sortie.
b. ? ? La conductrice fumeuse de cigares d'une BMW est sortie.
c. ? ? La conductrice d'une BMW qui fume des cigares est sortie.

(7.4) a. L'homme qui danse devant les spectateurs est mon mari.
b. ? ? Le danseur qui est devant les spectateurs est mon mari.

(7.5) a. La personne qui vend sa voiture est sortie.
b. ? ? Le vendeur de sa voiture est sorti.

Dans le premier cas (7.3), les verbalisations (7.3b-c) sont maladroites et surtout ne rendent pas compte de la simultanéité temporelle entre la consommation de cigares et la conduite d'une BMW – elles ne sont en ce sens pas des paraphrases de (7.3a). Dans le second cas (7.4), c'est la relation entre adverbe et évènement qui est occultée par la nominalisation : alors que (7.4a) décrit un évènement de danse qui prend place devant les spectateurs, (7.4b) relate un état où un danseur (dansant ou non) se trouve devant les spectateurs. Enfin dans l'exemple (7.5), la théorie du liage interagit avec la nominalisation pour rendre la pronominalisation (*sa*) impossible.

En résumé, si les paraphrases locales du type *femme qui conduit/conductrice* sont parfois possibles, elles ne le sont pas toujours «globalement», c'est-à-dire au niveau de la phrase dans laquelle elles apparaissent. Plus généralement, on sait [53, 54] que deux paraphrases ne sont pas nécessairement «égales» devant des co-textes différents. Ainsi, si les phrases (7.6a-b) expriment bien le même contenu sémantique, elles ne sont pas toujours interchangeables comme le montre les exemples (7.6c-d) :

(7.6) a. Marie a donné des roses à Pierre.
b. Pierre a reçu des roses de Marie.
c. David a persuadé Marie de donner des roses à Pierre.
d. David a persuadé Pierre de recevoir des roses de Marie.

Il est donc nécessaire que le processus de lexicalisation produise toutes les paraphrases locales possibles laissant au processus de génération tactique[42] le soin de filtrer les réalisations globales, c'est-à-dire les combinaisons de mots, impossibles.

[40] Nous utilisons les conventions de notation suivantes : les concepts sont représentés par des mots commençant avec une majuscule tandis que les relations binaires (les rôles) débutent avec une minuscule.

[41] En contexte, leur acceptabilité peut varier.

[42] Le processus de génération tactique produit une phrase bien formée à partir d'une grammaire et d'un ensemble de mots. Ce processus est présenté et discuté au chapitre 9.

7.1.2 Lexicalisation et inférence

Comme il a souvent été remarqué [181, 160], la lexicalisation est une tâche d'inférence typique des logiques de descriptions (LD).

D'une part, *les connaissances* lexicales nécessaires pour résoudre ce problème se prêtent particulièrement bien au type de représentation supportée par les logiques de description, à savoir les ontologies basées sur la relation de subsomption. Ainsi les logiques de description permettent en particulier, de *définir* des concepts par le biais d'axiomes terminologiques et de mettre ces concepts en relation par le biais d'axiomes d'inclusion. Par exemple, les termes *conducteur* et *conductrice* peuvent être définis par les axiomes terminologiques suivants :

$$\mathsf{Conducteur} \doteq \mathsf{Homme} \sqcap \exists \mathsf{agent}^{-}.\mathsf{Conduire}$$
$$\mathsf{Conductrice} \doteq \mathsf{Femme} \sqcap \exists \mathsf{agent}^{-}.\mathsf{Conduire}$$

Tandis que la relation de subsomption reliant ces deux termes au terme *personne* peut être capturée par les axiomes d'inclusion suivants :

$$\mathsf{Homme} \sqsubseteq \mathsf{Personne}$$
$$\mathsf{Femme} \sqsubseteq \mathsf{Personne}$$
$$\mathsf{Femme} \sqcap \mathsf{Homme} \doteq \bot$$

D'autre part, les prouveurs des LDs mettent à disposition un service de *retrieval* qui permet pour un individu i et pour une base de connaissances C, d'extraire l'ensemble des concepts dont i est une instance. Par exemple, étant donné l'ensemble d'assertions donné en (7.1) et les cinq axiomes ci-dessus, une requête de retrieval sur l'individu p retournera les concepts :

$$\mathsf{Conductrice}, \mathsf{Femme}, \mathsf{Personne}$$

En d'autres termes, l'opération de *retrieval* des LDs permet de sélectionner à partir d'un ensemble d'individus, d'un ensemble de faits et d'une base de connaissances regroupant les concepts lexicalisés, les concepts pertinents pour la génération de paraphrases. Ce sont les concepts dont on peut inférer qu'il sont vrais d'un des individus donnés en entrée. En l'occurrence par exemple, l'extraction permet de sélectionner à la fois le concept $\mathsf{Conductrice}$ qui permettra de générer la paraphrase (7.2b) et le concept Femme qui permettra de générer la paraphrase (7.2a)[43].

7.1.3 Lexicalisation et génération tactique

Dans [59], nous proposons un algorithme de génération qui intègre une lexicalisation basée sur l'inférence avec le processus de génération tactique ainsi qu'avec le processus de génération des expressions référentielles.

Cet algorithme de génération prend pour donnée la phase dite de macroplanning, c'est-à-dire de détermination et de structuration du contenu à communiquer ; et a comme entrée, un ensemble d'entités et un ensemble de faits.

[43]Etant donné les concepts sélectionnés, deux autres paraphrases semblent possibles à savoir, *La personne qui conduit la BMW est sortie* et *La personne femelle qui conduit la BMW est sortie*. La première peut être éliminée comme ne verbalisant pas l'ensemble de la sémantique donnée en entrée. La seconde devra être éliminée par un système de préférence qui reste à définir.

Dans un premier temps, les expressions référentielles sont traitées[44] ce qui a pour résultat d'incrémenter l'entrée puisque pour chaque entité qui a déjà été mentionnée dans le discours, un ensemble de faits est calculé qui permet au locuteur d'identifier de façon unique l'entité en question. Ce calcul se fait par un algorithme du type de ceux décrits au chapitre 10.

Dans un second temps, intervient la lexicalisation qui est définie comme suit :

> Si e est une entité, ϕ_e est un ensemble d'assertions vraies de e, KB_{Lex} est une ontologie reflétant la définition des concepts lexicalisés et T est une table reliant concepts et entrées lexicales, alors **lexicaliser e** consiste à :
>
> 1. extraire de l'ontologie les concepts $C_1 \dots C_n$ dont e est une instance et
> 2. extraire du lexique les entrées lexicales associées par T aux concepts $C_1 \dots C_n$ sélectionnés par inférence.

L'hypothèse est d'une part, que le sens des mots et les relations entre ces sens sont encodés dans une base de connaissances utilisée pour extraire les concepts lexicalisés pertinents et d'autre part, que ces concepts lexicalisés sont reliés par une table aux entrées lexicales de la grammaire utilisée par le générateur tactique. Le résultat de la lexicalisation est un ensemble d'entrées lexicales dans cette grammaire.

La troisième phase est la phase de *génération tactique* dont le but est la production d'un syntagme verbalisant la sémantique but à partir d'une sémantique but, d'un ensemble d'entrées lexicales et d'une grammaire. Cette phase peut être exécutée par un algorithme tabulaire standard tel celui présenté par [23] ou par un algorithme par contraintes tel celui décrit au chapitre 9.

Pour une présentation plus précise de cet algorithme, cf. [59]. Nous illustrons maintenant le fonctionnement de cet algorithme par un exemple.

7.1.4 Un exemple

Supposons que la sémantique but et la base de connaissances soit celles données en Figure 7.1.

Sémantique but	Base de connaissance
$\mathrm{I} = \{g, h, e\}$,	$\mathsf{Human} \sqsubseteq \mathsf{Object},\ \mathsf{Kind}$
g $\mathsf{:Woman}, g$ $\mathsf{:Kind}$,	$\mathsf{GrandMother} \equiv \mathsf{Woman} \sqcap \exists \mathsf{hasChild.Parent}$
(g, d_1) $\mathsf{:HasChild}$,	$\mathsf{Mother} \equiv \mathsf{Woman} \sqcap \exists \mathsf{hasChild.Human}$
(d_1, d_2) $\mathsf{:HasChild}$,	$\mathsf{Father} \equiv \mathsf{Man} \sqcap \exists \mathsf{hasChild.Human}$
d_1 $\mathsf{:Human}, d_2$ $\mathsf{:Human}$	$\mathsf{Man} \equiv \mathsf{Human} \sqcap \mathsf{Male}$
(g, h) $\mathsf{:with}$, h $\mathsf{:Hat}$,	$\mathsf{Male} \equiv \neg \mathsf{Female}$
e $\mathsf{:Run}$, (e, g) $\mathsf{:agent}$	$\mathsf{Running} \equiv \mathsf{Run} \sqcap \exists \mathsf{agent.Human}$
	$\mathsf{Parent} \equiv (\mathsf{Father} \sqcup \mathsf{Mother})$
	$\mathsf{Woman} \equiv \mathsf{Human} \sqcap \mathsf{Female}$
	$\mathsf{Means} \equiv \exists \mathsf{with.Object}$

FIG. 7.1 – Un exemple d'entrée

[44] La génération d'expressions référentielles vise à produire une description appropriée des entités mentionnées, au niveau de la forme comme au niveau du contenu. Au niveau de la forme, le problème est de choisir, en fonction du contexte discursif et situationnel, le déterminant approprié (déterminant défini, indéfini, démonstratif, pronom etc.). Au niveau du contenu, il est de déterminer en fonction des buts communicatifs (identification, description etc.) et du statut discursif de l'entité décrite (donné *versus* nouveau), le contenu sémantique du syntagme nominal. Pour plus de détails sur la génération d'expressions référentielles, voir Chapitre 10.

Pour faciliter l'exposition, nous ne considérerons ici qu'une seule des dérivations possibles, à savoir celle qui aboutit à la génération de la phrase[45] :

(7.7) The kind grandmother with a hat runs.

L'algorithme commence par extraire les concepts vrais des objets qui sont à décrire à savoir, h, g et e. Parmi ceux-ci, la phase de lexicalisation retournera en particulier *GrandMother*, *Kind*, and *Means* for g ; *Hat* for h ; and *Running* for e.

Ensuite les entrées lexicales correspondantes sont extraites du lexique. Dans l'hypothèse d'une grammaire d'arbres adjoints où l'interface syntaxe/sémantique est implantée comme indiqué dans [60], l'algorithme retournera entre autres, les entrées lexicales données dans la Figure 7.2.

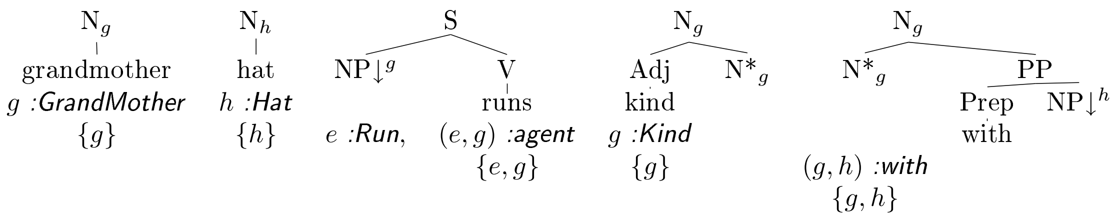


FIG. 7.2 – Résultat de la lexicalisation

Dans l'hypothèse où g est «donné» et h «nouveau»[46], les entrées lexicales pour *the* et *a* seront ajoutées à l'agenda. Celui-ci est alors traité par l'algorithme de réalisation avec pour résultat la construction de l'arbre et de la représentation sémantique donnée en Figure 7.3.

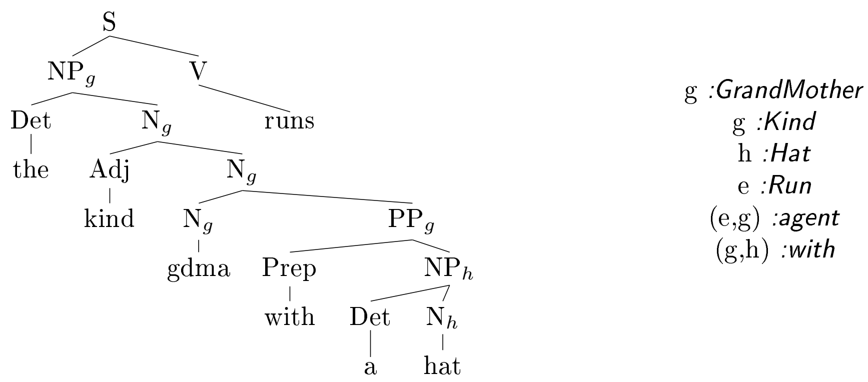


FIG. 7.3 – Résultat de la réalisation

La représentation sémantique associée à cet arbre implique les mêmes individus $\{g, h, e\}$ que l'entrée et pour chacun de ces individus, le même ensemble de propriétés peut être inféré de la base de connaissance à partir de cette sémantique qu'à partir de la sémantique but donnée en entrée. Plus précisément, l'ensemble des concepts les plus spécifiques dont g, h et e sont des instances est exactement le même que pour l'entrée : les concepts les plus spécifiques dont g est

[45]Pour une discussion des heuristiques et optimisations permettant de traiter la combinatoire résultant d'une exploration en parallèle de toutes les possibilités, nous renvoyons le lecteur à [59].

[46]Comme il est d'usage (cf. en particulier le générateur SPUD [183]), nous supposons qu'un historique du dialogue est maintenu qui permet de déterminer le statut discursif des entités : «donné» pour les entités déjà mentionnées ou présentes dans le contexte situationnel, «nouveau» pour les autres.

une instance sont *GrandMother,Kind* et *Means* ; de h, *Hat* et de e, *Running*. L'arbre dérivé est donc bien formé et la phrase (7.7) est générée.

7.1.5 Discussion

Nous avons posé les bases d'une phase de « micro-planning» intégrant une lexicalisation basée sur l'inférence taxonomique, la génération d'expressions référentielles et la réalisation de surface. Pour valider cette approche, plusieurs directions restent à explorer.

En premier lieu, il importe de construire une ontologie lexicale de taille plausible (correspondant au moins à une grammaire «noyau» du français) qui permette d'une part, de mieux comprendre la structuration des concepts lexicaux et leur relation aux items lexicaux et d'autre part, d'évaluer l'efficacité en «taille réelle» d'une lexicalisation basée sur l'inférence.

En second lieu, il est nécessaire d'explorer l'interaction entre la tâche de génération d'expressions référentielles, celle de lexicalisation et de celle de réalisation. Comme le montre [67], la génération d'expressions référentielles peut-elle aussi faire intervenir une inférence basée sur des connaissances lexicales (en particulier, dans le cas d'anaphores associatives du type *une maison/la porte*). Une première question concerne l'interaction entre les connaissances lexicales nécessaires pour cette tâche et les connaissances lexicales requises pour la lexicalisation. Ces connaissances doivent-elles être intégrées dans une même base de connaissances ? Quelles sont précisément les requêtes utilisées ? et comment peut on construire une base de connaissances adéquate ? Plus généralement, quel est le contenu d'une ontologie lexicale ? Et quels sont les critères permettant de définir ce contenu ?

L'interaction entre lexicalisation et réalisation pose elle aussi des questions intéressantes du fait en particulier, de la complexité impliquée. Synonymies, variantes terminologiques et variantes morpho-syntaxiques font que le nombre d'entrées lexicales sélectionnées par le processus de lexicalisation est en général très élevé. La combinatoire résultante est extrêmement complexe et il est en conséquence nécessaire d'optimiser le processus de réalisation. Nous reviendrons sur ce point au chapitre V.

7.2 Systèmes de question/réponse

Nous venons de voir comment les logiques de descriptions peuvent être mises en jeu pour modéliser la tâche de lexicalisation dans les systèmes de génération. Une autre application où leur utilisation est naturelle concerne le raisonnement lexical dans les systèmes de Question/Réponse (QR).

Dans les années 70 et 80, la recherche a porté sur le développement de systèmes QR permettant à l'utilisateur de stocker l'information dans des bases de données sans qu'il soit nécessaire ni de comprendre comment ces données étaient stockées, ni de savoir comment exprimer une requête relationnelle. Cette recherche visait à développer des systèmes interactifs permettant à l'utilisateur humain un dialogue en langue naturelle avec le système. Malheureusement, à cette époque, l'ordinateur individuel étant peu répandu, peu d'utilisateurs ressentaient le besoin d'interroger des bases de données en langue naturelle si bien que la recherche dans ce domaine mourut d'une mort naturelle dans le courant des années 80.

Depuis quelques années cependant, les systèmes automatiques de question/réponse connaissent un regain d'intérêt avec en particulier, l'introduction en 1999 aux États Unis de l'axe *Question Answering* dans la *Text Retrieval Conference*[47]. Ces nouveaux systèmes cherchent une réponse

[47] TREC `http://trec.nist.gov/` organise des compétitions ouvertes dans des domaines spécifiés chaque année,

aux questions posées non plus dans une base de données mais sur le web.

Si ces systèmes plus récents mettent le plus souvent en oeuvre un traitement linguistique superficiel des questions et des réponses, les dernières évaluations indiquent qu'un traitement sémantique plus approfondi permettrait d'améliorer leur précision [192]. Dans cette perspective, certains chercheurs ont commencé à explorer le rôle que pourrait jouer l'inférence dans ces systèmes [194]. Afin de pouvoir utiliser l'inférence d'une façon qui permette ce que [7] appelle l'évaluation de performance par composantes – évaluer la performance des composantes du système et déterminer leur impact sur la performance du système dans son ensemble – il est nécessaire d'identifier des sous-tâches spécifiques à ces systèmes qui soient potentiellements aptes à bénéficier de l'inférence.

Dans les premières générations de systèmes QR, l'inférence a été utilisée essentiellement pour permettre la production de réponses dites «cooopératives», c'est-à-dire de réponses qui pour des raisons d'ordre pragmatique (fausse présupposition de l'utilisateur, sous-informativité de la réponse directe etc.) vont au-delà du simple oui/non (pour les questions polaires) ou d'une énumération (pour les questions à trou) (cf. en particulier, [105, 132, 131, 151, 12] et, pour un résumé détaillé du rôle de l'inférence dans les premiers systèmes QR [194]).

Pour les systèmes QR de seconde génération, les systèmes mis en concurrence dans la compétition TREC, [194] identifie plusieurs sous-tâches où l'inférence pourrait être mise en jeu :

- Comparaison des questions posées : deux questions sont-elles sémantiquement équivalentes ? ou dans une relation d'implication ?
- Identification des réponses réelles parmi un ensemble de réponses potentielles : la réponse implique t-elle (une représentation modifiée de) la question posée ?
- Comparaison des réponses obtenues. Si deux réponses sont dans une relation d'implication, la réponse la plus sûre (c'est-à-dire celle qui sera le plus sûrement jugée correcte par les évaluateurs) est la réponse la plus spécifique.
- Production de réponses coopératives.

L'article identifie ainsi un ensemble de sous-tâches qui peuvent être résolues par l'inférence et donc, indirectement, un ensemble de tâches d'inférence. Néanmoins, cet article reste très théorique car il présuppose des connaissances encyclopédiques illimitées. Nous poursuivons ici un projet moins ambitieux mais plus réaliste qui consiste à se concentrer sur les cas de question/réponse où le raisonnement impliqué est un raisonnement basé exclusivement sur les connaissances lexicales (par opposition, aux connaissances encyclopédiques). Plus précisément, nous nous concentrons sur la tâche qui consiste à déterminer si un fragment de texte fournit ou non une réponse à une question étant données les connaissances lexicales disponibles.

7.2.1 Approche générale

Dans les systèmes de Q/R sur domaine ouvert, l'approche dominante consiste premièrement à utiliser des techniques d'extraction d'information pour sélectionner les textes potentiellement pertinents et deuxièmement, à identifier les réponses, c'est-à-dire à identifier les entités mentionnées par ces textes dont le type sémantique est subsumé par celui attendu pour la réponse.

Light et al. [128] estiment que la borne supérieure sur la performance de ces systèmes est d'environ 70% (dans l'hypothèse où extraction des passages pertinents, reconnaissance d'entités nommées et classification des questions sont parfaites) montrant ainsi clairement les limites de ces approches. Afin de pallier ce problème, on s'achemine actuellement vers des systèmes complétant l'approche standard à deux étapes (extraction de passages suivie par une comparaison de types

attirant des équipes de R&D du monde entier et stimulant le développement technologique.

sémantiques) par des techniques plus linguistiques. Ainsi, comme nous venons de le voir, [194] propose par exemple d'utiliser l'inférence pour généraliser et comparer les questions posées ; pour déterminer si une réponse potentielle est une réponse correcte ou pour comparer les réponses obtenues et produire des réponses coopératives.

Néanmoins, aussi sophistiqués que soient les prouveurs existants, la quantité de connaissances nécessaires pour tester équivalence et implication entre deux formules représentant le sens d'une paire ⟨ Question, Texte ⟩ arbitraire reste un problème majeur, parce qu'il n'existe ni un modèle, ni une ontologie encodant ces connaissances, et parce qu'il est peu probable que les prouveurs existants puissent traiter la combinatoire résultante.

Plutôt que d'aborder la tâche dans toute sa complexité en utilisant des corpus non restreints et en faisant l'hypothèse de connaissances encyclopédiques et lexicales complètes, nous poursuivons ici une approche dont le but principal est de *développer une architecture permettant l'intégration du raisonnement lexical dans un système de TAL*. Plus spécifiquement, l'approche présentée ici restreint la complexité de la tâche d'inférence de la façon suivante :

- elle considère (dans un premier temps) une *tâche d'inférence unique* à savoir, la tâche qui consiste à déterminer si un texte donné répond à une question étant donné des connaissances de sémantique lexicale standard ;
- elle se concentre sur l'utilisation du savoir *lexical* plutôt que sur l'ensemble des connaissances encyclopédiques, et plus particulièrement sur l'utilisation de quatre relations lexicales standards qui sont : l'hyponymie, la synonymie, l'antonymie et la troponymie ;
- elle se base sur l'utilisation des *Logiques de Descriptions* (LD), plutôt que de la logique des prédicats qui, si elle est plus expressive est aussi d'une complexité informatique plus élevée ;
- elle opère sur un ensemble d'exemples construits plutôt que sur un corpus ouvert. Le but sous-jacent est ici la construction d'une suite de tests à partir de laquelle il soit possible de comparer et d'optimiser les prouveurs existants sur des données provenant de la linguistique plutôt que, comme c'est le cas actuellement, sur des données mathématiques.

7.2.2 Divergences lexicales entre question et réponse

Pour déterminer si oui ou non, un fragment de texte fournit la réponse à une question donnée, les systèmes de Q/R existants utilisent pour la plupart des techniques «sacs de mots» qui privilégient les textes ayant le plus grand nombre de mots en commun avec la requête[48].

Pareil traitement de la relation entre question et réponse n'est de toute évidence qu'une approximation de la tâche à accomplir qui est une tâche de nature intrinsèquement sémantique. Ainsi pour déterminer si un fragment de texte procure une réponse à une question donnée, il est nécessaire de :

- comparer les types sémantiques des mots apparaissant dans la question et dans la réponse. Cela permettra par exemple de rendre compte de la divergence entre la question *Est-ce que Jean aime les pommes ?* et la réponse *Jean aime les fruits*
- comparer les relations intervenant entre ces types. On pourra ainsi détecter que *Marie aime Jean* n'est pas une réponse à la question *Quelle femme Jean aime-t-il ?*
- détecter les équivalences et implications sémantiques pour reconnaître par exemple que *Jean est parti* fournit une réponse à la question *Jean est-il resté ?*

[48] Cette description des techniques «sacs de mots» est une généralisation très approximative. En particulier, seuls les mots pleins (noms, verbes, adjectifs, adverbes) sont généralement pris en compte. En outre, certaines approches permettent une comparaison des types sémantiques plutôt que des mots.

De façon plus générale, l'approche poursuivie ici vise à permettre une comparaison sémantique de paires question/réponse qui puisse rendre compte des divergences liées à la synonymie, l'hyponymie, la troponymie et l'antonymie.

Synonymie. Deux mots sont des synonymes (ou plus exactement d'après la terminologie employée par [35], des quasi-synonymes) s'il sont substituables l'un à l'autre dans certains contextes sans qu'il y ait changement du sens communiqué. Une analyse informelle de corpus de FAQ (Frequently Asked Questions) montre que les variations synonymiques entre question et réponse sont relativement fréquentes. L'exemple suivant est tiré du corpus développé par la Corporation MITRE[49].

(7.8) Who **runs** the club ?
The club, **operated** by a non-profit society made up of school and community volunteers, has sold more than 30 memberships and hired a full-time co-ordinator.

Hyponymie. Cette relation est une relation d'inclusion sémantique : le mot M1 est un hyponyme du mot M2 si toutes les entités qui sont de type M2 sont aussi de type M1. L'exemple suivant (lui aussi tiré du corpus de la Corporation MITRE) illustre une divergence QR par hyponymie.

(7.9) What **four kind of foodstuffs** does the humanitarian aid contain ?
Sheila Watt-Cloutier [...] said project member Terry Fenge [...] reported no problems and that 200 of the 540 boxes of aid supplies – including **flour, sugar, cooking oil, pasta** and medical kits – had already been distributed to needy families.

Troponymie. Un verbe V1 est un troponyme du verbe V2 si V2 implique V1 et spécifie la manière/le but/l'instrument etc. de V1. Ainsi dans l'exemple ci-dessous, *suggest* est un troponyme de *indicate* (*suggérer* c'est *indiquer de façon incertaine*).

(7.10) What do the traps **indicate** about the number of mosquitoes ?
This season, the traps **suggest** that the number of mosquitoes is up almost ten times from the normal average for this time of year.

Antonymie. Des antonymes (ou opposés) sont des mots tels que la sémantique de l'un est équivalente à la négation de la sémantique de l'autre. Pour détecter que le texte (7.11b) est de fait une réponse à la question (7.11a), il est nécessaire de savoir que *rester* est sémantiquement équivalent à *ne pas partir*.

(7.11) a. Where did the Queen Mum **stay** during Germany's World War II bombings ?
b. The Queen Mum won Britain's love and admiration when she refused to **leave** London during Germany's World War II bombings.

Ces quatre relations étant encodées de façon relativement exhaustive dans WordNet [50], il est envisageable de traduire l'information contenue dans WordNet dans une base de connaissances formulée dans le cadre des logiques de descriptions[50]. En première approximation on pourra par exemple, considérer la traduction résumée en Figure 7.4.

[49]Le corpus de la Corporation MITRE est un corpus en langue anglaise de textes conçus pour évaluer les capacités de lecture et de compréhension des collégiens. Il contient des textes, des questions sur ces textes et, pour une partie des questions, les réponses attendues. Pour plus d'information sur ce corpus, cf. [86].

[50]Plus exactement, il est envisageable de traduire une information épurée car WordNet contient des distinctions de sens trop fines et trop nombreuses pour être directement utilisable par l'inférence.

w_1, w_2 sont dans le mêm synset	$\tau(w_1) = \tau(w_2)$
w_1 est un hyponyme de w_2	$\tau(w_1) \sqsubseteq \tau(w_2)$
w_2 est un troponyme de w_1 pour la manière M	$\tau(w_1) \doteq \tau(w_2) \sqcap \exists \mathsf{manner}.\tau(M)$
w_1 est un opposé de w_2	$(\tau(w_1) \sqsubseteq \neg\tau(w_2)$ and $\tau(w_2) \sqsubseteq \neg\tau(w_1))$ or $(\tau(w_1) \doteq \neg\tau(w_2))$

FIG. 7.4 – Encodage dans les LDs des relations WordNet

7.2.3 La tâche d'inférence

Les connaissances considérées seront donc celle encodées par les relations d'hyponymie, de synonymie, d'antonymie et de troponymie. Il reste à spécifier la requête d'inférence qui permettra de déterminer si un texte précis répond à une question donnée. En supposant que question et réponse soient chacune associée par un analyseur à une formule des LDs représentant leur sens (ϕ_Q et ϕ_R respectivement), deux approches sont possibles :

- un test d'implication, $\phi_R \models_{Lex} \phi_Q$, qui intuitivement revient à tester si la réponse est plus spécifique que la question ;
- quatre tests de consistance (voir ci-dessous) qui implantent une approximation de la sémantique formelle proposée pour les questions par [78] .

La première approche a l'avantage d'être moins coûteuse (une requête d'inférence au lieu de quatre) mais est inadéquate telle quelle pour rendre compte des données linguistiques. Elle ne permettra pas en particulier de rendre compte du fait que (7.12b) et (7.13b) sont des réponses aux questions (7.12a) et (7.13a) respectivement.

(7.12) a. Where can John obtain the book on Patanjali ?
b. The book on Patanjali can be obtained from the library.

(7.13) a. Where can the book on Patanjali be obtained ?
b. The book on Patanjali cannot be obtained anywhere.

Dans le premier cas, le problème est qu'un argument mentionné dans la question ne l'est pas dans la réponse empêchant ainsi une relation de spécification entre question et réponse. Dans le second cas, le même problème apparaît (la réponse n'est ni plus ni moins spécifique que la question) en raison cette fois d'une différence de polarité.

En bref, l'approche basée sur un test unique d'implication ne peut être utilisée telle quelle. Il faut pour pouvoir rendre compte des données linguistiques affiner cette requête de façon à rendre compte des différences argumentales et de polarités qui peuvent apparaître entre question et réponse.

La deuxième approche possible (4 tests de consistance) est celle proposée par [18]. Elle part de l'idée que la dénotation d'une question est l'ensemble des réponses possibles à cette question. Dès lors, un texte fournit une réponse à une question donnée s'il est compatible avec au moins une des réponses possibles et incompatible avec au moins une autre de ces réponses. Le test d'incompatibilité permet de vérifier la pertinence de la réponse : un texte non pertinent pour une question donnée sera consistant avec tous les types de réponses possibles pour cette question. Comme il est impossible d'énumérer les réponses possibles à une question (la combinatoire résultante serait trop complexe), [18] propose de décrire les types de réponses possibles pour une question donnée par quatre formules logiques approximant les quatre grands types de réponses qu'une question peut avoir. Si D est le domaine de la question, c'est-à-dire l'ensemble sur lequel porte la question Q, si C est la représentation sémantique du reste de la question (le «corps» de la question) et si T est la représentation sémantique du texte dont on veut savoir s'il répond à Q, alors les quatre formules dont il faudra tester la satisfiabilité afin de déterminer si T est une réponse à la question posée seront :

$$\begin{array}{l} (D \sqsubseteq C) \sqcap T \\ (D \sqcap C) \sqcap T \\ ((D \sqcap C) \sqsubseteq \bot) \sqcap T \\ (D \sqcap \neg C) \sqcap T \end{array}$$

La Figure 7.5 illustre cette approche par rapport à la question *Qui aime Marie ?* P représente le concept *Personne*, a le rôle *aime* et M, le concept *Marie*. La représentation associée par l'analyseur à chacune des phrases considérées figure sous cette phrase. En haut de chaque colonne, figure la formule dont la conjonction avec la représentation sémantique de la phrase analysée forme la formule dont il faut tester la satisfiabilité.

On y voit que les phrases (a-d) sont (correctement) analysées comme étant des réponses à cette question (pour chacune de ces phrases il y a au moins un test inconsistant et un test consistant), alors que *Pierre court* est marqué comme inacceptable dans ce contexte (sa représentation est consistante avec les quatre types de réponses possibles).

	$P \sqsubseteq \exists a.M$	$P \sqcap \exists a.M$	$(P \sqcap \exists a.M) \sqsubseteq \bot$	$P \sqcap \neg\exists a.M$
a. Tout le monde aime Marie $P \sqsubseteq \exists a.M$	C	C	I	I
b. Jean aime Marie $J \sqcap P \sqcap \exists a.M$	C	C	I	C
c. Personne n'aime Marie $(P \sqcap \exists a.M) \sqsubseteq \bot$	I	I	C	C
d. David n'aime pas Marie $D \sqcap P \sqcap \neg\exists a.M$	I	C	I	C
e. David court $D \sqcap P \sqcap C$	C	C	C	C

FIG. 7.5 – Qui aime Marie ?

7.2.4 Discussion

Les sections précédentes présentent les bases d'un traitement sémantique de la relation entre question et réponse. Comme pour la lexicalisation cependant, les «vrais» problèmes concernent l'identification, la structuration et la représentation des connaissances nécessaires à ce traitement : d'une part, la grammaire qui permettra d'associer à question et réponse une représentation sémantique dans le cadre des LDs et d'autre part, la base de connaissances englobant les connaissances lexicales mentionnées plus haut, la synonymie, l'hyperonymie, la troponymie et l'antonymie.

La grammaire devra capturer au mieux les relations paraphrastiques entre énoncés, c'est-à-dire associer aux énoncés ayant une même sémantique compositionnelle, des représentations sémantiques équivalentes (ou même identiques). Nous reviendrons sur ce point au chapitre V.

La base de connaissances devra construire sur l'ontologie contenant les concepts lexicaux évoqué en section 7.1.5 : une fois les concepts lexicaux déterminés et structurés dans une taxonomie, il incombera d'étendre cette taxonomie avec des liens synonymiques, antonymiques et troponymique. Là encore le problème de l'acquisition des connaissances se pose. On pourra s'appuyer à la fois sur les ressources existantes (dictionnaires de synonymes, thesaurus pour les variantes terminologiques, dictionnaires informatisés, eurowordnet etc.) et sur des méthodes d'acquisitions automatiques.

Enfin un troisième élément important concerne l'interaction entre concepts et variantes terminologiques. En effet, la synonymie relie non seulement des mots mais également des «expressions multi-mots». Surtout, elle peut concerner des groupes de mots plutôt que des unités lexicales de base. Dans le cadre des travaux sur l'extraction d'information, des recherches très intéressantes se concentrent sur l'identification de ces variantes. Ainsi [99] montre comment extraire des variantes *morpho-syntaxiques* de syntagmes à partir de corpus spécialisés et des travaux plus récents montrent comment extraire des variantes *intercatégorielles*. Il serait intéressant de se baser sur ces travaux pour produire une grammaire encodant ces connaissances, c'est-à-dire une grammaire qui associe à de telles variantes une sémantique équivalente.

7.3 Conclusion

Parce qu'elles sont optimisées pour la classification, c'est-à-dire pour la construction d'ontologies basées sur la relation de subsomption, les logiques de descriptions offrent un cadre naturel pour le traitement des tâches en langue naturelle qui font intervenir un raisonnement taxonomique.

Dans ce chapitre, nous avons identifié deux de ces sous-tâches (la lexicalisation en génération de textes et le raisonnement lexical dans les systèmes de question/réponse) ; précisé les requêtes d'inférence soulevées par ces problèmes et esquissé une architecture logicielle pour le traitement et l'évaluation de ces requêtes. Mais le gros du travail reste à faire qu'on peut diviser en deux grands axes.

Premièrement, il faut, pour pouvoir raisonner sur le sens des phrases, disposer d'une représentation de ce sens et ce dans le cadre des logiques de descriptions. Même en se restreignant à couvrir un ensemble de phrases prédéfinis et en évitant d'y inclure les phénomènes non compositionnels tels les phénomènes anaphoriques et elliptiques, le problème n'est pas trivial pour deux raisons.

D'une part, la plupart du travail accompli dans le domaine de la construction sémantiques s'est fait dans le cadre soit de la logique des prédicats (ou d'une logique équivalente telle par exemple, la sémantique «plate à trous» de Johan Bos), soit, et c'est plus courant, dans le cadre du lambda calcul typé. Il y a en bref peu de travail fait sur la représentation du sens de la langue naturelle par le biais des logiques des descriptions et en particulier, peu de travail fait sur la définition d'un processus de construction sémantique compositionnel dans le cadre des logiques de descriptions.

D'autre part, le développement d'une «grammaire sémantique» à grande échelle reste peu courant et il n'existe pas à l'heure actuelle de standard (même pour l'anglais) auquel on puisse se référer pour décider de la représentation sémantique à associer à chaque mot. Pour l'anglais, il existe par exemple une grammaire HPSG incluant une sémantique compositionelle mais la dimension sémantique de cette grammaire n'est pas documentée. Afin de faciliter le développement, la maintenance et le déboggage d'une grammaire sémantique pour le français, nous travaillons actuellement en collaboration avec d'autres chercheurs du LORIA, au développement d'une méta-grammaire pour les grammaires d'arbres adjoints. Parce qu'elle permet de spécifier à un niveau relativement abstrait les propriétés des arbres présents dans un lexique TAG, cette métagrammaire facilite la tâche de spécification et garantit une certaine cohérence.

Quatrième partie

Inférence ensembliste

Le troisième type d'inférence examiné dans ce mémoire est l'inférence dite ensembliste. A partir de variables portant sur des ensembles et de contraintes restreignant la valeur de ces variables, l'**inférence ensembliste** permet de déterminer les valeurs possibles des variables considérées.

Comme nous allons le voir, pour certains problèmes linguistiques, une formulation en termes de variables et de contraintes ensemblistes est extrêmement naturelle. Or pareille formulation permet une caractérisation déclarative du problème : elle permet de définir les solutions bien formées plutôt que de décrire une procédure permettant de construire ces solutions. Ainsi, le linguiste n'a pas à spécifier un algorithme mais à définir, par le biais des contraintes ensemblistes, les solutions attendues. Une fois ces contraintes posées, la sémantique formelle des contraintes ensemblistes (intersection, union, différence etc.) permet de déterminer l'ensemble des solutions possibles. En outre, une interprétation procédurale de ces contraintes est «donnée» par certains langages de programmation par contraintes, en particulier par Oz/Mozart (`http://www.mozart-oz.org`). Ces langages permettent en effet de résoudre des systèmes de contraintes ensemblistes, c'est-à-dire d'énumérer les assignations de valeurs aux variables satisfaisant les contraintes données en entrée.

Dans les chapitres qui suivent, nous proposons une formulation ensembliste pour les tâches suivantes :

1. La génération des arbres satisfaisant une description d'arbres. Comme nous le verrons, ce problème trouve différentes applications en linguistique informatique avec en particulier : la production des représentations sémantiques décrites par une représentation sous-spécifiée ; la production des arbres syntaxiques associés à une séquence de mots par une grammaire basée sur des arbres sous-spécifiés (analyse) ; ou inversement (génération), la production des arbres syntaxiques associés à une représentation sémantique par une grammaire basée sur des arbres sous-spécifiés.
2. La génération d'une description discriminante, c'est-à-dire la production pour une entité donnée, dans un contexte donné, d'un ensemble de propriétés permettant d'identifier cette entité de façon unique. La génération de descriptions discriminantes est une sous-tâche des algorithmes pour la génération de textes.

Pour chacune de ces applications, nous montrons comment formuler la tâche en termes de contraintes ensemblistes directement implantables en Oz/Mozart. Nous discutons ensuite des avantages de cette formulation et en particulier, des différences qu'elle entraîne par rapport aux approches existantes aussi bien en termes empiriques qu'en termes informatiques et théoriques.

Chapitre 8

Contraintes ensemblistes et programmation par contraintes

Ce chapitre introduit de façon informelle les notions sur la programmation par contraintes dont nous aurons besoin par la suite. La section 8.1 survole les notions de base de la programmation par contraintes tandis que la section 8.2 détaille les contraintes ensemblistes permises par le langage Oz/Mozart. La section 8.3 conclut en donnant une idée des règles d'inférence utilisées par Oz pour déterminer la valeur de variables ensemblistes, étant donné un ensemble de contraintes sur ces variables.

8.1 La programmation par contraintes

Dans le cadre de la programmation par contraintes, un problème est formulé comme un **problème de satisfaction de contraintes** (CSP, Constraint Satisfaction Problem), c'est-à-dire par :

- un ensemble de variables ;
- un ensemble de domaines (pour ces variables) ;
- un ensemble de contraintes (sur ces variables).

Une **solution** pour un CSP assigne à chaque variable une valeur de son domaine telle que toutes les contraintes soient satisfaites. Ces solutions peuvent être énumérées par un **solveur de contraintes** qui déploiera l'une des stratégies existantes possibles (*generate-and-test, backtracking*, etc.). Le solveur de contraintes utilisé par Oz/Mozart est un solveur basé sur la propagation qui utilise trois éléments pour déterminer une solution :

- la propagation de contraintes (inférence déterministe),
- la recherche non déterministe (distribution),
- une stratégie d'exploration de l'espace de recherche.

La **propagation** permet, à partir de règles d'inférence déterministes, d'exclure certaines des valeurs de variables qui sont incompatibles avec les contraintes du CSP. Ces règles d'inférence sont «peu coûteuses» (c'est-à-dire de complexité polynomiale) et permettent d'éliminer rapidement certaines valeurs impossibles. Le revers de la médaille est qu'elles sont également incomplètes, c'est-à-dire qu'elles ne permettent pas de déduire tout ce qui peut être déduit à partir des données et de la sémantique des contraintes ensemblistes. Afin de préserver la complétude, Oz alterne la propagation avec des étapes de **distribution**, c'est-à-dire des choix non déterministes : lorsque aucune inférence n'est possible et que l'espace de calcul[51] n'est ni un échec (le stock de contraintes contient la valeur $\perp$) ni une réussite (toutes les variables sont déterminées, c'est-à-dire ont une valeur unique), alors une distribution prend place créant à partir d'un espace de calcul E deux espaces de calcul $E+\beta$ et $E+\neg\beta$ avec β une contrainte basique, c'est-à-dire une relation entre variable et valeurs et $\neg\beta$ sa négation.

Si la distribution détermine la forme de l'espace de recherche, la **stratégie de recherche** détermine la façon dont cette espace sera traversé. Oz met à la disposition de l'utilisateur plusieurs stratégies possibles dont en particulier une stratégie de recherche en profondeur (depth-first) et une stratégie de recherche de la meilleure solution (d'après un critère de qualité donné). Oz permet également de choisir entre une recherche pour une solution unique ou pour toutes les solutions.

8.2 Les contraintes ensemblistes de Oz/Mozart

Oz/Mozart permet l'utilisation de contraintes ensemblistes pour des variables portant sur des ensembles d'entiers naturels. Ces variables ensemblistes sont approximées par des treillis encapsulant l'information connue sur la borne inférieure et supérieure des ensembles dénotés par les variables, ainsi que sur la borne inférieure et supérieure de leur cardinalité.

Une **contrainte ensembliste de base** B est soit une contrainte sur la borne inférieure et supérieure de la cardinalité et de l'ensemble dénoté par une variable ensembliste, soit la conjonction de deux contraintes de base, soit le booléen «faux» (d_i sont des ensembles d'entiers, S est une variable ensembliste et n_i un entier) :

$$\begin{array}{lcll} \mathrm{B} & = & n_1 \leq |S| \wedge |S| \leq n_2 \wedge d_1 \subseteq S \wedge s \subseteq d_2 & \text{(i)} \\ & | & B \wedge B & \text{(ii)} \\ & | & \perp & \text{(iii)} \end{array}$$

Une **contrainte ensembliste complexe** C est soit une contrainte de base (iv), soit une conjonction de contraintes complexes (v), soit une contrainte de la forme (vi) ou de la forme (vii) :

$$\begin{array}{lcll} \mathrm{C} & = & B & \text{(iv)} \\ & | & C \wedge C & \text{(v)} \\ & | & S_1 \supseteq S_2 \cap S_3 & \text{(vi)} \\ & | & S_1 \subseteq S_2 \cup S_3 & \text{(vii)} \end{array}$$

Une contrainte ensembliste complexe peut également faire intervenir une variable sur les entiers I comme suit :

$$\begin{array}{lcll} \mathrm{C} & = & I \in d & \text{(viii)} \\ & | & I = |S| & \text{(ix)} \\ & | & I \in S & \text{(x)} \end{array}$$

Comme l'indique le tableau 8.1, toutes les opérations ensemblistes peuvent être exprimées en Oz à partir des contraintes ensemblistes complexes et basiques permises par le langage ($\mathcal{U}$ est l'ensemble fini d'entier $\mathcal{U} = \{e | 0 \leq e \leq 2^{28} - 2\}$).

[51] Dans Oz, un espace de calcul comprend un ensemble de contraintes de base (le «stock de contraintes») et un ensemble de propagateurs. Un propagateur est une contrainte non basique implantée par un agent concurrent réagissant aux changements intervenant dans le stock de contraintes.

Inclusion	$S_1 \subseteq S_2$	$\equiv$	$S_1 \subseteq (S_2 \cup S_3) \wedge S_3 \subseteq \emptyset$
Séparation	$S_1 \| S_2$	$\equiv$	$S_3 \supseteq (S_1 \cap S_2) \wedge S_3 \subseteq \emptyset$
Union	$S_1 = S_2 \cup S_3$	$\equiv$	$S_1 \subseteq (S_2 \cup S_3) \wedge S_2 \subseteq S_1 \wedge S_3 \subseteq S_1$
Complément	$S_1 = S_2^C$	$\equiv$	$S_3 \subseteq \emptyset \wedge (S_1 \cap S_2) \subseteq S_3 \wedge \mathcal{U} \subseteq S_4 \wedge S_4 \subseteq (S_1 \cup S_2)$
Intersection	$S_1 = S_2 \cap S_3$	$\equiv$	$S_1 \supseteq (S_2 \cap S_3) \wedge S_1 \subseteq S_2 \wedge S_1 \subseteq S_3$
Différence	$S_1 = S_2 \setminus S_3$	$\equiv$	$S_1 \subseteq (S_2 \cap S_3^C) \wedge (S_2 \cap S_3^C) \subseteq S_1$
Partition	$S_1 = S_2 \uplus S_3$	$\equiv$	$S_1 = (S_2 \cup S_3) \wedge S_2 \| S_3$

TAB. 8.1 – Opérations ensemblistes

8.3 Contraintes ensemblistes et Propagation

Comme nous l'avons déjà mentionné (cf. section 8.1), la propagation permet, à partir de règles d'inférence déterministes, d'exclure certaines des valeurs de variables qui sont incompatibles avec les contraintes du CSP. Ainsi par exemple, les règles d'inférence (ou «règles de propagation») utilisées par Oz pour les contraintes de base sont les suivantes.

Propagation sur les bornes. Si une contrainte de base implique que la borne supérieure d'une variable est incluse dans sa borne inférieure, il y a échec :

$$\frac{d_1 \subseteq S \wedge S \subseteq d_2}{\perp} \text{ si } d_1 \not\subseteq d_2 \tag{8.1}$$

La propagation de contraintes permet d'ajouter des éléments à une borne inférieure (règle 8.2) ainsi que de retirer des éléments d'une borne supérieure (règle 8.3).

$$\frac{d_1 \subseteq S \wedge d_2 \subseteq S}{d_1 \cup d_2 \subseteq S} \tag{8.2}$$

$$\frac{S \subseteq d_1 \wedge S \subseteq d_2}{S \subseteq d_1 \cap d_2} \tag{8.3}$$

Propagation sur la cardinalité. Lorsqu'une contrainte de base implique que la borne supérieure de la cardinalité d'une variable est inférieure à la borne inférieure de sa cardinalité, il y a échec :

$$\frac{n_1 \leq |S| \wedge |S| \leq n_2}{\perp} \text{ si } n_1 > n_2 \tag{8.4}$$

Propagation sur bornes et cardinalité. Ces règles permettent de déterminer la valeur d'une variable ensembliste. Ainsi si la cardinalité de la borne inférieure d_1 d'un ensemble S est égale à la borne supérieure de la cardinalité de cet ensemble, S est identifié avec sa borne inférieure d_1 (règle 8.5). La règle 8.6 est une règle miroir pour les bornes supérieures.

$$\frac{d_1 \subseteq S \wedge |S| \leq n}{S \subseteq d_1} \text{ si } |d_1| = n \tag{8.5}$$

$$\frac{S \subseteq d_2 \wedge |S| \geq n}{d_2 \subseteq S} \text{ si } |d_2| = n \tag{8.6}$$

Les deux dernières règles servent à maintenir l'invariant selon lequel pour la contrainte de base $d_1 \subseteq S \wedge S \subseteq d_2 \wedge n_1 \leq |S| \wedge |S| \leq n_2$, $|d_1| \leq n_1 \leq n_2 \leq |d_2|$.

$$\frac{d \subseteq S}{|d| \leq |S|} \tag{8.7}$$

$$\frac{d \subseteq S}{|S| \leq |d|} \tag{8.8}$$

Pour les contraintes complexes, d'autres règles de propagation sont appliquées. Nous renvoyons le lecteur intéressé à la thèse de Tobias Müeller [142] pour plus de détails et d'explication.

Ceci conclut notre brève introduction à la programmation par contraintes. Dans les chapitres qui suivent, nous montrons comment les contraintes ensemblistes implantées dans Oz/Mozart peuvent être utilisées en linguistique informatique.

Chapitre 9

Descriptions d'arbres

L'arbre est une structure très utilisée en linguistique. On l'utilise par exemple, pour représenter la structure syntaxique d'une expression ou, par le biais des formules logiques, pour représenter le sens des expressions de la langue naturelle.

Pour des raisons d'ordre informatique ou linguistique, il arrive que le linguiste veuille manipuler au lieu d'un arbre, une description d'arbre. Ainsi par exemple, certaines insuffisances des grammaires d'arbres adjoints ont donné lieu à l'introduction des grammaires de descriptions d'arbres (Description Tree Grammars, [158]), c'est-à-dire de grammaires qui utilisent des descriptions d'arbres comme unités lexicales élémentaires plutôt que des arbres comme c'est le cas dans les grammaires d'arbres adjoints. De même, dans le domaine de la sémantique, on a proposé d'utiliser les descriptions de formules plutôt que des formules afin de pouvoir sous-spécifier certaines relations sémantiques telle que par exemple, la relation de portée [143].

Néanmoins au final, la structure pertinente reste l'arbre : la structure syntaxique d'une phrase est un arbre et non une description d'arbre tandis que sa représentation sémantique est une formule logique (un arbre) et non une description de formules logiques. Il est donc nécessaire de pouvoir produire, à partir d'une description d'arbres, l'ensemble des arbres (e.g., structure syntaxique, formule logique) satisfaisant cette description.

Dans ce chapitre, nous commençons par montrer que cette tâche peut être formulée comme une tâche d'inférence ensembliste. Nous montrons ensuite comment elle peut être mise en oeuvre pour servir différentes applications linguistiques et en particulier pour produire les représentations sémantiques (formules logiques) décrites par une représentation sous-spécifiée et pour produire les arbres syntaxiques associés à une séquence de mots ou à une représentation sémantique par une grammaire basée sur des arbres sous-spécifiés.

9.1 La génération d'arbres comme tâche d'inférence ensembliste

La formulation ensembliste de la tâche de génération d'arbres part d'une approche basée sur les logiques d'arbres [143, 16]. Nous commençons donc par résumer cette approche (section 9.1.1). Nous montrons ensuite comment elle peut être implantée dans une perspective ensembliste (9.1.2).

9.1.1 Des descriptions aux objets

Les logiques d'arbres [143, 16] sont des logiques où les modèles sont des arbres (finis) et les formules des descriptions d'arbres. En utilisant ces logiques, on peut donc formuler le problème de la production des arbres satisfaisant une description comme un problème de génération de modèles :

> Etant donné une logique d'arbre $\mathcal{L}_\mathcal{A}$ et une formule ϕ de cette logique, le problème de la production des arbres satisfaisant une description ϕ se réduit à générer les modèles $\mathcal{T}$ tels que $\mathcal{T} \models \phi$ dans $\mathcal{L}_\mathcal{A}$.

Comme il existe une quantité infinie de modèles satisfaisant une formule donnée, on a généralement recours à la notion de minimalité pour restreindre l'ensemble des solutions à un ensemble fini. Le générateur de modèles produira ainsi non pas, tous les modèles, mais tous les modèles minimaux satisfaisant la formule considérée. Dans les applications qui nous concernent, les modèles minimaux considérés sont les formes résolues des descriptions (voir plus bas).

Syntaxe. La logique d'arbres $\mathcal{L}^\tau$ que nous utilisons a la syntaxe suivante.

$$\phi \quad ::= \quad x\,R\,y \ \mid \ x{:}f(x_1,\ldots,x_n) \ \mid \ \phi \wedge \phi'$$

où x, y, x_i portent sur un ensemble infini de variables de noeuds, f porte sur une signature finie Σ, et R porte sur des sous-ensembles arbitraires des symboles relationnels $\{=, \lhd^+, \rhd^+, \perp\}$. Le symbole $\lhd^+$ dénote la dominance stricte et $\perp$ représente la disjonction structurelle ($x \perp y$ ssi x et y ne sont reliés ni par l'égalité ni par la dominance. Dans un littéral structurel, c'est-à-dire un littéral de la forme $x\,R\,y$, R est un opérateur ensembliste interprété comme une forme restreinte de disjonction : une des relations dans R doit être vérifiée entre les noeuds dénotés par x and et y. Par exemple, $x\,\{=,\perp\}\,y$ est satisfait soit si les noeuds dénotés par x et y sont égaux, soit s'ils sont inclus dans des sous-arbres disjoints.

Sémantique. L'interprétation de $\mathcal{L}^\tau$ est donnée par une fonction d'interprétation sur des arbres finis.

Un **noeud** dans un arbre est identifié avec le chemin menant de la racine au noeud. Un chemin π est une séquence d'entiers positifs. Nous écrivons ϵ pour le chemin vide et $\pi_1\pi_2$ pour la concaténation de π_1 et π_2.

S'il existe $\pi'' \neq \epsilon$ tel que $\pi = \pi'\pi''$, alors $\pi' \lhd^+ \pi$.

S'il existe des chemins π, π'_1, π'_2 et des entiers $i \neq j$ tels que $\pi_1 = \pi i \pi'_1$ et $\pi_2 = \pi j \pi'_2$, alors π_1 et π_2 sont disjoints : $\pi_1 \perp \pi_2$.

Un **domaine d'arbre** est un ensemble non vide de chemins fermé par les préfixes. Un **arbre fini** τ portant sur la signature Σ est un triplet de la forme (D_τ, L_τ, A_τ) avec D_τ un domaine d'arbre, A_τ une fonction d'arité $A_\tau : D_\tau \to N$ et L_τ une fonction d'étiquettage $L_\tau : D_\tau \to \Sigma$ telle que pour tout chemin $\pi \in D_\tau, \pi i \in D_\tau$ ssi $1 \leq i \leq A_\tau(\pi)$.

V_ϕ dénote l'ensemble des variables apparaissant dans ϕ. Un **modèle** pour une formule ϕ de $\mathcal{L}^\tau$ est une paire (τ, α) où $\tau = (D_\tau, L_\tau, A_\tau)$ est un arbre fini et $\alpha : V_\phi \to D_\tau$ assigne à chaque variable de ϕ un noeud dans τ. Nous écrivons $(\tau, \alpha) \models \phi$ pour la relation de satisfaction et définissons cette relation comme suit :

$$\begin{array}{lll}
(\tau,\alpha) \models \phi \wedge \phi' & \text{si} & (\tau,\alpha) \models \phi \text{ and } (\tau,\alpha) \models \phi' \\
(\tau,\alpha) \models x\,R\,y & \text{si} & \alpha(x)\,r\,\alpha(y) \quad \text{pour un } r \in R \\
(\tau,\alpha) \models x{:}f(x_1,\ldots,x_n) & \text{si} & L_\tau(\alpha(x)) = f \quad \wedge \\
 & & A_\tau(\alpha(x)) = n \quad \wedge \\
 & & \alpha(x)i = \alpha(x_i) \quad \text{pour } 1 \leq i \leq n
\end{array}$$

[117] montre que la satisfaisabilité de cette logique est NP-difficile.

Comme nous l'avons déjà mentionné, il existe généralement un ensemble infini de modèles satisfaisant une formule donnée. Une forme résolue contient cependant toute l'information nécessaire pour reconstruire cet ensemble infini. Une **forme résolue** est un arbre contenant outre les liens de dominances immédiates (liens I) classiques, des liens de dominance (liens D). Les noeuds d'une forme résolue sont soit ouverts soit fermés et doivent obéir aux contraintes suivantes :

- un noeud fermé a uniquement des liens I. Son arité est fixe.
- un noeud ouvert a uniquement des liens D. Son arité est inconnue.

Formellement (cf. [47]), un forme résolue est une saturation (par le système inférentiel défini par [48]) cohérente d'une description qui garantit que la description initiale est satisfiable. Intuitivement, c'est l'objet minimal satisfaisant la description.

9.1.2 Génération de modèles et contraintes ensemblistes

Rappelons que la tâche qui nous intéresse est celle qui consiste à générer les arbres satisfaisant une description et plus particulièrement ses formes résolues. Nous montrons maintenant que cette tâche peut être formulée comme un problème de satisfaction de contraintes ensemblistes.

L'idée de base est que, pour toute paire de noeuds n_1, n_2 apparaissant dans un arbre, la relation structurelle entre ces deux noeuds est soit la dominance (n_1 domine n_2 ou n_2 domine n_1), soit l'identité ($n_1 = n_2$), soit la séparation (n_1 et n_2 sont inclus dans des sous arbres disjoints). Pour chaque paire de noeuds, ces relations sont mutuellement exclusives : seule une de ces relations peut être vraie. Aussi lorsque pour toutes les paires de noeuds dénotant les variables apparaissant dans la description considérée, la relation structurelle vraie de cette paire de noeud est déterminée, alors la structure de l'arbre est complètement déterminée.

L'inférence ensembliste intervient par le biais des relations structurelles dont elle détermine l'extension. Pour chaque variable de noeud x apparaissant dans la description donnée en entrée et pour chaque relation structurelle $\rho \in \{=, \lhd^+, \rhd^+, \perp\}$, l'inférence ensembliste vise à déterminer la valeur :

$$E^x_\rho = \{y \mid x\rho y\}$$

Les variables sur lesquelles porte le CSP représentent ces extensions. Ainsi, pour chaque variable de noeud x apparaissant dans la description considérée, une structure de traits N_x est créée qui est de la forme :

$$\begin{bmatrix} \textbf{eq} & : & E^x_= \\ \textbf{down} & : & E^x_{\lhd^+} \\ \textbf{up} & : & E^x_{\rhd^+} \\ \textbf{disjoint} & : & E^x_\perp \end{bmatrix}$$

où $E^x_=, E^x_{\lhd^+}, E^x_{\rhd^+}, E^x_\perp$ sont des variables ensemblistes portant sur l'ensemble $\mathcal{V}$ des variables apparaîssant dans la description considérée et représentant respectivement :

- $E^x_=$: l'ensemble des variables dont la dénotation est égale à la dénotation de la variable x ;
- $E^x_{\lhd^+}$: l'ensemble des variables dont la dénotation est dominée par la dénotation de la variable x ;
- $E^x_{\rhd^+}$: l'ensemble des variables dont la dénotation domine la dénotation de la variable x ;
- $E^x_\perp$: l'ensemble des variables dont la dénotation est disjointe de la dénotation de la variable x.

Etant donné cette représentation initiale, les contraintes auront pour rôle de déterminer pour chaque structure de traits N_x, la valeur des variables ensemblistes $E^x_=, E^x_{\lhd^+}, E^x_{\rhd^+}, E^x_\perp$. Lorsque pour une variable, la valeur de ces quatre variables est déterminée, la position dans l'arbre du noeud dénoté par cette variable est connue. Lorsque pour toutes les variables apparaissant dans la description donnée en entrée, la valeur de ces variables est déterminée, la structure de l'arbre est connue.

Deux types de contraintes contribuent à déterminer la valeur de ces variables : les contraintes provenant de la description donnée en entrée d'une part, et les contraintes de bonne formation des arbres d'autre part. Ces contraintes logiques sont traduites en contraintes sur les variables ensemblistes $E^{x_i}_=, E^{x_i}_{\lhd^+}, E^{x_i}_{\rhd^+}, E^{x_i}_\perp$ contribuant ainsi à déterminer leur valeur. Ainsi par exemple, la contrainte de dominance $x \lhd^+ y$ est traduite par la contrainte $N^x \lhd^+ N^y$ telle que :

$$N^x \lhd^+ N^y \quad \equiv \quad \begin{cases} E^x_{\lhd^+} \supseteq E^y_{\lhd^+} \cup E^y_= \\ E^x_{\rhd^+} \cup E^x_= \subseteq E^y_{\rhd^+} \\ E^x_\perp \subseteq E^y_\perp \end{cases}$$

L'ensemble des contraintes utilisées pour déterminer la valeur des variables ensemblistes intervenant dans la représentation des descriptions d'arbres est détaillée dans [46] qui montre également comment la résolution du problème de satisfaction de contraintes ainsi posé, peut être optimisé en Oz par l'utilisation d'une disjonction concurrente faisant intervenir des variables de choix.

Pour résumer : les contraintes ensemblistes et l'interprétation procédurale qui leur est donnée en Oz, permettent de construire, à partir d'une formule des logiques d'arbres, une représentation des formes résolues satisfaisant cette formule.

9.2 Discours : incrémentalité, ambiguïté et sous-spécification

Ayant brièvement décrit les techniques de base permettant de générer les formes résolues d'une description donnée, nous en examinons maintenant diverses applications au traitement de la langue naturelle. Nous commençons par la construction incrémentale d'une représentation sémantique compositionnelle du discours et de ses ambiguïtés.

L'ambiguïté est une caractéristique fondamentale de la langue naturelle qui se manifeste à tous les niveaux et en particulier : au niveau lexical (le même mot peut avoir plusieurs sens et appartenir à différentes catégories syntaxiques, cf. exemple 9.1) ; au niveau syntaxique (une même expression peut avoir plusieurs structures syntaxiques possibles, cf. exemple 9.2) et au niveau sémantique (une même expression peut avoir plusieurs interprétations possibles, cf. exemple 9.3).

(9.1) La belle ferme la **porte**. *porte : Nom/Verbe*

(9.2) L'homme regarde la femme avec des jumelles
$[_P\ [_{GN}$ L'homme $[[_{GV}\ [_V$ regarde$]\ [_{GN}$ la femme$]]\ [_{GP}$ avec des jumelles$]]]]$
$[_P\ [_{GN}$ L'homme $[_{GV}\ [_V$ regarde$]\ [[_{GN}$ la femme$]\ [_{GP}$ avec des jumelles$]]]]]$

(9.3) Tout yogi a un guru.
$\forall x.(yogi(x) \rightarrow \exists y.(guru(y) \wedge has(x, y)))$
$\exists y.(guru(y) \wedge \forall x.(yogi(x) \rightarrow has(x, y)))$

Lors d'une analyse automatique non incrémentale, ces ambiguïtés se combinent entre elles multipliant ainsi le nombre d'analyses possibles pour une même phrase. Un système de préférence est alors nécessaire pour déterminer l'analyse la plus probable.

Par ailleurs, les études psycholinguistiques et, en particulier, l'identification d'«effets de labyrinthe» (*garden path effects*, voir plus bas) ont montré que le processeur humain fonctionne de façon incrémentale : si les «phrases labyrinthes» induisent une erreur puis une modification de l'interprétation assignée dans un premier temps au début de la phrase, c'est nécessairement qu'il y a eu une telle interprétation et donc que le processeur humain permet de construire une interprétation au fur et à mesure du discours.

Afin de modéliser ces effets de labyrinthe et l'incrémentalité du processeur humain, divers analyseurs ont été proposés ([52, 141, 179, 155]) qui ont en commun d'être à la fois incrémentaux et quasi-déterministes. Ces analyseurs traitent une phrase mot par mot et de gauche à droite : pour chaque nouveau mot rencontré, les différents rattachements syntaxiques (et parfois sémantiques) de ce mot sont considérés, et un choix unique est fait. Ce quasi-déterminisme permet de prédire les effets de labyrinthe. Il permet aussi un traitement local et «économique» de l'ambiguïté au sens où les ambiguïtés du nouveau mot ne se combinent qu'avec l'analyse unique construite pour la séquence initial (la multiplication des ambiguïtés des différents éléments de la phrase entre elles est évitée). Enfin il est compatible avec un traitement préférentiel des ambiguïtés globales.

Dans ce qui suit, nous commençons par montrer que les différents types d'ambiguïté sémantique identifiés au niveau de la phrase se retrouvent également au niveau du discours induisant des effets de labyrinthe similaires à ceux observés au niveau de la phrase. Nous montrons ensuite comment les descriptions d'arbres peuvent être utilisées pour représenter ces ambiguïtés. Enfin nous montrons comment la notion de «formes résolues» combinée avec un critère de choix permet à la fois : (i) de déterminer l'interprétation la plus probable à chaque étape d'une analyse incrémentale et (ii) de modéliser les effets de labyrinthe.

9.2.1 Ambiguïtés discursives

[141] montre qu'au niveau de la phrase, l'utilisation de descriptions d'arbres, plutôt que d'arbres, permet (i) un traitement monotone des ambiguïtés locales, c'est-à-dire des ambiguïtés qui ne se répercutent pas au niveau de l'interprétation finale ; (ii) un traitement déterministe des ambiguïtés globales ; et (iii) une différentiation entre ambiguïtés locales «simples» et ambiguïtés locales induisant un labyrinthe.

Dans [69], nous montrons que ces différents types d'ambiguïtés, bien qu'identifiés au niveau syntaxique de la phrase, se retrouvent également au niveau de la sémantique compositionnelle du discours.

Ambiguïtés locales simples (ou ambiguïtés de rattachement)

Les ambiguïtés dites d'attachement (ou de rattachement) sont liées à l'incrémentalité : à un certain point pendant l'analyse, l'unité à traiter X est un sous-constituant d'un noeud N de la structure déjà construite, mais on ne peut déterminer si X est une fille directe ou indirecte de N. Au niveau syntaxique, l'exemple classique est :

(9.4) a. I saw my aunt.

b. I saw my aunt's cat

c. I saw my aunt's cat's litter box

Dans les trois cas, le groupe nominal *my aunt* est un sous-constituant du syntagme nominal objet de *saw*. Structurellement, le sous-arbre représentant la contribution syntaxique de *my aunt* est dans tous les cas un sous-arbre du noeud représentant l'argument objet du verbe.

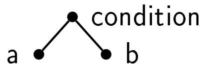
Au niveau de la sémantique du discours, l'ambiguïté locale simple peut être illustrée par l'exemple suivant :

(9.5) a. (a) Jean et Marie ne vont au cinéma que lorsqu' (b) on y donne un film irlandais *condition*(e_a, e_b)

c1. aussi (c1) y vont-ils peu souvent. *cause(condition*(e_a, e_b)*)*, e_{c_1}*)*

c2. ou quand (c2) le film a bonne presse. *condition*$(e_a,$ *ou*$(e_b, e_{c_2}))$

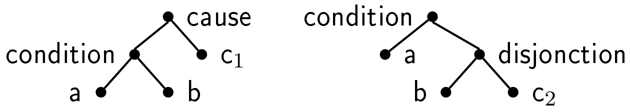
Dans le premier segment de ce discours, la conjonction *lorsque* exprime une relation de condition qui prend pour arguments les éventualités dénotées par les phrases (a) et (b). Nous représentons cette relation par la formule *condition*(e_a, e_b) et, de façon équivalente, par la structure arborescente suivante :



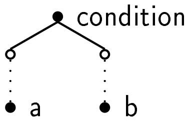
(9.6)

Selon la suite du discours, cette interprétation est soit confirmée (9.5c1), soit révisée (9.5c2).

Continuation (9.5c1) Continuation (9.5c2)



Dans aucun des cas cependant, un processus de réparation n'est perceptible. En d'autres termes, le type de révision induite par les ambiguïtés de rattachement ne semble pas augmenter la charge de traitement et est donc sans doute mieux modélisé par un processus monotone. L'utilisation de descriptions plutôt que d'arbres permet une modélisation de ce type. En effet supposons que l'interprétation de (9.5a) soit représentée non par l'arbre (9.6) mais par la description (9.7)[52] :



(9.7)

Comme l'illustre les représentations «Continuation (9.5c1)» et «Continuation (9.5c2)» ci-dessus, cette description est compatible avec les deux types de continuation possibles (confirmation et révision). La description initiale (9.7) rend ainsi compte de l'ambiguïté locale de rattachement introduite par (9.5a-b) sans introduire de non-déterminisme. L'hypothèse déterministe de Marcus est ainsi satisfaite : les ambiguïtés locales de rattachement qui ne correspondent pas à une réanalyse consciente sont traitées de façon déterministe.

[52] La notation logique devenant rapidement obscure, nous utilisons des images graphiques plutôt que des formules logiques pour représenter les descriptions d'arbres. Les conventions suivantes sont adoptées. Les noeuds représentent des variables de noeuds, les traits pleins la dominance stricte et les traits pointillés, la dominance. Les étiquettes sur les noeuds représentent des structures de traits par exemple, NP :x représente la structure de traits $\{cat{:}np, idx{:}x\}$. Dans les formes résolues, un noeud ouvert est représenté par un cercle vide et un noeud fermé par un cercle plein.

Ambiguïtés locales «labyrinthe»

Comme nous venons de le voir, une *ambiguïté locale simple* sera associée avec une sous-spécification de la dominance, que ce soit au niveau de la phrase ou du discours : dans les deux cas, il s'agit de déterminer le niveau d'enchâssement du nouvel élément. Cet élément est-il dominé directement ou indirectement par un noeud déjà existant ?

Par contraste, l'ambiguïté locale induisant un effet de labyrinthe fera intervenir une description reflétant le fait que le nouvel élément pourrait être dominé soit par la fille directe gauche d'un noeud déjà existant, soit par sa fille droite (cf. Figure 9.1).

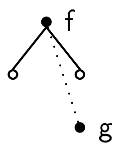


Fig. 9.1 – Forme des descriptions utilisés en cas d'ambiguïté locale labyrinthe

Ainsi dans l'exemple (9.8) ci-dessous, le noeud associé avec le verbe *raced» sera dominé soit par le noeud racine du syntagme nominal sujet the horse raced past the barn (exemple 9.8b), soit par le noeud racine du syntagme verbal " raced past the barn* (exemple 9.8a).

(9.8) a. The horse raced past the barn.

b. The horse raced past the barn fell.

Un exemple d'ambiguïté locale discursive induisant un labyrinthe est donné par le discours suivant :

(9.9) Because (a) Jon is easily upset, whenever (b) he flies, (c) he gets very nervous.

(9.10) Because (a) Jon is easily upset, whenever (b) he flies, (c) he gets very nervous for example, (d) he should practice yoga.

On observe une forte différence entre (9.9) and (9.10) : alors que (9.9) est facile à interpréter, (9.10) est beaucoup plus difficile et semble induire un effet de labyrinthe nécessitant une réanalyse des structures construites.

Le type de descriptions utilisées ci-dessus permet de rendre compte de cette différence. En effet, comme pour l'exemple syntaxique (9.8), l'ambiguïté est une ambiguïté sur la domination du nouvel élément par la fille soit gauche, soit droite d'un noeud déjà présent dans la structure construite. Spécifiquement, soit le segment (b-c) fournit le deuxième argument de la conjonction *because*, soit il modifie (élabore) le sens de (a). Dans le premier cas, le sous-arbre représentant le sens du segment (b-c) est dominé par la branche droite de la relation causale introduite par *because* alors que dans le second cas, il sera dominé par sa fille immédiate gauche. Si le critère de choix utilisé pour préserver le déterminisme est par exemple, la saturation des foncteurs par leurs arguments, la préférence ira à l'interprétation correspondant à l'exemple (9.9) – le segment (b-c) fournit le deuxième argument de la conjonction *because* – et le discours devra être réinterprété si la continuation est celle donnée en (9.10).

Ambiguïté globale

Enfin comme la phrase, le discours présente des ambiguïtés sémantiques globales qui peuvent être traitées de façon déterministe en utilisant la sous-spécification.

Un premier cas d'ambiguïté de ce type sont les ambiguïtés de portée illustrées par le discours suivant :

(9.11) a. Sarah reads a novel

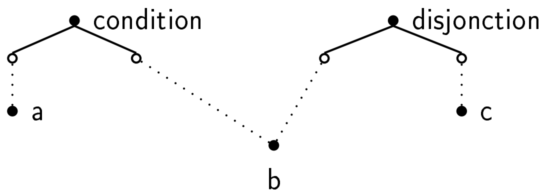
b. if she is bored

c. or she is unhappy.

Deux interprétations sont possibles :

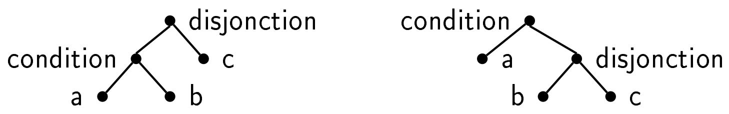
(9.12) *Sarah lit un roman quand elle s'ennuie ou quand elle est malheureuse*
Sarah lit un roman quand elle s'ennuie ; sinon elle est malheureuse

Ce type d'ambiguïté est capturé comme dans [143] par une sous-spécification des relations structurelles entre opérateurs de portée telle que la description résultante est satisfaite par exactement les deux structures représentant les interprétations possibles. Ainsi par exemple, la description associée à l'exemple (9.11) est :



(9.13)

Description qui est satisfaite par les deux structures ;



Un autre type d'ambiguïté globale est celui des ambiguïtés entre conjonction et subordination qui peut être illustré par l'exemple suivant :

(9.14) a. Jon didn't come to work

b. because the trains aren't running.

c. Buses aren't either.

Comme il l'est montré dans [15], ce type d'ambiguïté peut être capturé par le même type de descriptions que celles utilisées pour les ambiguïtés de portée si les modèles satisfaisant ces descriptions peuvent être non seulement des arbres mais également des graphes dirigés acycliques.

9.2.2 Analyse incrémentale du discours

Nous venons de voir que les trois grands types d'ambiguïté survenant dans une analyse incrémentale (de la phrase ou du discours) peuvent être sous-spécifiés par le biais de descriptions d'arbre. Si P, P_1, P_2 sont des noeuds présents dans la structure déjà construite et N est le nouveau noeud à inclure dans cette structure, alors les descriptions associées avec les différents types d'ambiguïtés peuvent être décrits comme suit :

Ambiguïté locale simple : P domine N et l'ambiguïté porte sur la distance entre P et N. N peut être soit une fille directe, soit une fille indirecte de P.

Ambiguïté globale : P_1, P_2 dominent N et l'ambiguïté porte sur la relation de domination entre P_1 et P_2 : P_1 domine P_2 ; ou P_2 domine P_1 ; ou P_1 et P_2 ne sont pas dans une relation de domination et la représentation sémantique est un graphe (non un arbre).

Ambiguïté locale à effet de labyrinthe : P domine N et l'ambiguïté porte sur la domination de N soit par la fille immédiate gauche de P, soit par sa fille immédiate droite.

Dans ce qui suit, nous ébauchons un système d'analyse qui permet à partir de telles descriptions, de modéliser d'une part, un traitement monotone des ambiguïtés locales et globales et d'autre part, un traitement non monotone des discours induisant un effet de labyrinthe. Ce système peut être résumé comme suit :

Pour chaque nouvelle unité discursive de base (conjonction ou clause) :
– la description courante est incrémentée sur la base des informations syntaxiques et sémantiques disponibles
– les formes résolues de la description ainsi produite sont calculées
– le critère de préférence (saturation des arguments) est appliqué pour déterminer la forme résolue préférentielle
– cette forme résolue préférentielle forme la nouvelle description courante et est utilisée pour la suite de l'analyse

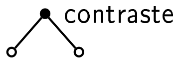
Le critère de préférence «Saturation des Arguments» est similaire au principe incrémental de Gorrell (Incremental Licensing Principle, [75]) ou à la préférence pour la substitution plutôt que l'adjonction de [187] : il privilégie les formes résolues réalisant le plus grand nombre d'arguments.

Analyse des Ambiguïtés locales simples.

Pour ce premier type d'ambiguïté, nous montrons par un exemple que l'analyse se fait de manière monotone et déterministe.

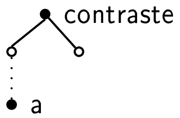
(9.15) On the one hand (a) Jon is content.
On the other hand (b) Mary isn't.

Ainsi étant donné l'exemple (9.15) ci-dessus, les étapes de l'analyse peuvent être résumées de la façon suivante. Dans un premier temps, la conjonction *on the one hand* est traitée dont la représentation sémantique indique une relation de contraste entre deux propositions (encore inconnues) :



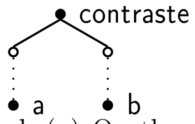
On the one hand,

Le deuxième élément discursif de base est donné par la clause *Jon is content* qui, du fait de sa contiguïté avec la conjonction *on the one hand* doit faire partie de son premier argument syntaxique et sémantique :



On the one hand, (a)

Dans un troisième temps, *on the other hand* est traité qui complète la première conjonction *on the one hand* mais n'apporte pas de contribution sémantique nouvelle. Enfin la deuxième clause est traitée qui par le même raisonnement que pour la première doit faire partie du deuxième argument sémantique de la relation de contraste introduite par la conjonction discontinue *on the one hand ... on the other hand*. A ce stade du traitement incrémental, la représentation du texte (9.15) est donc :

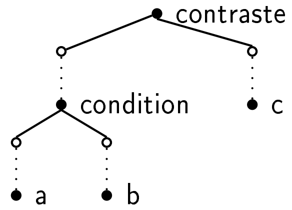


On the one hand, (a) On the other hand (b).

Considérons maintenant l'exemple (9.16) où, lors du traitement incrémental, la relation entre le segment (a) et la relation de contraste exprimé par *one the one hand ... on the other hand* est révisée pour devenir une relation entre (a) et la relation de condition exprimée par *if*.

(9.16) On the one hand (a) Jon is content if (b) he can read a novel.
On the other hand (c) he is too poor to buy books.

Dans ce cas, l'analyse procède de façon identique pour les deux premières étapes (analyse de *on the one hand, Jon is content*). Lorsque la conjonction *if* est traitée en revanche, la description est incrémentée comme indiquée en Figure 9.2 : comme *if* apparaît immédiatement à la droite du segment (a), (a) doit faire partie de l'argument syntaxique et sémantique gauche de *if*[53]. Par un raisonnement similaire, le segment (b) doit contribuer au deuxième argument de la relation de condition introduite par *if*. Enfin, lorsque la deuxième partie de la conjonction *on the one hand ... on the other hand* est traitée, la description peut être contrainte pour inclure le segment *a if b* dans le premier argument de la relation de contraste introduite par cette conjonction multiple.



On the one hand (a) if (b). On the other hand (c).

FIG. 9.2 – Analyse de l'exemple (9.16)

Dans les deux cas, malgré la présence de l'ambiguïté locale, l'analyse reste monotone et déterministe (il n'y a qu'une seule forme résolue possible à chaque étape).

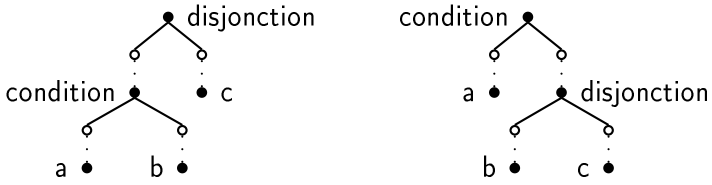
Analyse des Ambiguïtés globales

Alors que l'analyse d'une ambiguïté locale simple fait intervenir une forme résolue unique à chaque étape, une ambiguïté globale en fera intervenir plusieurs rendant ainsi l'analyse non-déterministe (ce qui est justifié : il y aura autant d'analyses qu'il y a d'interprétations possibles)[54].

[53] Afin de simplifier l'exposition, nous ignorons ici l'ambiguïté préfixe/infixe de *if*.

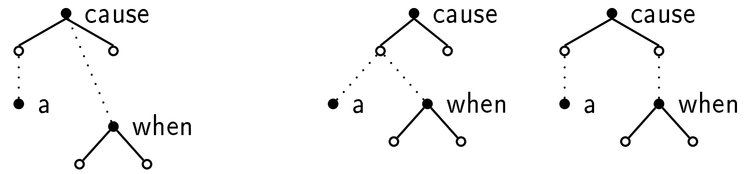
[54] Bien sûr, l'analyse pourrait être rendue déterministe par l'utilisation d'un système de préférence. En pratique cependant, il est extrêmement difficile de déterminer des préférences pour ce type d'ambiguïté en particulier pour les ambiguïtés de portée. En outre, même si ces préférences pouvaient être identifiées, leur application ferait intervenir des facteurs pragmatiques difficiles à modéliser au plan informatique (connaissances du monde, de l'interlocuteur, etc.).

Ainsi par exemple, la description (9.13) associée à l'exemple (9.11) ci-dessus a deux formes résolues :



Analyse des ambiguïtés locales à effet de labyrinthe

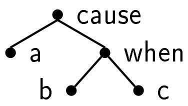
Enfin l'analyse d'une ambiguïté locale à effet de labyrinthe fait intervenir deux formes résolues dont l'une est privilégiée par le principe de saturation des arguments. Si la forme résolue choisie à partir de ce principe n'est pas compatible avec la suite du discours, il y a échec et un retour arrière est nécessaire pour «défaire» le choix fait et ré-analyser le discours. Ainsi, après analyse du segment *Because (a), whenever* de l'exemple (9.9), la description obtenue est celle donnée en (9.3), qui a les deux formes résolues indiquées.



Because (a), whenever Forme résolue 1 Forme résolue 2

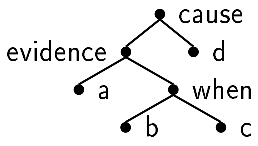
FIG. 9.3 – Analyse de l'exemple (9.9)

Le principe de saturation des arguments privilégie la forme résolue 2 qui est compatible avec la suite de l'exemple (9.9) dont la représentation sémantique finale est :



(9.17)

En revanche, la forme résolue 2 n'est pas compatible avec la sémantique (9.18) de la suite alternative (9.10) puisque le sous-arbre dominé par le noeud *when* ne peut être à la fois dominé par la fille immédiate gauche du noeud *cause* et par sa fille immédiate droite.



(9.18)

9.3 Grammaires de descriptions et réalisation de surface

Certaines grammaires informatiques sont basées sur les descriptions d'arbres. Ainsi, [158] introduit les DTG (Description Tree Grammar) pour remédier à certaines insuffisances théoriques et empiriques des grammaires d'arbres adjoints ; [101] propose d'utiliser TDG (Tree Description Grammar) pour permettre une sous-spécification syntaxique et sémantique des structures construites ; et [148] développe les grammaires d'interaction à partir des grammaires basées sur la logique linéaire.

Pour utiliser ces grammaires – en analyse comme en génération –, on peut, bien sûr, adapter les méthodes et algorithmes existants. Cependant, une alternative intéressante est ouverte qui consiste à mettre en oeuvre les techniques de génération de modèles présentées dans les sections précédentes. Dans ce qui suit, nous explorons cette possibilité et montrons comment ces techniques peuvent être utilisées pour analyser et générer des textes à partir de grammaires de descriptions.

9.3.1 Les grammaires de descriptions

Une grammaire de descriptions (GD)[55] est un ensemble d'entrées lexicales $\langle \tau, \sigma \rangle$ où τ est une description d'arbre et σ est la représentation sémantique associée avec τ.

Descriptions d'arbres. Comme indiqué en section 9.1, une description d'arbre est une conjonction de littéraux qui spécifient soit l'étiquette d'une variable de noeud, soit sa position par rapport à d'autres variables de noeuds. Pour les grammaires de descriptions, les variables de noeud sont en outre étiquetées avec une *polarité* qui peut être positive (variable de noeud noir), négative (variable de noeud vide) ou neutre (variable de noeud gris). Intuitivement, une variable de noeud négative représente une valence ouverte qui doit être satisfaite par exactement une variable de noeud positive tandis qu'une variable de noeud neutre est une variable qui ne peut être identifiée avec aucune autre variable. Formellement, les polarités sont utilisées pour définir l'ensemble des modèles saturés. Un *modèle saturé* M pour une description d'arbre τ ($M \models_{\mathrm{S}} \tau$) est un modèle dans lequel chaque variable de noeud négative est identifiée avec exactement une variable de noeud positive ; chaque variable de noeud positive avec exactement une variable de noeud négative et les variables neutres ne sont identifiées avec aucune autre variable. Concrètement, un modèle saturé pour une description d'arbre est le plus petit arbre satisfaisant cette description et tel que toutes les valences syntaxiques sont satisfaites. Par contraste, un *modèle libre* M pour τ ($M \models_{\mathrm{L}} \tau$) est un modèle tel que chaque noeud dans ce modèle interprète exactement une variable de noeud dans τ.

Pour illustration, la figure 9.5 donne une représentation graphique d'une mini-grammaire de descriptions.

Représentation Sémantique. Le langage logique utilisé pour la représentation sémantique est une version «plate» de la logique des prédicats, c'est-à-dire une version où les propositions sont étiquetées et non récursives, la récursivité étant capturée par le jeu des étiquettes. Ce langage de représentation permet une interface syntaxe-sémantique simple, le sens associé à un arbre dérivé étant la conjonction du sens des descriptions lexicales utilisées pour le dériver, augmentée des instanciations de variables faites durant la dérivation par unification. Comme l'illustre la figure 9.5, ces variables sont des variables partagées qui appariassent à la fois dans les structures de traits étiquetant les variables de noeud et dans la représentation sémantique. Lorsque deux noeuds sont identifiés, toute instanciation par unification d'une variable partagée apparaissant

[55] Comme il l'a été mentionné plus haut, il existe plusieurs types de grammaires utilisant les descriptions d'arbres. La grammaire présentée ici diffère des DTG par son approche axiomatique. En particulier, la seule opération combinatoire est la conjonction de descriptions. La substitution et l'adjonction ne sont pas utilisées. Elle diffère des TDG, par l'utilisation d'une sémantique plate plutôt que par le recours à la synchronisation de grammaires. Enfin elle diffère des grammaires d'interaction par l'interdiction de juxtaposition qui exclut que la dénotation de deux descriptions se juxtaposent (au delà de l'identification de noeuds).

—	τ_{root}	*true*
John	τ_{john}	*true*
saw	τ_{see_1}	τ_{see_2}
Mary	τ_{mary}	*true*

FIG. 9.4 – Exemple de matrice d'analyse

dans l'arbre syntaxique est immédiatement répercutée sur la variable partagée apparaissant dans la représentation sémantique.

9.3.2 Grammaires de descriptions et analyse

Comme il est montré dans [49], l'analyse basée sur des grammaires de descriptions peut être formulée comme un problème de génération de modèles. Etant donné une phrase P de n mots, l'analyse de P consiste à trouver tous les arbres saturés qui produisent P et satisfont la description :

$$(Lex_1^1 \vee \ldots \vee Lex_m^1) \wedge \ldots \wedge (Lex_1^n \vee \ldots \vee Lex_k^n)$$

où $m \ldots k$ indiquent le nombre d'entrées lexicales associées avec le mot considéré.

Afin d'éviter l'explosion combinatoire qui résulterait d'une approche *generate and test* sur un tel problème, l'approche proposée utilise une matrice qui permet de maximiser la propagation de contraintes et de restreindre le non-déterminisme. Cette matrice représente la conjonction de disjonctions obtenues par sélection lexicale : les rangs correspondent aux mots de la phrase considérée, les colonnes aux entrées lexicales associées par la grammaire à ces mots. Chaque case de la matrice qui est vide est assignée la valeur *true* qui est vraie dans tous les modèles. La Figure 9.4 montre un exemple de matrice d'analyse pour la phrase *John saw Mary* étant donné la grammaire de la Figure 9.5.

Etant donné une matrice d'analyse, l'analyse d'une phrase consiste à :

1. sélectionner exactement une entrée par rang,
2. former la conjonction des entrées sélectionnées,
3. construire le(s) modèle(s) saturé(s) pour cette conjonction qui produisent la phrase donnée en entrée,
4. construire un modèle libre pour chacune des entrées restantes.

Cette formulation exige que toutes les contraintes présentes dans la matrice soient satisfaites permettant ainsi une propagation accrue et une réduction importante des choix possibles[56].

9.3.3 Grammaires de descriptions et Génération tactique

Dans [68], nous montrons que le modèle d'analyse de [49] s'adapte facilement à la génération de phrases.

Idée de base

Deux modifications sont nécessaires pour transformer l'analyseur en générateur. D'une part, la phase de sélection lexicale doit se faire sur la base d'une information non pas phonologique

[56]Pour une présentation plus détaillée des techniques d'analyses utilisées dans cette approche, voir [49].

mais sémantique : une description lexicale sera sélectionnée si sa sémantique subsume une partie de la sémantique donnée en entrée. D'autre part, la contrainte de bonne formation globale sera elle-aussi sémantique plutôt que phonologique : la somme des cardinalités des multi ensembles de propositions associées avec les descriptions sélectionnées doit être égale à la cardinalité de la représentation sémantique donnée en entrée. En conjonction avec l'exigence précédente selon laquelle la sémantique des entrées sélectionnées doit subsumer une partie de la sémantique donnée en entrée, cette dernière contrainte garantit que la sémantique des «arbres solutions» est identique à celle donnée en entrée.

L'exemple suivant illustre cette idée. Supposons que la sémantique donnée en entrée soit :

$$\{name(x, john), name(y, mary), see(e, x, y)\}$$

et que la grammaire soit celle donnée en Figure 9.5. La matrice de génération sera alors :

$name(x, john)$	τ_{john}	$true$
$see(e, x, y)$	τ_{see_1}	τ_{see_2}
$name(y, mary)$	τ_{mary}	$true$

Etant donné cette matrice, deux modèles de matrices seront produits : l'un avec le modèle saturé M_3 satisfaisant $\tau_{john} \wedge \tau_{see_1} \wedge \tau_{mary}$ et le modèle libre satisfaisant τ_{see_2} ; l'autre avec le modèle saturé M_4 satisfaisant $\tau_{john} \wedge \tau_{see_2} \wedge \tau_{mary}$ et le modèle libre satisfaisant τ_{see_1}. La première solution produit la phrase *John sees Mary*, la seconde la phrase avec extraction *Mary, John sees.*

Extensions

La méthode d'analyse présentée dans [68] souffre d'une restriction sévère : elle présuppose que chaque entrée lexicale soit associée avec un mot unique. Elle est en ce sens insuffisante pour des grammaires associant une entrée lexicale avec soit plusieurs mots (dans le cas d'idiomes, par exemple cf. [1]), soit aucun (dans le cas de catégories vides). Dans une optique de génération, la restriction miroir est que chaque entrée lexicale doit être associée avec une proposition unique. En d'autre termes, cette restriction exclut l'existence d'entrées lexicales ayant soit une sémantique vide, soit une sémantique multi-propositionnelle. Dans ce qui suit, nous commençons par montrer que cette restriction est trop forte. Nous montrons ensuite comment adapter le générateur pour éliminer cette restriction.

Sémantique vide. Certains mots tels la conjonction *that* ou la particule *to* peuvent être considérés comme ayant une contribution sémantique vide (cf. [171]). En génération, ces mots sont problématiques puisque la sélection lexicale exige que la sémantique du mot sélectionné subsume une partie de la sémantique donnée en entrée. Grâce à leur domaine étendu de localité, les grammaires d'arbres (et de descriptions d'arbres) permettent de pallier cette difficulté au sens où les mots à sémantique vide peuvent être inclus dans un arbre ancré par un mot à sémantique pleine. Ainsi par exemple, la conjonction *that* pourra ancrer une description en combinaison avec un mot la légitimant tel par exemple le verbe *say* et, de façon similaire, la particule *to* pourra être incluse dans les descriptions d'arbres ancrées par des verbes à l'infinitif.

Propositions multiples. Les entrées lexicales dont la sémantique est multi-propositionnelle sont très courantes. Ainsi par exemple, une sémantique à la Davidson associera la sémantique $run(e), agent(e, x)$ avec le verbe *run* ou $run(e, x), past(e), agent(e, x)$ avec sa variante *ran*. De même, une grammaire à sémantique lexicale riche pourra par exemple associer la sémantique $want(e_1, x, e_2)$, $have(e_2, x)$ avec *want* afin de pouvoir rendre compte d'exemples tels que *Reuters wants the report tomorrow* où *tomorrow* modifie *having* et non *wanting*(cf. [134]).

Parce qu'il présuppose qu'une entrée lexicale est associée avec une proposition sémantique unique, le générateur décrit ci-dessus ne peut traiter ni des entrées à sémantique vide, ni des entrées à sémantique multiples. Une méthode naïve pour remédier à ce problème consisterait à partitionner l'entrée sémantique de toutes les façons possibles, puis à utiliser ces partitions comme base de la sélection lexicale (chaque élément de la partition servant de base pour sélectionner les entrées lexicales dont la sémantique subsume cet élément).

Mais une approche de ce type est problématique, aussi bien du point de vue informatique que du point de vue théorique. Du point de vue théorique, le problème est que la partition est faite indépendamment des connaissances linguistiques et qu'il serait préférable de décomposer l'entrée sémantique à partir de l'ensemble des sémantiques lexicales défini par un dictionnaire. Du point de vue informatique, cette méthode implante une procédure *generate and test* qui est à la fois combinatoire et contraire à l'esprit de l'approche par contraintes.

Nous proposons donc une procédure alternative qui repose sur le marquage, dans chaque représentation sémantique lexicale, d'une proposition unique appelée *tête sémantique*. Ce marquage est arbitraire : peu importe quelle proposition est désignée comme étant la tête dès lors que la représentation a une tête unique. Nous utilisons ensuite cette tête pour la sélection lexicale : une entrée lexicale n'est plus sélectionnée sur la base de sa représentation sémantique globale, mais à partir de sa tête. Spécifiquement, une entrée lexicale est sélectionnée si et seulement si sa tête subsume un proposition apparaissant dans la sémantique d'entrée. Afin de préserver la cohérence du générateur, la contrainte additionnelle selon laquelle la sémantique globale de chaque entrée sélectionnée doit subsumer une partie de la sémantique d'entrée est maintenue.

Etant donné la grammaire de la Figure 9.5 (les têtes sémantiques sont soulignées) et la sémantique d'entrée $run(e, x), name(x, john), past(e)$, la matrice de génération sera :

$name(x, john)$	τ_{john}	$true$
$run(e, x)$	τ_{run}	τ_{ran}
$past(e)$	τ_{did}	$true$

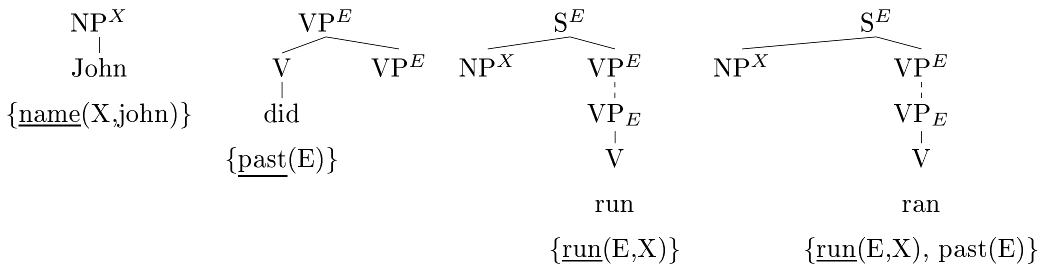


Fig. 9.5 – Une grammaire de descriptions

Pour cette matrice, deux solutions seront identifiées : une première solution impliquant l'arbre saturé satisfaisant la conjonction $\tau_{john} \wedge \tau_{ran}$ et projetant *Jon ran* et une deuxième solution impliquant l'arbre saturé satisfaisant $\tau_{john} \wedge \tau_{run} \wedge \tau_{did}$ et projetant *John did run*.

Il n'y a pas d'autres solutions puisque, pour toute autre conjonction de descriptions rendue possible par la matrice, aucun modèle saturé ne peut être produit.

Comparaison avec les approches existantes

L'approche par contraintes introduit une nouvelle perspective sur le problème de la génération tactique. Au lieu de construire – comme le font traditionnellement les algorithmes existant – des structures syntaxiques intermédiaires qui sont ensuite combinées entre elles pour former un arbre syntaxique couvrant l'ensemble de la sémantique donnée en entrée, l'approche par contraintes élimine par propagation les structures locales impossibles, ne laissant au final que la structure globale permise conjointement par les données et par la grammaire.

Dans ce qui suit, nous comparons l'approche par contraintes aux approches «constructives» existantes, et montrons qu'elle intègre les aspects positifs des générateurs ascendants et descendants se rapprochant ainsi des algorithmes de génération guidés par les têtes [172].

Génération ascendante

Les générateurs ascendants ou «guidés par les mots» (cf. [171, 196, 108, 23]) partent d'un sac d'entrées lexicales dont la composante sémantique est instanciée et génèrent un arbre syntaxique par application récursive des règles grammaticales dont le côté droit subsume un ensemble de syntagmes présent dans l'entrée.

Deux grands types de problèmes caractérisent ces générateurs. Premièrement, leur complétude n'est garantie que lorsqu'ils sont couplés avec une grammaire sémantiquement monotone, c'est-à-dire une grammaire où pour chaque règle, la sémantique de chaque fille subsume une partie de la sémantique du noeud mère. Deuxièmement, ils sont fortement non-déterministes et nécessitent pour un fonctionnement en taille réelle des optimisations encore peu étudiées telles par exemple celles proposées dans [23].

Non-déterminisme. Le non-déterminisme des générateurs ascendants a deux causes principales : l'absence d'un schéma d'indexation (dans la charte) et la présence de modificateurs intersectifs (dans la langue naturelle).

[171] présente un générateur tabulaire ascendant dénué de schéma d'indexation : tous les arcs lexicaux ont le même vertex initial et final. Chaque fois qu'un arc est ajouté à la charte, il faut donc considérer tous les arcs qui y sont déjà présents. La solution standard à ce problème (cf. [108]) consiste à indexer les arcs avec des index sémantiques (par exemple, l'arc avec catégorie $N/x : dog(x)$[57] sera indexé avec x) et à restreindre les combinaison d'arcs aux arcs ayant des indices compatibles. Ainsi, un arc actif de catégorie $A(...)/C(c...)$ où c est l'index sémantique du syntagme manquant ne peut se combiner qu'avec des arcs inactifs de catégorie $C(c...)$ et *vice versa*.

Notre générateur n'utilise pas de charte, mais les contraintes utilisées (cf. [49]) imposent une restriction similaire : deux noeuds ne pourront être identifiés que si (i) leurs polarités sont opposées et (ii) ils sont étiquetés avec le même index sémantique.

Les modificateurs intersectifs sont la deuxième grande cause du non-déterminisme des générateurs ascendants. Dans une approche constructive ascendante, le nombre des structures (arcs ou syntagmes) construites pour une phrase contenant n modificateurs intersectifs est 2^n dans le cas où la grammaire impose un ordre linéaire unique sur ces modificateurs. Par exemple, la génération du syntagme *The fierce little black cat* entraînera la construction des syntagmes intermédiaires (9.19), syntagmes qui ne sont pas utilisés dans la construction du syntagme final *The fierce little black cat.*

(9.19) a. The fierce black cat
b. The fierce little cat
c. The little black cat
d. The black cat

[57] Ici et dans ce qui suit, X/Y abrège une catégorie avec partie du discours X et représentation sémantique Y.

e. The fierce cat
f. The little cat
g. The cat

Diverses heuristiques et stratégies d'analyse ont été proposées pour résoudre ce problème. Ainsi [20] combine un générateur shift-reduce avec un mécanisme de propagation de contraintes s'appliquant entre chaque réduction. [23] propose un algorithme en deux étapes qui dans un premier temps, construit un squelette syntaxique complet et, dans un second temps, y adjoint les modificateurs. Enfin [153] utilise une méthode de reconstruction d'arbre qui améliore l'arbre syntaxique de façon incrémentale et s'arrête lorsque l'arbre construit est accepté par la grammaire.

En se basant uniquement sur les contraintes, l'approche décrite ici radicalise l'approche de Brew et permet d'explorer la capacité du mécanisme de propagation de contraintes à élaguer l'espace de recherche exploré par un générateur tactique en particulier, dans le cas des modificateurs intersectifs.

Dans la programmation par contraintes telle qu'elle est implantée en Oz, les solutions d'un problème de contraintes sont construites par une itération d'étapes de propagation et de distribution. La distribution spécifie un choix binaire sur la valeur d'une variable tandis que la propagation est un processus d'inférence déterministe qui répercute les conséquences d'un choix en éliminant toutes les valeurs de variables infèrées inconsistantes avec les contraintes données.

En spécifiant les propriétés globales du résultat attendu et en laissant la propagation répercuter les conséquences des choix locaux, les états «cul-de-sac» (c'est-à-dire ceux qui ne peuvent mener à une structure finale bien formée) peuvent être détectés rapidement. En particulier, la contrainte globale selon laquelle la sémantique de l'arbre construit doit être identique à la sémantique donnée en entrée exclut la génération des expressions données en (9.19). Dans la pratique, on observe que la propagation de contraintes est de fait extrêmement efficace pour réduire l'espace de recherche. Comme la Figure 9.6 l'indique, le nombre de points de choix pour les exemples donnés augmente très lentement avec la taille de l'entrée.

Exemple	Points de Choix	Time
The cat likes a fox	1	1.2s
The fierce cat likes a fox	1	1.4s
The fierce cat likes a tame fox	1	1.5s
The little brown cat likes a yellow fox	2	1.8s
The fierce little brown cat likes a yellow fox	2	5.5s
The fierce little brown cat likes a tame yellow fox	3	8.0s

FIG. 9.6 – Modificateurs et temps de traitement

Monotonicité sémantique. Comme nous l'avons vu plus haut, le processus de sélection lexicale extrait les entrées dont la sémantique subsume une portion de la représentation sémantique donnée en entrée. Aussi, si une règle grammaticale fait intervenir une catégorie fille dont la sémantique n'apparaît pas dans la sémantique de la catégorie mère, c'est-à-dire si la grammaire n'est pas sémantiquement monotone, cette règle ne sera jamais appliquée. Par exemple, supposons que la grammaire contienne la règle suivante :

$$\text{vp/call\_up(X,Y)} \rightarrow \text{v/call\_up(X,Y), np/Y, pp/up}$$

Et supposons que la sémantique donnée en entrée soit $call\_up(john, mary)$. A partir de cette entrée, le processus de sélection lexicale retournera les catégories V/call_up(john,mary), NP/john et NP/mary mais pas la catégorie PP/up. La phrase *John called Mary up* ne pourront donc pas être générée.

De façon plus générale, la contrainte de monotonicité sémantique rend problématique la génération de collocations et d'idiomes. Dans une grammaire d'arbres adjoints cependant, le domaine de localité étendu et en particulier le fait qu'idiomes et collocations peuvent être représentés par des arbres à ancres multiples [1] permet de contourner ce problème. Etant donné une de ces grammaires, il est en effet possible, à partir de la représentation sémantique d'un idiome, de sélectionner un arbre couvrant l'ensemble des mots intervenant dans cet idiome.

Génération descendante

On sait qu'un algorithme d'analyse descendant peut ne pas terminer lorsqu'il est combiné avec une grammaire récursive à gauche. Comme [172] le montre, un problème similaire apparaît en génération. Par exemple, supposons que la grammaire utilisée soit la suivante :

r1. s/S → np/NP, vp(NP)/S
r2. np/NP → det(N)/NP, n/N
r3. det(N)/NP → np/NP0, poss(NP0,N)/NP
r4. np/john → John
r5. poss(NP0,N)/mod(N,NP0) → s
r6. n/father → father
r7. vp(NP)/left(NP) → left

Dans une exploration descendante, en profondeur et de gauche à droite de l'espace de recherche créé par ces règles, l'algorithme ne terminera pas. En effet, la représentation sémantique de la catégorie racine `s` étant indépendante de celle du `np` dans la règle r1, la sémantique but du NP ne sera pas instanciée permettant ainsi un cycle infini d'application des règles r2 et r3.

Dans l'approche par contraintes décrite ci-dessus, ce problème de terminaison n'apparaît bien sûr pas puisque l'algorithme est guidé par les données. Ainsi par exemple, étant donné une grammaire de descriptions appropriée et la sémantique d'entrée $\{left(e,x), father(x,y), name(y,john)\}$, le générateur sélectionnera simplement les descriptions lexicales associées aux mots *left*, *John*, *s* and *father* et générera le(s) modèle(s) saturé(s) satisfaisant la conjonction de ces descriptions.

Implantation

L'approche par contraintes présentée ci-dessus a été implantée à l'aide du langage de contraintes concurrent Oz [176]. L'implantation inclut un générateur de modèles pour la logique d'arbres présentée en section 9.1.1, deux modules de sélection lexicale (un pour l'analyse et un pour la génération) et une grammaire de descriptions pour l'anglais qui a été testée sur un ensemble de phrases en analyse comme en génération. Cette implantation montre que l'utilisation de contraintes permet de restreindre le non-déterminisme (peu de points de choix cf. tableau 9.6) mais que les performances diminuent rapidement avec la longueur de l'entrée.

9.4 Conclusion

Les descriptions d'arbres sont souvent utiles pour sous-spécifier les structures linguistiques. Dans ce chapitre, nous avons vu que le traitement de ces descriptions pouvait être vu comme une tâche de génération de modèles et que cette tâche pouvait être formulée comme un problème de satisfaction de contraintes ensemblistes.

La perspective ensembliste et plus généralement l'approche par contraintes ouvrent la voie à un traitement linguistique «éliminatif» (plutôt que constructif) qui rejoint le courant de recherche portant sur le développement de grammaires basées sur la théorie des modèles [33].

Cette approche est particulièrement intéressante parce que, un peu comme les grammaires d'unification introduites il y a quelques décennies par Martin Kay, elle permet une vision différente de l'analyse linguistique informatique, une vision en particulier qui permet à la fois, une forte lexicalisation et la mise en oeuvre de principes plus généraux. Cette vision qui est sans doute, celle de la plupart des linguistes, n'a malheureusement toujours pas trouvé de cadre informatique adéquate pour s'exprimer et s'instancier. L'approche par contraintes présentée dans ce chapitre peut être vue comme un premier pas dans cette direction. Plus récemment, et à une échelle beaucoup plus crédible, les travaux de Denys Duchier sur les grammaires de dépendances basées sur les contraintes posent les bases d'un traitement par contraintes du langage naturel adapté à la fois aux exigences linguistiques et aux exigences informatiques.

Chapitre 10

Génération de descriptions définies

La génération d'expressions référentielles est une sous-tâche importante de la tâche de génération de texte. En effet, comme les exemples (10.1) l'illustrent, différents types d'expressions référentielles sont permis (pronoms, noms propres, descriptions définies, indéfinies et démonstratives, etc.). Leur bonne utilisation est en outre régie par des règles qui, si elles sont souvent mal comprises, sont essentielles pour la génération de textes de bonne qualité[58].

(10.1) a. Jean^1 est parti. Il_1 est extraordinaire.
b. Jean^1 est parti. Cet_1 homme est extraordinaire.
c. ? Jean^1 est parti. L'_1 homme est extraordinaire.
d. ? ? Il_1 / Cet_1 homme / L_1'homme va à Paris. Jean^1 est parti.
etc.

Pour les descriptions définies identifiantes[59] comme par exemple *le chat noir assis derrière la porte*, les algorithmes de génération existants sont *incrémentaux* au sens où la description est construite par ajouts successifs d'information jusqu'à ce que le critère de terminaison soit satisfait. Ainsi, pour définir le contenu d'une description définie, l'algorithme de Dale et Reiter [37] sélectionne un fait après l'autre s'arrêtant lorsque la conjonction des faits sélectionnés identifie le, ou les objets que la description doit permettre d'identifier (les **objets cibles**). De même, pour construire le syntagme réalisant les faits sélectionnés, les algorithmes de [90, 184] intègrent un calcul incrémental de la **description identifiante** (l'ensemble des faits permettant d'identifier les objets cibles de façon unique) avec une construction incrémentale de l'arbre syntaxique associé par la grammaire à cette description. Chaque fait utilisé pour mieux identifier l'objet cible sert de base pour sélectionner une entrée lexicale qui est à son tour immédiatement intégrée dans l'arbre syntaxique en cours de construction.

Dans ces approches doublement incrémentales, c'est ainsi à la fois le contenu et la forme d'une description définie qui sont déterminés par des algorithmes «greedy», c'est-à-dire par des procédures de choix *locales* et *sans retour arrière*.

Dans ce qui suit, nous montrons que cette double incrémentalité pose problème d'un point de vue aussi bien linguistique qu'informatique, puis nous proposons une alternative non-incrémentale

[58]Comme il est d'usage, les points d'interrogation indiquent les séquences dont l'acceptabilité est discutable tandis que les index et exposants marquent la coréférence.

[59]Une description définie est un syntagme nominal dont le déterminant est défini. Notons par ailleurs que les algorithmes de génération sont pour l'instant limités aux descriptions définies *identifiantes*, c'est-à-dire aux descriptions définies dont la fonction communicative est d'identifier un objet. Par opposition, les descriptions définies *attributives* [44] qui servent à communiquer une information sur l'objet décrit, ne sont généralement pas prises en compte.

qui permet d'une part, la production de descriptions identifiantes bien formées linguistiquement et d'autre part, un contrôle accru sur le processus de réalisation du syntagme nominal. Cette alternative, basée sur les contraintes, tire profit de l'inférence ensembliste.

10.1 Un algorithme non-incrémental pour la génération de descriptions définies

L'algorithme de référence pour la génération de descriptions identifiantes est celui de [37] qui permet, à partir d'un contexte et d'une entité cible, de produire une description identifiant cette cible, c'est-à-dire une conjonction de propriétés dont la dénotation est égale à l'élément cible. Etant donné un contexte comprenant un ensemble de référents D (le domaine ou univers de discours) et un ensemble de relations vraies de ces référents, [37] définit une description identifiante pour l'élément cible t de la façon suivante :

> Soit t, le référent cible et D_t l'ensemble des distracteurs de t, c'est-à-dire l'ensemble des référents autres que t présents dans le contexte ($D_t = D - \{t\}$). Alors, un ensemble L de relations sera une description identifiante pour t ssi :
>
> **C1 :** Toutes les relations contenues dans L sont vraies de t et
>
> **C2 :** Pour chaque distracteur d de t, il y a au moins une relation $r \in L$ qui n'est pas vraie de d (r exclut d).

Cette définition est relativement restrictive puisqu'elle ne s'applique en particulier qu'aux descriptions singulières (un individu unique est décrit) et aux descriptions conjonctives de relations positives ne permettant pas entre autre de générer des descriptions faisant intervenir disjonction ou négation. Comme le montre [189], il est toutefois aisé de la généraliser pour prendre en compte des ensembles de référents cibles et des combinaisons booléennes de relations et c'est sur cette généralisation que se base l'approche proposée ci-dessous.

Proposition pour une approche non-incrémentale. Bien que les algorithmes proposés par [37, 189] pour construire des descriptions identifiantes soient incrémentaux, il est clair que la définition d'une description identifiante se prête très naturellement à une formulation ensembliste. Ainsi, une description pour un ensemble cible C peut être représentée par une paire de variables ensemblistes $D_C = \langle P_C^+, P_C^- \rangle$ telle que :

- chaque relation de P_C^+ est vraie de chaque élément de C. En d'autres termes, P_C^+ doit être un sous-ensemble de l'ensemble des propriétés vraies de tous les éléments de C.
- chaque relation de P_C^- est fausse de chaque élément de C. En d'autres termes, P_C^- doit être un sous-ensemble de l'ensemble des propriétés fausses de tous les éléments de C.
- pour chaque distracteur d (c'est-à-dire pour chaque élément non contenu dans C), il y a au moins une relation P telle que soit P est vraie de tous les éléments de C mais pas de d, soit P est fausse de tous les éléments de C et vraie de d.

La Figure 10.1 formalise ces intuitions : elle propose une définition ensembliste des descriptions identifiantes qui, étant donné les contraintes ensemblistes supportées par Oz/Mozart peut, de fait, être vue comme un CSP. Ce CSP spécifie les contraintes qu'une paire de variables $\langle P_S^+, P_S^- \rangle$ doit satisfaire pour être une description identifiante. Etant données ces contraintes, le solveur de contraintes de Oz pourra ensuite énumérer les assignements de valeurs satisfaisant

$\mathcal{I}$: l'univers de discours ;
$\mathcal{P}_x^+$: l'ensemble des propriétés vraies de x ;
$\mathcal{P}_x^- = \mathcal{P} \setminus \mathcal{P}_x^+$: l'ensemble des propriétés fausses de x ;
$\mathcal{P}_C^+ = \bigcap_{x \in C} \mathcal{P}_x^+$: l'ensemble des propriétés vraies de tous les éléments de C ;
$\mathcal{P}_C^- = \mathcal{P} \setminus \bigcup_{x \in C} \mathcal{P}_x^+$: l'ensemble des propriétés fausses de tous les éléments de C ;
$D_C = \langle P_C^+, P_C^- \rangle$ est une **description identifiante** basique pour S ssi :

1. $P_C^+ \subseteq \mathcal{P}_C^+$,
2. $P_C^- \subseteq \mathcal{P}_C^-$ et
3. $\forall c \in \mathcal{I} \setminus C, |(P_C^+ \setminus \mathcal{P}_c^+) \cup (P_C^- \cap \mathcal{P}_c^+)| > 0$

FIG. 10.1 – Une définition ensembliste des descriptions identifiantes de l'ensemble cible C

le CSP, c'est-à-dire générer les descriptions identifiantes pour un ensemble cible donné, dans un contexte donné.

Oz permet en outre de spécifier une stratégie de distribution déterminant l'ordre dans lequel l'espace des solutions sera exploré. Afin de produire en premier lieu, les solutions minimales, c'est-à-dire celles contenant le moins de relations, il suffit de distribuer sur la cardinalité de la sortie ($|P_S^+ \cup P_S^-|$) en commençant par la valeur la plus petite. En d'autres termes, le solveur commencera par chercher une description $\langle P_S^+, P_S^- \rangle$ de cardinalité une, puis de cardinalité deux, etc. Il s'arrête dès qu'une solution est obtenue. De cette façon, la description produite est toujours la description identifiante la plus courte.

Extension aux propriétés disjonctives. Le solveur présenté ci-dessus étend l'algorithme de Dale et Reiter à des cibles ensemblistes et aux relations négatives. Nous montrons en Figure 10.2 comment il peut également être étendu pour prendre en compte les propriétés disjonctives.

$D_S = D_{S_1} \vee \ldots \vee D_{S_M}$ est une description identifiante pour un ensemble d'individus S ssi :
- $1 \leq M \leq |S|$
- $S = S_1 \cup \ldots \cup S_M$
- Pour $1 \leq i \leq M, D_{S_i}$ est une description identifiante basique pour S_i

FIG. 10.2 – Le CSP étendu aux propriétés disjonctives

Une description identifiante est alors une disjonction de descriptions identifiantes telle que l'union des ensembles cibles identifiés par ces descriptions est égale à l'ensemble cible. Comme auparavant, la stratégie de distribution continuera à privilégier les descriptions les plus courtes, c'est-à-dire les descriptions contenant le moins de relations atomiques possibles.

Intégration avec la réalisation de surface. Afin de permettre la génération de descriptions définies à partir de descriptions identifiantes, le solveur présenté ci-dessus doit être intégré avec un processus de réalisation de surface (un générateur tactique)[60].

Dans l'hypothèse (usuelle) où le processus de génération est guidé par des buts communicatifs et en particulier, par des buts d'information et de description, cette intégration peut être réalisée par une mise à jour, par le biais des descriptions identifiantes, des buts communicatifs : chaque fois qu'un entité doit être décrite (but de description) et qu'une description identifiante peut être

[60]Pour plus de détails sur cette intégration, cf. [58]

calculée, cette description sera ajoutée à la sémantique-but courante guidant ainsi le processus ultérieur de génération. Pour une sémantique-but donnée, le générateur cherchera à verbaliser cette sémantique en construisant un arbre syntaxiques qui (i) verbalise cette sémantique, (ii) soit syntaxiquement complet et (iii) soit pragmatiquement approprié[61].

Implantation. Le solveur de contraintes et l'algorithme de réalisation de surface décrits ci-dessus ont été implantés au sein du générateur InDiGen. Pour les exemples discutés ici, le générateur retourne une solution minimale en quelques milli-secondes. Pour une comparaison détaillée des performances de l'approche par contraintes et de l'approche incrémentale cf. [3].

10.2 Descriptions identifiantes inappropriées

L'algorithme incrémental, particulièrement lorsqu'il est généralisé pour utiliser des propriétés booléennes et décrire des ensembles d'individus, peut produire des descriptions identifiantes qui sont inappropriées, cognitivement et linguistiquement [58]. Supposons, par exemple, que le contexte soit celui représenté par la Figure 10.3 et que l'ensemble cible soit $\{x_1, x_2, x_3, x_4\}$.

	White	Small	Medium	Big	Dog	Rabbit	Cat	Horse	Sheep
x_1	•	•			•				
x_2	•	•				•			
x_3	•			•			•		
x_4	•			•			•		
x_5	•			•					•
x_6	•		•						•
x_7			•						•
x_8			•					•	
x_9			•					•	
x_{10}			•					•	

Fig. 10.3 – Un contexte

Pour construire une description identifiante pour un ensemble cible donné, l'algorithme incrémental de [189] parcourt une liste de relations suivant un ordre fixe et sélectionne à chaque nouvelle étape une relation telle que (i) tous les éléments de l'ensemble cible satisfont cette relation et (ii) cette relation exclut au moins un distracteur (l'extension de la relation sélectionnée ne peut pas être un sous-ensemble de l'ensemble des distracteurs).

Les relations sont ordonnées selon deux critères : premièrement, la longueur de la disjonction (l'algorithme considère d'abord les relations disjonctives de longueur un, puis deux etc.) et deuxièmement, certains critères sémantiques (cf. [37]). La recherche pourra se faire, par exemple, suivant l'ordre suivant :

$$\text{type} < \text{couleur} < \text{taille}$$

Ceci étant, les étapes suivies par l'algorithme incrémental pour construire une description identifiante pour l'ensemble cible $\{x_1, x_2, x_3, x_4\}$ dans le contexte décrit par la Figure 10.3 peuvent être décrites de la façon suivante (Figure 10.4 résume ce traitement).

[61] La notion d'adéquation pragmatique se restreint dans le cadre de ce mémoire à celle de descriptions identifiantes : une verbalisation sera appropriée si pour tout référent qui dans le contexte donné a déjà été évoqué et peut être identifié de façon unique, le générateur a produit une description identifiante

L'algorithme commence par chercher une relation non disjonctive dont l'extension (i) contient l'ensemble cible et (ii) n'est pas contenue dans l'ensemble des distracteurs, c'est-à-dire $\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\}$. Deux relations unaires satisfont ces critères : $\neg$ *Horse* and $\neg$ *Sheep*. Supposons que la relation $\neg$ *Horse* soit sélectionnée. Cette relation est alors ajoutée à la description identifiante (initialement vide) et l'ensemble des distracteurs est mis à jour pour devenir l'intersection de l'ensemble des distracteurs courant avec l'extension de la relation sélectionnée, c'est-à-dire $\{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$.

Ensuite les «relations de taille» sont considérées et la relation $\neg$ *Medium* est sélectionnée entraînant la réduction de l'ensemble des distracteurs à $\{x_1, x_2, x_3, x_4, x_5\}$. Comme aucune relation de couleur ne satisfait les critères de sélection, les relations disjonctives de longueur deux sont alors prises en compte. Mais aucune de ces relations ne satisfait les critères de sélection. Les relations disjonctives de longueur trois sont donc considérées et la relation *Dog* $\vee$ *Rabbit* $\vee$ *Cat* est sélectionnée aboutissant à l'ensemble de distracteurs $\{x_1, x_2, x_3, x_4\}$. A ce stade, l'algorithme termine et retourne la description identifiante suivante :

$$(\neg \textit{ Horse}) \wedge (\neg \textit{ Medium}) \wedge (\textit{Dog} \vee \textit{Rabbit} \vee \textit{Cat})$$

Dans ce cas, l'algorithme incrémental va donc produire une description identifiante qui peut être paraphrasée par la description définie :

The dogs, rabbits and cats that are not horses and that are not medium size.

alors qu'une description identifiante plus courte et plus naturelle existe qui peut être paraphrasée par

The dogs, the rabbits and the cats.

	Relation sélectionnée	Ensemble des distracteurs
Disjonction de longueur 1		
Type	$\neg$ *Horse*	$\{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$
Taille	$\neg$ *medium*	$\{x_1, x_2, x_3, x_4, x_5\}$
Couleur	Aucune	$\{x_1, x_2, x_3, x_4, x_5\}$
Disjonction de longueur 2		
Type	Aucune	$\{x_1, x_2, x_3, x_4, x_5\}$
Taille	Aucune	$\{x_1, x_2, x_3, x_4, x_5\}$
Couleur	Aucune	$\{x_1, x_2, x_3, x_4, x_5\}$
Disjonction de longueur 3		
Type	*Dog* $\vee$ *Rabbit* $\vee$ *Cat*	$\{x_1, x_2, x_3, x_4\}$

FIG. 10.4 – Construction incrémentale d'une description identifiante

Plus généralement, cet exemple illustre trois types de problèmes liés à l'utilisation d'une approche incrémentale :

– **Descriptions contextuellement redondantes :** la description produite est redondante parce qu'une relation présente dans cette description est impliquée *dans le contexte considéré* par une information présente dans une autre partie de la description. Par exemple, dans la description ($\neg$ *Horse*) $\wedge$($\neg$ *Medium*) $\wedge$(*Dog* $\vee$ *Rabbit* $\vee$ *Cat*), la relation ($\neg$ *Medium*) est contextuellement redondante puisque dans le contexte décrit par la Figure 10.3 chiens, lapins et chats sont tous soit grands soit petits et donc pas de taille moyenne.

- **Descriptions redondantes épistémiquement :** Une description est redondante épistémiquement lorsqu'une relation apparaissant dans cette description peut être déduite par le biais de connaissances encyclopédiques, à partir d'une ou plusieurs relations présentes par ailleurs dans la description. Ainsi, dans la description ci-dessus, la relation ($\neg$ *Horse*) est redondante épistémiquement puisque nous savons que chiens, chats et lapins ne sont pas des chevaux.
- **Descriptions complexes logiquement :** une description sera logiquement complexe si elle fait intervenir de façon superflue, un nombre élevé de connecteurs logiques. Par exemple, la description ci-dessus contient deux négations, deux disjonctions et deux conjonctions alors qu'une description identifiante plus simple existe qui ne contient que deux conjonctions.

Parce qu'elle produit des descriptions minimales, l'approche par contraintes présentée ci-dessus permet de pallier ces problèmes.

10.3 Descriptions définies inappropriées

En cas de succès, la sortie de l'algorithme incrémental est une description identifiante, c'est-à-dire une conjonction de relations (arbitrairement complexes) dont la dénotation est égale à l'ensemble-cible.

Pour construire une description définie, une sorte de «double incrémentalité» a parfois été proposée [90, 184] consistant à imbriquer l'algorithme incrémental pour le calcul d'une description identifiante avec un algorithme de réalisation de surface c'est-à-dire avec la construction de l'arbre syntaxique servant à verbalisant le message. Chaque relation sélectionnée est utilisée pour sélectionner une entrée dans le lexique et cette entrée est immédiatement intégrée dans l'arbre en cours de construction. De cette façon, on peut s'assurer que la description identifiante peut être réalisée par une description définie : lorsque l'entrée lexicale sélectionnée ne peut être intégrée dans l'arbre en cours de construction, une autre relation est utilisée.

Dans ces approches doublement incrémentales, aucune information *globale* n'est disponible pour guider le processus de construction du syntagme nominal ce qui parfois aboutit à des syntagmes peu acceptables. Reprenons l'exemple ci-dessus. Comme nous l'avons noté, la description identifiante produite par l'algorithme incrémental pourrait dans ce cas être :

$$(\neg\ \mathit{Horse})\ \wedge(\neg\ \mathit{Medium})\ \wedge(\mathit{Dog} \vee\ \mathit{Rabbit} \vee\ \mathit{Cat})$$

Si, comme les algorithmes doublement incrémentaux le proposent, les propriétés présentes dans cette description sont réalisées au fur et à mesure de leur sélection, les différentes étapes de la réalisation de la description définie correspondante seront :

(10.2) The non horses.
The non medium sized non horses.
The non medium sized non horses that are either dogs, rabbits or cats.

alors qu'une réalisation bien plus naturelle de cette description identifiante existe :

(10.3) The dogs, rabbits and cats that are not horses and that are not medium size

Le problème est que l'ordre dans lequel l'algorithme incrémental sélectionne les relations est déterminé par des contraintes qui sont largement orthogonales à celles régissant l'ordre dans lequel ces relations devraient être réalisées pour obtenir un syntagme acceptable. De façon plus générale, l'absence d'information globale sur le contenu sémantique à verbaliser empêche une optimisation, par le processus de génération, de la structure globale du syntagme. Ainsi par exemple, le générateur est dans le cas ci-dessus incapable de reconnaître que la «meilleure» description définie réalisant la description identifiante calculée est un GN constitué de trois GN disjoints et de deux modificateurs conjoints (plutôt que d'un GN avec trois modificateur disjoints).

En revanche, un réalisateur de surface guidé par une sémantique-but peut utiliser la structure de cette sémantique pour, à la fois, planifier et optimiser la structure syntaxique du syntagme à générer.

Pour commencer, un tel générateur n'échouera que si la sémantique-but n'est pas réalisable. Par contraste, l'algorithme incrémental peut faire un choix local qui mène à une impasse alors qu'une description définie est en fait constructible. Supposons par exemple que la sémantique-but soit une description identifiante telle que :

- les relations à réaliser soient P_1, P_2, P_3,
- les entrées lexicales associées soient $T_1^1 \dots T_1^n,\ T_2^1 \dots T_2^k,\ T_3^1 \dots T_3^m$
- et la seule combinaison d'entrées lexicales permettant la construction d'un SN bien formé soit T_1^1, T_2^4, T_3^2

Si l'algorithme incrémental choisit la séquence initiale T_1^1, T_2^3, aucune entrée lexicale verbalisant P_3 ne pourra être incluse dans l'arbre courant et la génération échouera.

En outre, différentes stratégies d'optimisation peuvent être mises en oeuvre à partir de la structure présente dans la description identifiante. Ainsi par exemple, la forme de la sortie produite par le solveur de contraintes décrit à la section 10.1 (une forme normale disjonctive, c'est-à-dire une disjonction de conjonctions de littéraux) peut être utilisée pour décider du nombre de conjoints de la description à générer et, récursivement, du nombre de disjoints contenus dans chaque conjoint. De façon similaire, parce que les relations négatives et positives sont clairement identifiées, des stratégies de lexicalisation peuvent être envisagées pour optimiser la structure du SN. Par exemple, si la description identifiante contient plusieurs relations négatives, des antonymes peuvent être utilisés afin de minimiser le nombre de négations explicitement verbalisées. Enfin, des techniques d'agrégation (factorisation) et de lexicalisation (choix des mots) peuvent être appliquées à l'entrée afin d'éviter soit une verbalisation répétée de relations apparaissant plusieurs fois dans la description identifiante donnée en entrée, soit une description inutilement longue. Ces techniques permettront par exemple, de générer *les chiens et chats noirs* plutôt que *les chiens noirs et les chats noirs* – par le biais de l'agrégation ; et *la conductrice de la BMW* plutôt que *la femme qui conduit la BMW* – par le biais de la lexicalisation.

Un dernier avantage de l'algorithme non incrémental présenté en section 10.1 ci-dessus est qu'il fournit une entrée plus riche pour la réalisation de surface que celle retournée par l'algorithme de Dale et Reiter. Rappelons à ce propos que la sortie produite par cet algorithme est un ensemble de relations. Elle ne contient en particulier pas d'information sur l'extension de ces relations et donc, aucune information sur leur cardinalité. Pour cette raison, il est impossible de faire des choix informés sur le nombre du GN : la cardinalité des relations à verbaliser étant inconnue, il est impossible de décider de façon appropriée si le syntagme engendré doit être pluriel (cardinalité supérieure à un) ou singulier (cardinalité égale à un).

Par contraste, la sortie de l'algorithme non incrémental pour le calcul des descriptions identifiantes est une séquence de descriptions D^{1-n} doublée de la séquence des ensembles correspondants S^{1-n} (chaque description D_i dans D^{1-n} est une description identifiante pour l'ensemble correspondant S_i dans S^{1-n}). Ce couplage entre descriptions et ensembles décrits permet de déterminer le nombre du SN correspondant : si l'ensemble est de cardinalité un, le SN correspondant sera un syntagme singulier, sinon ce sera un syntagme pluriel.

10.4 Conclusion

La génération d'expressions référentielles pose des problèmes linguistiques et informatiques non triviaux. Dans l'approche ensembliste, ce sont les aspects linguistiques qui ont la préséance au sens où la bonne formation linguistique des descriptions est privilégiée au détriment de la complexité informatique des algorithmes utilisés pour les produire. En effet, les algorithmes incrémentaux sont nés de considérations essentiellement informatiques. Comme le montre [37], trouver une description identifiante minimale est une tâche NP difficile puisqu'elle peut être formulée comme un problème de couverture minimale d'ensembles [71]. L'utilisation d'algorithmes incrémentaux permet de rester dans des classes de complexité polynomiales. Mais c'est à un coût linguistique non négligeable puisque ces algorithmes ne sont ni complets (ils peuvent échouer à produire une description même si une description existe [189]), ni minimaux (ils ne produisent pas nécessairement la plus petite description identifiante possible).

L'approche ensembliste elle, est aussi bien complète que minimale. Elle est en revanche peu efficace comme le montre la comparaison des performances relatives de l'approche ensembliste et de l'approche incrémentale présentée dans [3]. Néanmoins une complexité théorique n'est jamais qu'une complexité du «pire cas» et de nombreux travaux ont montré qu'en pratique une complexité théorique élevée n'entraîne pas nécessairement des temps de traitement importants[62]. On peut ainsi espérer que l'approche ensembliste puisse être suffisament optimisée pour être utilisable. Un premier pas dans cette direction consisterait à compléter le CSP présenté avec des contraintes additionnelles reflétant certaines exigences ou préférences linguistiques et psycholinguistiques. On pourrait, par exemple, contraindre une description identifiante à contenir une propriété dénotant le type ontologique de l'objet décrit (chien, chat, chiens et chats, etc.). Ou encore partitionner les propriétés comme cela est fait dans les algorithmes incrémentaux selon des classes ontologiques naturelles (couleur, taille, forme, etc.) et utiliser ces partitions pour contraindre la stratégie de recherche.

D'un point de vue linguistique, il est clair que comme l'approche incrémentale, l'approche ensembliste ne rend compte que d'une partie de l'iceberg. Outre qu'elle se limite aux descriptions définies alors que comme nous l'avons vu, le répertoire des expressions référentielles est très riche, elle ne rend compte que d'une partie relativement minime des emplois de ces descriptions à savoir les usages anaphoriques et identifiants reposant sur des propriétés données. Pour pallier ce problème, nous explorons deux pistes en parallèle : d'une part l'extension des algorithmes pour la génération de descriptions définies aux description définies dites «associatives», c'est-à-dire de cas où le lien entre anaphore et antécédent est un lien méronymique (relation partie-tout) plutôt que coréférentiel [66] ; d'autre part, l'extension de ces algorithmes à des descriptions définies ou démonstratives communiquant une information nouvelle (plutôt que connue) sur les référents-cibles [65].

[62]Les prouveurs de théorèmes pour les logiques de descriptions en sont un bon exemple : ils permettent d'utiliser des logiques de plus en plus expressives sans pour autant buter sur des problèmes d'inefficacité insurmontables dans la pratique.

Cinquième partie

Perspectives

A travers l'étude de différents types d'inférences et de leur utilité pour le traitement de la langue naturelle, ce mémoire identifie un certain nombre de phénomènes linguistiques et de sous-tâches du TAL pour lesquels l'inférence est requise :

- la résolution de l'ellipse et des énoncés désaccentués (chapitre 1, 2, 3),
- la désambiguïsation des réciproques ainsi que des anaphores nominales et relationnelles (chapitre 6),
- la lexicalisation et la production de descriptions définies en génération de textes (chapitres 7 et 10),
- l'identification de paires question/réponse (chapitre 7),
- la représentation et le traitement incrémental d'ambiguïtés discursives (chapitre 9)
- et le traitement de grammaires basées sur les descriptions d'arbres (chapitre 9)

Même si ce n'est qu'une étape préliminaire pour le traitement de l'inférence dans le TAL, cet inventaire permet de mettre en avant plusieurs points importants.

Premièrement, l'utilisation de procédures inférentielles prédéfinies permet une dimension opérationnelle souvent absente des analyses linguistiques et sans laquelle il peut être difficile de vérifier les prédictions faites par l'analyse proposée. L'usage de l'unification d'ordre supérieur pour le traitement interprétatif des constructions parallèles par exemple a permis de passer «d'analyses papier» telle que la théorie de Fiengo et May [51], à des analyses opérationnelles c'est-à-dire, à des analyses dont on peut tester les prédictions sur ordinateur. Etant donné les interactions complexes entre interprétation des constructions parallèles, résolution des anaphores et ambiguités de portée, la vérification sur machine des prédictions d'une telle théorie n'est certainement pas un luxe. De façon plus générale, la complexité des contraintes linguistiques et de leur interaction suggèrent qu'une modélisation informatique des théorisations linguistiques n'est pas superflue.

Deuxièmement, l'application à certains phénomènes linguistiques ou à certains sous-problèmes du TAL de techniques inférentielles existantes ouvre souvent une perspective nouvelle sur un problème ancien offrant ainsi de nouvelles possibilités algorithmiques ou formelles. L'utilisation des contraintes pour le traitement des descriptions d'arbres par exemple, d'une part a permis le développement rapide de prototypes efficaces sur des masses intermédiaires de données et d'autre part, a abouti à une re-formulation du problème en termes de graphes et d'algorithmes portant sur les graphes. De même, l'utilisation des contraintes pour la génération de descriptions définies offre une alternative intéressante à l'approche incrémentale, permettant en particulier l'intégration simple de contraintes additionnelles et surtout une généralisation directe au cas des descriptions définies plurielles.

Enfin, le recours à l'inférence est dans certains cas au moins, incontournable. C'est le cas en particulier, de l'inférence logique dont il est clair, bien qu'on puisse argumenter sur le type de logique nécessaire, qu'il est nécessaire à une modélisation fine des processus interprétatifs impliqués dans l'activité langagière. Sur ce point, notre contribution a été d'identifier des tâches précises et suffisament «locales» pour pouvoir être traitées sans intervention humaine : les connaissances impliquées ne portent pas sur le monde en général mais plutôt sur le sens des mots.

Pour aller plus loin et en particulier, pour permettre un traitement en taille réelle des tâches d'inférence identifiées, plusieurs conditions doivent être satisfaites. Il convient d'abord de développer les ressources (grammaires sémantiques, ontologies, base de connaissances lexicale) et l'infrastructure TAL nécessaires (analyseur, résolveur, générateur, etc.). Il faut ensuite adapter et optimiser les outils d'inférence utilisés par rapport aux données linguistiques. Enfin il importe d'évaluer les résultats obtenus à partir des approches proposées : ces résultats sont-ils corrects/redondants/partiels au plan linguistique ? Les temps de traitements sont ils acceptables ? etc.

Notre travail actuel porte sur le développement parallèle d'une infrastructure pour la sémantique computationnelle (ressources, outils, etc.) et d'ensembles de tests permettant l'évaluation de cette infrastructure. Dans ce qui suit, nous discutons des problèmes soulevés par ces objectifs et de la façon dont nous pensons les aborder.

10.5 Développement d'une grammaire sémantique.

Dans ce mémoire, nous avons, dans une large mesure présupposé d'une grammaire et d'un analyseur permettant d'associer une représentation sémantique à toute expression de la langue naturelle. Mais ce présupposé est infondé si bien qu'un tâche importante concerne le développement d'une «grammaire sémantique» pour le français, c'est-à-dire, d'une grammaire associant à chaque expression de la langue naturelle non seulement une structure syntaxique mais également une représentation sémantique. Pour ce faire, nous prenons pour base la grammaire d'arbres adjoints développée par Anne Abeillé pour le français. Cette grammaire du français est en effet unique par bien des respects :

- elle utilise un formalisme linguistique précis dont les propriétés mathématiques et informatiques sont à la fois bien comprises et favorables au traitement automatique ;
- elle a une couverture non triviale (335 000 formes et 7 5000 items lexicaux) basée sur des ressources lexicales extensives et fines (les dictionnaires du LADL, des concordances du journal *Le Monde*, etc.) ;
- elle a été évaluée sur la base de phrases tests TSNLP (Test Suite for Natural Language Processing) qui comporte plusieurs milliers de phrases grammaticales et agrammaticales ;
- elle est documentée de façon à la fois précise, exhaustive et systématique par le livre de Anne Abeillé, *Une grammaire électronique du français.*

Équiper cette grammaire d'une dimension sémantique passe par un travail informatique (intégration et évaluation d'une composante sémantique) et linguistique (définition d'un langage de représentation sémantique et d'une méthode d'évaluation).

Pour les aspects informatiques, nous nous appuyons sur les travaux nancéens qui suivant les travaux de M.H. Candito [22] visent au développement d'un formalisme méta grammatical et d'un compilateur de métagrammaire pour les grammaires d'arbres adjoints. L'idée est ici de permettre une factorisation de l'information linguistique et par là, une meilleure modularité dans la description des faits linguistiques. Le linguiste, au lieu d'écrire une grammaire, spécifie une métagrammaire décrivant les concepts linguistiques clés puis les combine pour définir les items lexicaux constituant le lexique de la grammaire. A partir de ces concepts et de ces définitions, c'est ensuite le compilateur qui produit les entrées du lexique. Outre la minimisation des risques d'erreurs permise par la factorisation, le compilateur fait intervenir plusieurs mécanismes de vérification qui conjointement facilitent l'implantation d'une grammaire correcte et cohérente.

Au delà des aspects informatiques, se pose la question de ce qu'est une «bonne grammaire sémantique». Notre but est ici de développer une grammaire à bon pouvoir paraphrastique c'est-à-dire une grammaire qui dans la mesure du possible associe à différentes paraphrases du même contenu, une même représentation sémantique. Une telle grammaire associerait la même représentation sémantique aux trois phrases données en 10.4 par exemple.

(10.4) a. La croisière coûte cher.
b. Le coût de la croisière est élevé.
c. La croisière a un coût élevé.

En effet, si ces trois phrases peuvent présenter des différences au plan pragmatique et/ou communicatif (par exemple en ce qui concerne, leurs présuppositions ou leur structure informationnelle), leur sémantique compositionnelle de base est à priori la même.

Pouvoir rendre compte de l'équivalence sémantique entre paraphrases est crucial pour la correction de nombreuses applications en particulier, pour les systèmes de question/réponse et pour les systèmes de génération. Nous avons ainsi testé deux systèmes de question/réponse sur les différents types de paraphrases illustrés par les exemples 10.5 ci-dessous.

(10.5) a. *Verbe/Déverbal événementiel*
What site did Lindbergh begin to fly from in 1927 ?
What site did Lindbergh begin his flight from in 1927 ?

b. *Verbe/Déverbal agentif* Who invented the telephone ?
Who is the inventor of the telephone

c. *Adjectif/Nom*
How far is the Earth from Mars ?
What is the distance between Mars and the Earth ?

d. *Variante terminologique*
Who was president in 1881 ?
Who was Head of State in 1881 ?

e. *Verb/Converse*
Which films did Jane Birkin act in ?
Which films starred Jane Birkin ?

f. *Antonymes*
Which president was unmarried ?
Which president was single ?

Le premier système AnswerBus ne traite pas de la paraphrase, le deuxième (Start) plus sophistiqué fait intervenir des règles paraphrastiques permettant de créer des variantes de la requête donnée en entrée. Les résultats obtenus (cf. figure 10.5) montrent que pour les deux systèmes, il existe une forte proportion de cas où seule l'une des deux paraphrases permet d'obtenir des réponses. En d'autres termes, si la «mauvaise» paraphrase est utilisée, le système ne trouve pas de réponse. En outre, Start qui traite certains des types de paraphrases illustrés ci-dessus, identifie un ensemble de réponses identiques pour les deux paraphrases dans 39% des cases (contre 0% pour AnswerBus)[63]

La génération de textes est un autre domaine où le traitement des paraphrases est essentiel. D'un point de vue purement théorique, un module de réalisation de surface doit à priori pouvoir produire toutes et seulement les verbalisations d'un même contenu. En pratique cependant, il est souvent contré que le pouvoir paraphrastique du langage est tel, qu'il est impensable de systématiquement générer toutes les formulations possibles d'un contenu donné. Ainsi par exemple, [138] montre que la phrase (10.6) accepte entre deux et trois millions de paraphrases.

(10.6) Le style de persécutions policières des gens de lettres en Union Soviétique a évidemment connu, depuis un demi-siècle, des changements sérieux.

S'il n'est de fait sans doute pas concevable de développer des systèmes de génération produisant l'ensemble des paraphrases d'un contenu sémantique donné, il n'en est pas moins vrai que le choix d'une paraphrase donnée est contrainte par son contexte linguistique ce qui implique de pouvoir générer plusieurs variantes du même contenu pour pouvoir ensuite choisir parmi celles ci celle qui est compatible avec ce contexte. Ainsi par exemple, si les deux variantes en (10.7a)

[63]Les systèmes ayant des champs d'application distincts, ils n'ont pu être testés sur les mêmes paires de paraphrases et la distribution relative des différents types de paraphrases est différente pour chacun des deux systèmes. C'est ce qui explique la relativement mauvaise performance de Start.

		AnswerBus		START		
Type de Paraphrase	N_{tot}	N_{same}	N_{one}	N_{tot}	N_{same}	N_{one}
Verb/Déverbal	33	0	42	10	80	20
V/Déverbal agentif	21	0	19	12	42	33
Adj/Nom	27	0	30	13	31	31
V. Termino	7	0	43	4	50	50
Converse	34	0	32	7	14	86
Antonyme	12	0	36	6	17	67

N_{tot} nombre de paires de paraphrases analysées
N_{same} Nombre des cas en pourcentage où la même réponse est trouvée pour les deux paraphrases
N_{one} Nombre des cas en pourcentage où une réponse n'est trouvée que pour une des deux paraphrases

FIG. 10.5 – Paraphrases et Systèmes de Questions/Réponses

sont interchangeables dans un contexte nul ou dans le contexte d'un verbe de perception tel que «voir» (10.7b), elles ne le sont pas dans le contexte d'un verbe à contrôle (10.7c). Il faudra donc pour que le générateur n'échoue pas à verbaliser le contenu désiré, qu'il soit en mesure de générer aussi bien la première que la deuxième variante.

(10.7) a. Marie emprunte un livre à Paul.
Paul prête un livre à Marie.
b. Jean a vu Marie emprunter un livre à Paul.
Jean a vu Paul prêter un livre à Marie.
c. Jean persuade Marie d'emprunter un livre à Paul.
Jean persuade Paul de prêter un livre à Marie.

Ces deux applications (systèmes de question/réponse et réalisation de surface) sont donc des applications où un bon traitement de la paraphrase est important. Nous visons à développer un traitement «grammatical» de la paraphrase c'est-à-dire une traitement qui s'appuie autant que possible sur la grammaire pour associer à différentes formulations d'un même contenu, une même représentation sémantique. En effet ce traitement ne va pas de soi et les grammaires sémantiques à large couverture qui existent pour l'anglais ne rendent pas compte de ces relations. Par exemple étant donné les deux phrases :

(10.8) a. The person who drives the car is mad.
b. The driver of the car is mad.

la grammaire HPSG de Ann Copestake and Dan Flickinger retourne deux représentations sémantiques distinctes :[64] :

(10.9) a. the(x, person(x) $\wedge$ the(y, car(y) $\wedge$ drive(e,x,y) $\wedge$ mad(x)))
a. the(y, car(y) $\wedge$ the(x, driver(x,y) $\wedge$ of(x,y)) $\wedge$ mad(x))

Il est bien sur possible d'argumenter que si elles sont syntaxiquement distinctes, ces deux formules peuvent être sémantiquement équivalentes étant donné les connaissances lexicales pertinentes. Mais une solution qui évite de placer cette charge supplémentaire sur le processus de raisonnement sémantique est sans nul doute meilleure.

[64]Les représentations ont été simplifiées dans un souci de lisibilité. Les représentations produites par le systèmes sont en fait les suivantes :
prpstn(def(x,person(x)$\wedge$prpstn(def(y,car(y), drive(e1,v1,x,y,v2),v3)), mad(e2,x,v4),v5)
prpstn(def(x,person(x)$\wedge$prpstn(def(y,car(y), drive(e1,v1,x,y,v2),v3)), mad(e2,x,v4),v5)
prpstn(def(y,car(y)$\wedge$prpstn(def(x, driver(x,y) $\wedge$ of(e1,x,y,v1), mad(e2,x,v2,v3)))))

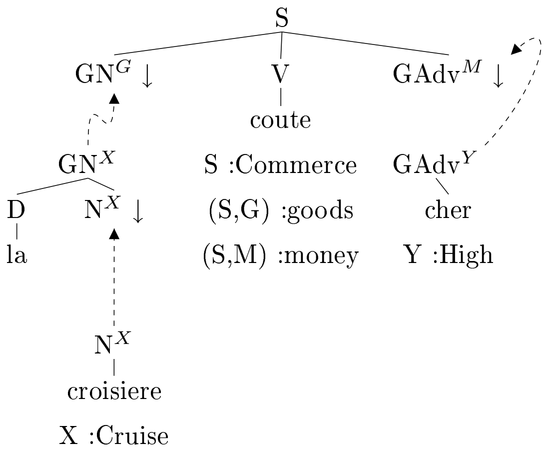


FIG. 10.6 – La croisiere coûte cher

La grammaire que nous souhaitons développer est une grammaire d'arbres adjoints où l'interface syntaxe/sémantique est modélisée selon les principes décrits dans [60]. Le développement de la composante sémantique implique d'une part le choix d'un ensemble de primitives sémantiques et d'autre part, la mise en place d'une méthodologie permettant d'évaluer sa couverture paraphrastique.

Pour le premier point (le choix d'un ensemble de primitives sémantiques), nous projetons d'utiliser le répertoire de concepts et de rôles établi par FrameNet [29]. FrameNet est une ressource lexicale informatisée et disponible sur le web qui est développée selon les principes de la théorie des cadres de Fillmore. Dans cette approche, chaque mot évoque un cadre sémantique (une éventualité ou une situation) et chaque cadre est associé à un certain nombre d'éléments cadres caractérisant les acteurs de ce cadre. Enfin chaque cadre est associé avec un ensemble de mots, les mots qui évoquent ce cadre.

En bref, FrameNet associe aux synonymes un même concept à savoir, le cadre évoqué par ces synonymes. Nous projetons de mettre à profit ce trait de FrameNet et de baser notre langage de représentation sémantique sur les cadres et les rôles de FrameNet. Ainsi par exemple, parce que les paraphrases de l'exemple (10.4) évoquent le cadre COMMERCE et instantient deux de ses éléments cadres, GOODS et MONEY, leur représentation sémantique sera :

(e,g,m) :commerce, g :cruise, (e,g) :goods, m :high, (e,m) :money

Etant donné cette représentation, la grammaire devra associer avec chacune des paraphrases en (10.4) cette même représentation. Les figures 10.6, 10.7 et 10.8 montre comment la grammaire d'Anne Abeillé peut être équipée d'une sémantique compositionnelle dérivant cette représentation pour les trois formulations.

Pour le second point (la mise en place d'une méthodologie permettant d'évaluer la couverture paraphrastique de la grammaire), nous nous appuyons sur une suite de tests structurée qui regroupe les paraphrases entre elles. L'idée générale est de développer une suite de tests qui soit basée sur des considérations sémantiques plutôt que, comme c'est le cas dans TSNLP, par des considérations syntaxiques, ces considérations sémantiques dictant à la fois la macro- et la micro-structure de la suite. Par exemple, la macro-structuration de la suite pourrait être basée sur des critères de variations entre items tels que (1) la famille WordNet du prédicat principal ; (2) son type aspectuel ; (3) Son arité etc.

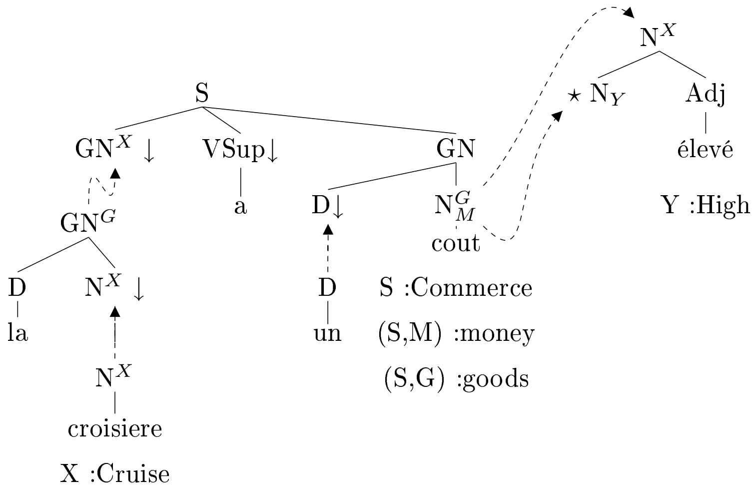


FIG. 10.7 – *La croisiere a un coût élevé*

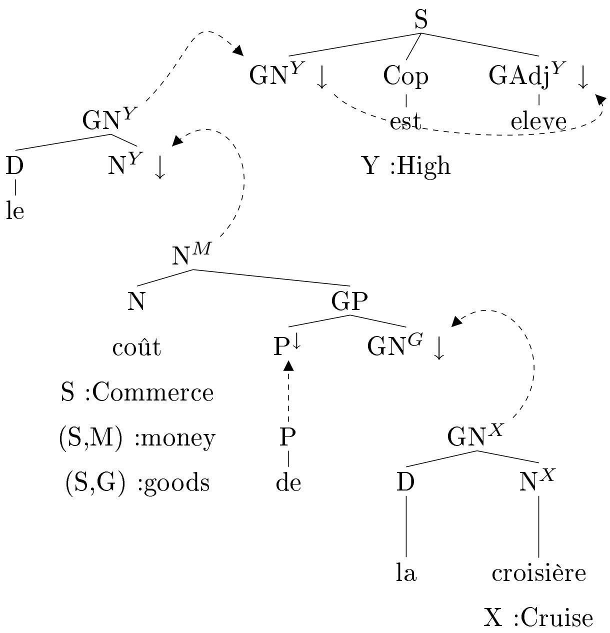


FIG. 10.8 – Le cout de la croisière est élevé

La micro-structuration en revanche est dictée par la relation paraphrastique au sens où l'unité de base est la paraphrase. Plus précisément, une unité englobera un contenu sémantique, les verbalisations de ce contenu et les annotations syntaxico-sémantiques jugées pertinentes. Pour illustration, on pourra avoir pour une application en génération, l'item de base suivant :

ID : 1
Semantics : (e,g,m) :commerce, g :cruise, (e,g) :goods, m :high, (e,m) :money
Sem : [11.2, state, arity_2]
Realisations :
Variant1 : La croisière coûte cher
Syntax : [+V, [GN, V, Adv]]
Variant2 : Le coût de la croisière est élevé
Syntax : [+N, [GN(GP), cop, Adj], cop]
Variant3 : La croisière a un coût élevé
Syntax : [+N, [GN, avoir, GN(Adj)], sup(avoir)]

L'annotation sémantique **Semantics** (**Sem**) classifie l'item par rapport aux critères sémantiques mentionnés supra. Pour chaque paraphrase, **Syntax** est un ensemble d'annotations linguistiques et applicatives. Par exemple, +V dans la première variante indique que le prédicat principal (ici : Cost) est réalisé par un verbe tandis que la séquence [GN, V, Adv] décrit les constituants majeurs présents dans la phrase.

10.6 Évaluation

Afin de pouvoir évaluer la pertinence linguistique de la grammaire paraphrastique que nous projetons de développer, des outils sont nécessaires dont en particulier, un analyseur, un générateur et une suite de tests.

Nous avons indiqué plus haut les pistes à poursuivre pour le développement de la suite de tests.

Pour l'analyse, il s'agit essentiellement d'adapter un analyseur TAG existant de façon à permettre le calcul sémantique illustré par les dérivations données en figures 10.6, 10.7 et 10.8.

Pour la génération en revanche, le modèle proposé diffère de celui proposé dans [41, 40] et reste à explorer. En effet tandis que le modèle de Danlos part d'un niveau de représentation conceptuel (l'arbre de g-dérivation) plus abstrait que l'arbre de dérivation pour reconstruire par des choix lexicaux, syntaxiques et morphologiques successifs, un arbre de dérivation puis un arbre dérivé, l'approche que nous poursuivons est une sorte d'analyse inversée où la grammaire est utilisée pour générer les verbalisations associées par cette grammaire à la représentation sémantique donnée en entrée.

Comme pour l'analyse, le nombre élevé d'arbres élémentaires associés *a priori* à chaque prédicat sémantique multiplie les possibilités de combinaison. C'est pourquoi il est important de mettre en oeuvre des techniques d'optimisation permettant de réduire les temps de traitement et le nombre des structures intermédiaires inutiles.

Nous travaillons actuellement avec Carlos Areces (CR INRIA) et Eric Kow (DEA Nancy) au développement d'un réalisateur de surface utilisant une grammaire d'arbres adjoints à sémantique plate du type illustré par les figures 10.6, 10.7 et 10.8 et présenté dans [60]. Ce générateur intègre une charte et utilise en priorité l'opération de substitution (de préférence à l'adjonction) permettant ainsi de minimiser le nombre de structures intermédiaires superflues construites en cas de modifieurs multiples d'une même tête (e.g., *j'ai vu le chat de la voisine, près de la souris, dans une folle rage*, cf. [23]). Mais ces premières optimisations ne suffisent pas pour une utilisation avec une grammaire d'arbres adjoints de taille réelle[65]. Nous projetons donc d'explorer et d'évaluer l'effet de différents types d'optimisations actuellement utilisées pour l'analyse dont en particulier, les techniques de préfiltrage proposées par [177, 97], le préfiltrage basé sur les polarités de [149] et les techniques plus heuristiques de suppertagging [177, 178, 25] et d'hypertagging [113].

Une autre piste de recherche particulièrement importante à explorer dans une optique de génération, est celle de la définition de principes de préférences permettant de choisir parmi les paraphrases possibles, la paraphrase la plus adaptée aux but du locuteur. Plusieurs pistes sont possibles. Ainsi, [11] propose d'utiliser la théorie de l'optimalité pour ordonner des verbalisations différant dans leur ordonnancement et la forme de leurs expressions référentielles. Par ailleurs, [112] utilise des pondérations pour optimiser la «cohérence référentielle» d'un texte c'est-à-dire, l'interaction entre ordonnancement des syntagmes et utilisation de pronoms. Enfin, nous envisageons d'explorer l'utilisation des polarités de Guy Perrier pour guider le choix de la paraphrase la plus adaptée.

10.7 Vers un traitement sémantique en taille réelle

Une fois les outils de base mis en place, on peut envisager à la fois une extension des ressources créées visant à traiter du texte libre et un enrichissement des ressources et du traitement sémantique.

10.7.1 Extension à des textes arbitraires

Une évaluation à plus grande échelle portera sur du texte libre plutôt que sur des batteries de tests et pourra se faire par exemple, dans le cadre d'un système de question/réponse ou d'un système de traduction automatique.

Dans les deux cas, deux grands types de problèmes devront être traités : celui de la couverture (lexicale, syntaxique et sémantique) et celui de la Désambiguïsation.

Des grammaires à large couverture. Pour la couverture lexicale (parties du discours, information morpho-syntaxique), il s'agit d'établir un répertoire associant a chaque lemme, une ou plusieurs catégories morpho-syntaxiques. Par exemple, le lemme *rire* sera associé avec la catégorie *V* ainsi qu'avec la catégorie *N, gen :masc* et *voile* avec les catégories *N, gen :masc* et *N, gen :fem*. Les formes fléchies de chaque lemme ainsi que l'information morpho-syntaxique associée seront soit énumérées, soit générées automatiquement par un fléchisseur tel que PC-Kimno.

Plusieurs méthodes et ressources existent qui permettent d'obtenir une bonne couverture. Les lexiques et dictionnaires informatisés tels ceux développés dans le cadre du projet Multext [191] ou au LADL [174] fournissent une bonne base de départ. Mais si ces connaissances encodées manuellement sont très fines et fiables, elles sont aussi inévitablement incomplètes. On peut les compléter par des méthodes d'acquisition automatiques, en exécutant un programme d'étiquettage morpho-syntaxique (taggeur) sur un corpus de grande taille par exemple ou en extractant les informations pertinentes d'un dictionnaire informatisé à large couverture tel par exemple le TLFI (Trésor de la langue française informatisé, [43]). Plusieurs initiatives locale, nationales et internationales sont actuellement en cours qui visent à explorer les différentes façons d'obtenir à

[65]Selon [8], le dictionnaire syntaxique utilisé par la grammaire d'arbres adjoints d'Anne Abeillé contient une moyenne de 139 arbres élémentaires pour chaque forme verbale.

partir de ces diverses méthodes et ressources, un lexique morpho-syntaxique à large couverture pour le français – il sera intéressant de coupler le résultat de ces initiatives avec le développement d'une grammaire d'arbres adjoints pour le français à large couverture.

Pour ce qui est de la couverture syntaxique, la grammaire d'arbres adjoints d'Anne Abeillé est une grammaire noyau du français qui reste incomplète. Le développement d'un corpus arboré pour le français [2] devrait cependant permettre de pallier cet inconvénient et on pourra s'appuyer sur des méthodes d'extraction automatique de grammaires [199, 26] pour compléter cette grammaire noyau.

Enfin, les aspects sémantiques donneront vraisemblablement lieu à un travail similaire à celui utilisé en syntaxe. Nous pensons en particulier, au développement d'un corpus arboré syntaxico-sémantique où chaque phrase serait associée non seulement à une structure syntaxique mais également à une représentation sémantique. L'annotation sémantique d'un corpus de grande taille permettrait dans un premier temps une meilleure analyse des phénomènes sémantiques à couvrir et de la généralité du système de codage mis en place dans la phase d'expérimentation basée sur les suites de tests. La mise au point de méthodes d'extraction automatique devrait en outre permettre à partir d'un tel corpus arboré de produire une grammaire d'arbres adjoints syntaxico-sémantique à large couverture pour le français.

Le problème de la Désambiguïsation. Nous avons déjà évoqué ce problème *supra* en relation avec la tâche de génération : non seulement la langue naturelle est ambiguë, mais les grammaires informatisées sont telles que pendant le traitement (analyse ou génération), de nombreuses structures intermédiaires sont produites qui multiplient les ambiguités et diminuent les performances des systèmes. Il est donc impératif, pour un traitement en temps réel de mettre en oeuvre des techniques de Désambiguïsation aussi bien au niveau local (ambiguités ne se projetant pas au niveau de la structure globale associée par l'analyseur ou le générateur à la phrase traitée) qu'au niveau global (choix parmi les différentes structures produites, de la structure la plus plausible). Les techniques que nous avons évoquées *supra* pour la génération sont des techniques issues de l'analyse et sont donc disponibles et utilisables. Il s'agira de les intégrer et de les améliorer. Il sera également intéressant d'explorer en quoi l'information sémantique contenue par la grammaire peut influencer l'ambiguïté et/où la Désambiguïsation. Par exemple, les techniques de préfiltrage utilisées par les analyseurs TAG sont basées sur des critères essentiellement syntaxiques (nombre de valences, nombres de noeuds a droite d'une tête lexicale etc.). Il serait intéressant de compléter ces critères par des critères plus sémantiques telles que les restrictions de sélection et d'examiner l'effet de pareille modification sur le traitement.

10.7.2 Affinement du traitement sémantique

L'approche décrite dans les sections précédentes est une première étape à la fois nécessaire (elle pose les bases pour un calcul du sens compositionnelle des expressions de la langue naturelle) et relativement grossière (elle ne rend pas compte de la polysémie, des métaphores, des métonymies etc.) du traitement sémantique de la langue naturelle.

Parce que ce traitement est une étape nécessaire, il est important d'en extraire l'information qui lui est sous-jacente et de systématiser cette information dans ce que nous appellerons, un lexique sémantique informatisé qui sera mis à disposition de la communauté scientifique. Parce qu'il est insuffisant, il est nécessaire de l'enrichir en tenant compte en particulier, d'un nombre maximum d'informations sémantiques et en y ajoutant des mécanismes permettant de dépasser le sens compositionnelle.

Un lexique sémantique. Le travail de recherche décrit ci-dessus vise en particulier, au développement d'une grammaire d'arbres adjoints sémantique à bon pouvoir paraphrastique. Une telle grammaire peut être vue comme un lexique informatisé associant à chaque forme fléchie d'un lemme, un ensemble de structures syntaxiques et une représentation sémantique. En outre, parce que la grammaire est une grammaire paraphrastique, elle encode de façon indirecte les liens de synonymie et d'antonymie (deux synonymes auront la même représentation sémantique, deux antonymes auront la même représentation sémantique modulo une négation). Ce lexique met donc en relation une information morphosyntaxique riche avec une information sémantique incluant représentation sémantique de base, synonymie et antonymie. Il pourra être étendu à la fois en augmentant sa couverture sémantique par le biais par exemple, de dictionnaires de synonymes et d'antonymes et en y ajoutant d'autres relations lexicales telle que l'hyperonymie et la méronymie. Pour cette dernière extension, on pourra s'appuyer en particulier sur l'EuroWordNet pour le Français ainsi que sur des processus d'acquisition automatique (par exemple, un processus d'extraction automatique des liens d'hyperonymie contenus dans le TLFI). Une autre source intéressante d'enrichissement du lexique concerne la morphologie dérivationnelle qui permet d'extraire, à partir de la sémantique combinée de la racine et des affixes, des traits sémantiques et des définitions [32]. Ainsi [146] montre par exemple, qu'il est possible d'inférer du fait que *abreuvoir* résulte de la combinaison de la racine *abbreuv-* avec l'affixe *-oir* les traits CONCRET, INANIMÉ, LIEU OU INSTRUMENT ; ou encore d'inférer que la sémantique de *portable* est *qu'il est possible de porter.* Il serait intéressant de voir dans quel mesure ces travaux sur l'acquisition automatique de traits et de sens lexicaux peuvent être utilisé pour enrichir un lexique sémantique informatisé.

Au delà du sens compositionnel. L'idée de développer un système permettant le calcul du sens compositionnel du type décrit dans ce mémoire est criticable. En effet, elle repose sur l' hypothése non fondée selon laquelle le sens des mots est donné indépendamment du contexte linguistique et situationnel. On sait pourtant que le sens d'un mot est extrêmement variable et dépend en grande partie de son contexte. Ainsi l'adjectif *plat* aura des sens très différents selon qu'il apparaît dans le syntagme *une route place, de l'eau plate* ou *un style plat.* Comme les exemples suivant l'illustrent, les prépositions sont elles aussi extrêmement polysémiques (*Paul a construit ce meubles avec ces planches/avec un marteau/avec Jean/avec application*). Plus généralement, divers travaux ont clairement montré que pour traiter de la créativité sémantique du langage, des représentations et mécanismes sont nécessaires qui sont nettement plus sophistiqués que ceux proposés par l'approche montagovienne (cf. [73] pour une revue des problèmes posés par l'approche Montagovienne et des approches proposant un traitement «dynamique» du calcul sémantique).

Il est clair qu'un traitement sémantique approfondi ne pourra se faire sans l'introduction et l'utilisation d'un processus plus dynamique de construction du sens que celui mis en avant par l'approche Montagovienne. Néanmoins, nous pensons qu'il n'est pas inutile, dans un premier temps, de développer puis de tester les limites d'une pareille approche. Une fois ces limites clairement identifiées, il sera intéressant d'examiner les extensions les plus urgentes et de les modéliser. On pourra s'appuyer entre autres sur les travaux de Patrick Saint-Dizier concernant l'utilisation de représentations sémantiques lexicales fines et sous-spécifiées [169] ; sur la théorie du lexique génératif [157] ; et sur l'utilisation des logiques des défauts [31, 109].

10.8 Connaissances et inférence

Parallèlement au travail de spécification et de développement des outils nécessaires au raisonnement sémantique (grammaire paraphrastique, base de connaissances lexicales, analyseur et générateur), il importe de continuer à préciser les tâches d'inférence logiques impliquées par le traitement automatique des langues.

Notre travail dans ce domaine porte sur la lexicalisation (chapitre 7), sur la relation question/réponse (chapitre 7) et sur les anaphores associatives.

Pour la lexicalisation, le travail empirique visera à identifier puis à modéliser dans le cadre des logiques de descriptions, les ensembles de faits «définitoires», c'est-à-dire ceux qui correspondent à une définition lexicale et peuvent donc être exprimés par l'élément défini. On s'appuiera dans ce travail sur la lexicologie et les types de définitions qu'elle préconise ; sur la morphologie dérivationnelle [32, 146] et la description qu'elle permet de la sémantique des mots dérivés ; et sur les diverses ressources électroniques existantes (gloses de WordNet, dictionnaires électroniques, etc.).

Alors que le travail sur la lexicalisation porte essentiellement sur les notions de synonymie et de définition lexicale, la recherche sur la relation entre question et réponse élargit le champ d'exploration aux paraphrases : il s'agit de déterminer si une réponse potentielle est dans une relation paraphrastique avec la question considérée. La comparaison question/réponse est donc un champ d'étude privilégié pour l'évaluation du pouvoir paraphrastique de la grammaire. Nous pensons l'utiliser à la fois pour évaluer le pouvoir paraphrastique d'une grammaire d'arbres adjoints du type décrit à la section 10.5 et pour évaluer celui d'une grammaire plus superficielle mais plus robuste du genre de celles qui peuvent être développée avec l'analyseur XIP (Xerox Incremental Parser) de Xerox.

Enfin le traitement des anaphores associatives fait intervenir des connaissances de différents types. Une étude de corpus sur le français [65] montre que si environ 50% des cas font intervenir la méronymie (relation partie/tout), environ 30% des cas font intervenir une définition lexicale, les quelques 20% restant impliquant soit des connaissances encyclopédiques, soit la structure du discours. Nous envisageons de modéliser les cas identifiés dans la cadre des logiques de descriptions et d'intégrer ces connaissances dans des algorithmes d'analyse et de génération.

Plus généralement, le travail empirique permettra d'évaluer et de consolider les décisions théoriques prises sur la forme et le contenu du lexique sémantique ; et de mesurer les performances des prouveurs existants sur les données linguistiques.

Bibliographie

[1] A. Abeillé and Y. Schabes. Parsing idioms in lexicalised TAGs. In *Proceedings of EACL '89*, Manchester, UK, 1989.

[2] A. Abeillé, L. Clément, and F. Toussenel. *Building and using Parsed Corpora*, chapter Building a treebank for French. KLUWER, 2003.

[3] M. Amoia, C. Gardent, and S. Thater. Using set constraints to generate distinguishing descriptions. In *Proceedings of the 7th International Workshop on Natural Language Understanding and Logic Programming (NLULP'02*, Copenhagen, Denmark, 2002.

[4] D. Appelt. *Planning English Sentences.* Cambridge University Press, Cambridge, England, 1985.

[5] N. Asher. *Reference to abstract objects in discourse.* Kluwer, Dordrecht, 1993.

[6] N. Asher and A. Lascarides. Bridging. *Journal of Semantics*, 15 :83–113, 1998.

[7] V. Barr and J. Klavans. Verification and validation of language processing systems : Is it evaluation ? In *Proceedings of ACL Workshop on Evaluation Methodologies for Langauge and Dialogue Systems*, Toulouse, France, 2001.

[8] N. Barriet, S. Barrier, and A. Kinyon. Lexik : a maintenance tool for ftag. In *Actes du colloque TAG+5*, 2000.

[9] J. Barwise and J. Perry. *Situations and Attitudes.* Bradford Books, Cambridge, MA, 1983.

[10] P. Baumgartner and M. Kühn. Abducing coreference by model construction. In Christof Monz and Maarten de Rijke, editors, *Proceedings of Inference in Computational Semantics*, pages 21–39, Amsterdam, 1999.

[11] D. Beaver. The optimization of discourse anaphora. *Linguistics and Philosophy*, 2002.

[12] F. Benamara, C. Gardent, and P. Saint-Dizier. Représentation et inférence dans un système pour la génération de réponses coopératives. In *Actes des Journées scientifiques Sémantique et Modélisation"*, Paris, 2003.

[13] P. Blackburn and J. Bos. Representation and inference in natural language : A first course in computational semantics. Volume II : Working with Discourse Representations. Latest draft available at `http ://www.coli.uni-sb.de/~patrick/`, 1999.

[14] P. Blackburn, J. Bos, M. Kohlhase, and H. de Neville. Inference and Computational Semantics. In *Third International Workshop on Computational Semantics (IWCS-3)*, Tilburg, The Netherlands, 1999.

[15] P. Blackburn and C. Gardent. A specification language for discourse. In *Proceedings of Logical Aspects of Computational Linguistics*, 1998.

[16] P. Blackburn, C. Gardent, and W. Meyer-Viol. Talking about trees. In *Proceedings of EACL '93*, Utrecht, 1993.

[17] J. Bos. Implementing the binding and accommodation theory for anaphora rseolution and presupposition projection. *Computational Linguistics*, 29(2) :179–210, 2003.

[18] J. Bos and M. Gabsdil. First-order inference and the interpretation of questions and answers. In *Proceedings of Goetalog 2000. Fourth Workshop on the Semantics and Pragmatics of Dialogue*, 2000.

[19] R. J. Brachman and H.J. Levesque. Competence in knowledge representation. In *Proc. of the National Conference on Articial Intelligence*, pages 189–192, 1982.

[20] C. Brew. Letting the cat out of the bag : Generation for shake-and-bake MT. In *Proceedings of COLING '92*, Nantes, France, 1992.

[21] S Bry, F. et Torge. Minimal model generation with positive unit hyperresolution tableaux. In D. Mundici et M. Ornaghi P. Miglioli, U. Moscato, editor, *Proceedings TABLEAUX'96*, pages 143–159, 1996.

[22] M.-H. Candito. *Representation modulaire et parametrable de grammaires electroniques lexicalisees.* PhD thesis, University of Paris 7, 1999.

[23] J. Carroll, A. Copestake, D. Flickinger, and V. Paznański. An efficient chart generator for (semi-)lexicalist grammars. In *Proceedings of EWNLG '99*, 1999.

[24] E. Charniak. *Towards a Model of Children's Story Comprehension.* PhD thesis, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1972.

[25] J. Chen, B. Srinivas, and K. Vijay-Shanker. New models for improving supertag disambiguation. In *Actes conférence EACL*, Bergen, 1999.

[26] J. Chen and K. Vijay-Shanker00. Automatated extraction of tags from the penn treebank. In *Actes du 6e International Workshop on Parsing Technology*, 2000.

[27] G. Chierchia and R. Turner. Semantics and property theory. *Linguistics and Philosophy*, 1983.

[28] A. Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5 :56–68, 1940.

[29] C.Johnson, C. Fillmore, M. Petruckand C. Baker, M. Ellsworth, and J. Ruppenhofer. Framenet : Theory and practice. Technical report, Berkeley, 2002.

[30] R. Cooper. The interpretation of pronouns. In F. Heny and H.S. Schnelle, editors, *Syntax and Semantics*, number 10, pages 61–93. 1979.

[31] A. Copestake and A. Lascarides. Default representation in constraint-based frameworks. *Computational Linguistics*, 25 :55–107, 1999.

[32] D. Corbin. *Morphologie dérivationnelle et structuration du lexique.* P.U.L, 1991.

[33] T. Cornell and J. Rogers. Model theoretic syntax. In L. Cheng and R. Sybesma, editors, *The GLOT International State of the Article Book 1.* Holland Academic Graphics, The Hague, To appear.

[34] S. Crain and M. Steedman. On not being led up the garden path : the use of context by the psychological parser. In Lauri Kartunnen, David Dowty, and Arnold Zwicky, editors, *Natural Language Parsing*, pages 320–358. Cambrdige University Press, 1985.

[35] D. Cruse. *Lexical Semantics.* CUP, Cambridge, UK, 1986.

[36] Ö. Dahl. On so-called sloppy identity. Gothenburg Papers in Theoretical Linguistics 11, University of Göteborg, 1972.

[37] R. Dale and E. Reiter. Computational interpretations of the gricean maxims in the generation of referring expressions. *Cognitive Science*, 18 :233–263, 1995.

[38] M. Dalrymple, M. Kanasawa, Y. Kim, S. Mchombo, and S. Petes. Reciprocal expressions and the concept of reciprocity. *Linguistics and Philosophy*, 21(2) :159–210, April 1998.

[39] M. Dalrymple, S. Shieber, and F. Pereira. Ellipsis and higher-order unification. *Linguistics & Philosophy*, 14 :399–452, 1991.

[40] L. Danlos. Gtag : un formalisme lexicalisé pour la génération. *TAL*, 2000.

[41] L. Danlos and F. Meunier. Gtag : présentaiton et applications industrielles. In *Actes 3e Conférence ILN*, 1996.

[42] H. de Nivelle. A resolution decision procedure for the guarded fragment. In *CADE 15*, 1998.

[43] J. Dendien and J.M. Pierrel. Le trésor de la langue française informatisé : un exemple d'informatisation d'un dictionnaire de langue de référence. *TAL*, 2003.

[44] K. Donnellan. Reference and definite descriptions. *Philosophical Review*, 1966.

[45] G. Dowek. Third order matching is decidable. In *LICS7*, pages 2–10, 1992.

[46] D. Duchier and C. Gardent. A constraint-based treatment of descriptions. In H.C. Bunt and E.G.C. Thijsse, editors, *Proceedings of IWCS-3*, Tilburg, 1999.

[47] D. Duchier and C. Gardent. Tree descriptions, constraints and incrementality. In *Computing Meaning*, volume 2 of *Studies in Linguistics and Philosophy Series*. Kluwer Academic Publishers, 2001.

[48] D. Duchier and J. Niehren. Dominance constraints with set operators. In *Proceedings of CL2000*, LNCS. Springer, 2000.

[49] D. Duchier and S. Thater. Parsing with tree descriptions : a constraint-based approach. In *NLULP'99*, Las Cruces, New Mexico, 1999.

[50] C. Fellbaum, editor. *WordNet : An Electronic Lexical Database*. MIT Press, 1998.

[51] R. Fiengo and R. May. *Indices and Identity*. MIT Press, Cambridge, 1994.

[52] L. Frazier. *On comprehending sentences : Syntactic Parsing Strategies*. PhD thesis, University of Connecticut, 1979.

[53] C. Fuchs. La signification linguistique appréhendée à travers l'ambiguïté et la paraphrase. *Modèles linguistiques*, V(2), 1983.

[54] C. Fuchs. *Paraphrase et énonciation*. Ophrys, Paris, 1994. en particulier, le chapitre IV.

[55] C. Gardent. Dynamic semantics and VP ellipsis. In Jan van Eijck, editor, *Logics in AI*, pages 251–267, Amsterdam, The Netherlands, 1990. Springer-Verlag.

[56] C. Gardent. Deaccenting and higher–order unification. *Journal of Logic, Language and Information*, 1999.

[57] C. Gardent. Unifying parallels. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 49–57, Maryland, USA, 1999.

[58] C. Gardent. Generating minimal definite descriptions. In *40th Annual Meeting of the Association for Computational Linguistic - ACL'02, Philadelphia, USA*, Jul 2002.

[59] C. Gardent and E. Jacquey. Lexicalisation as a description logic inference task. In *Proceedings of the 4th International Workshop on Computational Semantics*, Nancy, France, 2003.

[60] C. Gardent and L. Kallmeyer. Semantic construction in ftag. In *Proceedings of the 10th meeting of the European Chapter of the Association for Computational Linguistics*, Budapest, Hungary, 2003.

[61] C. Gardent and M. Kohlhase. Focus and higher–order unification. In *Proceedings of the 16th International Conference on Computational Linguistics*, Copenhagen, 1996.

[62] C. Gardent and M. Kohlhase. Computing parallelism in discourse. In *Proceedings of IJCAI '97*, pages 1016–1021, Tokyo, 1997.

[63] C. Gardent and K. Konrad. Interpreting Definites using Model Generation. *Journal of Language and Computation*, 2000.

[64] C. Gardent and K. Konrad. Understanding "each other". In *Proceedings of the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics*, Seattle, US, 2000.

[65] C. Gardent, H. Manuelian, K. Striegnitz, and M. Amoia. Generating definite descriptions : Non incrementality, inference and data. In T. Pechman and C. Habel, editors, *Speech production*. Walter de Gruyter, Berlin, 2003.

[66] C. Gardent and K. Striegnitz. Generating Indirect Anaphora. In *Fourth International Workshop on Computational Semantics (IWCS-4)*, Tilburg, The Netherlands, 2001.

[67] C. Gardent and K. Striegnitz. Generating bridging definite descriptions. En soumission., 2003.

[68] C. Gardent and S. Thater. Generating with a grammar based on tree descriptions : a constraint-based approach. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, Toulouse, France, 2001.

[69] C. Gardent and B. Webber. Varieties of ambiguity in incremental discourse processing. In *Proceedings of Architectures and Mechanisms for Language Processing*, Freiburg,Germany, 1998.

[70] C. Gardent and B. Webber. Automated deduction and discourse disambiguation. *Journal Logic, Language and Information*, 10, 2001.

[71] W. Garey and D. Johnson. *Computers and Intractability : a Guide to the Theory of NP-Completeness*. W.H.Freeman, San Francisco, 1979.

[72] J.-M. Gawron. Comparatives, superlatives, and resolution. *Linguistics & Philosophy*, 18(4) :333–380, 1995.

[73] F. Gayral. Créativité du sens en contexte et hypothèse de compositionnalité. *TAL*, 1998.

[74] G. Gazdar, E. Klein, G. Pullum, and I. Sag. *Generalized Phrase Structured Grammar*. Blackwell, Oxford, UK, 1985.

[75] P. Gorrell. *Syntax and Parsing*. Cambridge University Press, 1995.

[76] William Eben Gould. A matching procedure for ω-order logic. Technical report, Applied Logic Corporation, One Palmer Square, Princeton, NJ, 1966.

[77] P. Grice. Logic and conversation. In P. Cole and J. Morgan, editors, *Syntax and semantics 3 : Speech acts*, pages 41–58. Academic Press, New York, 1975.

[78] J. Groenendijk and M. Stokhof. On the semantics of questions and the pragmatics of answers. In *Varieties of Formal Semantics*. Foris Publication, 1984.

[79] C. Grover, C. Brew, S. Manandhar, and M. Moens. Priority union and generalisation in discourse grammar. In *Proceedings of the 31th Annual Meeting of the Association for Computational Linguistics*, 1994.

[80] V. Haarslev and R. Möller. Description of the racer system and its applications. In *Proceedings International Workshop on Description Logics (DL-2001*, Stanford, USA, 2001.

[81] D. Hardt. Dynamic interpretation of VP ellipsis. *Linguistics and Philosophy*, 22 :187–221, 1999.

[82] Irene Heim. *The Semantics of Definite and Indefinite Noun Phrases*. PhD thesis, University of Massachusetts, 1982.

[83] J. Hindley and J. Seldin. *Introduction to Combinators and Lambda Calculus*. Cambridge University Press, 1986.

[84] E. Hinrichs. Temporal anaphora in discourses of English. *Linguistics & Philosophy*, 9 :63–82, 1986.

[85] J. Hintikka. Model minimization – an alternative to circumscription. *Journal of Automated Reasoning*, 4 :1–13, 1988.

[86] L. Hirschman, M. Light, E. Breck, and J. Burger. Deep read : A reading comprehension system. In *Proc. of the 37th Annual Meeting of the Association for Computational Linguistics*, 1999.

[87] J. Hobbs and A. Kehler. A theory of parallelism and the case of VP ellipsis. In *Proceedings of ACL*, Madrid, 1997.

[88] J. Hobbs, M. Stickel, D. Appelt, and P. Martin. Interpretation as abduction. *Artificial Intelligence*, 63 :69–142, 1993.

[89] Jerry Hobbs. *Literature and Cognition*. Number 21 in CSLI Lecture Notes. CSLI, 1990.

[90] H. Horacek. An algorithm for generating referential descriptions with flexible interfaces. In *Proceedings of the 35^{th} Annual Meeting of the Association for Computational Linguistics)*, pages 206–213, Madrid, 1997.

[91] I. Horrocks. 'fact'. In *Proceedings of the International Workshop on Description Logics*, pages 62–63, Trento, Italy, 1998.

[92] I. Horrocks and P. Patel-Schneider. Dl systems comparison (summary relation). In *Proceedings of the International Workshop on Description Logics*, Trento, Italy, 1998.

[93] G. P. Huet. The undecidability of unification in third order logic. *Information and Control*, 22(3) :257–267, 1973.

[94] G. P. Huet. A unification algorithm for the typed λ-calculus. *Theoretical Computer Science*, 1 :27–57, 1975.

[95] D. Hutter and M. Kohlhase. A coloured version of the λ-calculus. In *Proceedings of CADE'97*, pages 291–305, 1997.

[96] D. Hutter and M. Kohlhase. Managing structural information by higher-order colored unification. *Journal of Automated Reasoning*, 1999. forthcoming.

[97] F. Issac. *Analyse syntaxique et apprentissage des langues*. PhD thesis, Paris XIII, 1997.

[98] Ray S. Jackendoff. *Semantic Interpretation in Generative Grammar*. The MIT Press, 1972.

[99] C. Jacquemin. Syntagmatic and paradigmatic representations of term variation. In *Actes du 37e Annual Meeting of the Association for Computational Linguistics*, 1999.

[100] P. Johnson-Laird. *Mental Models : Towards a cognitive science of language, inference and consciousness*. Cambridge University Press, 1983.

[101] L. Kallmeyer. *Tree Description Grammars and Underspecified Representations*. PhD thesis, Universität Tübingen, 1999.

[102] H. Kamp. Events, instant and temporal reference. In U. Egli R. Bauerel and A. von Stechow, editors, *Semantics from Different Points of View*, pages 376–417. Springer Verlag, 1979.

[103] H. Kamp. A theory of truth and semantic representation. In J. Groenendijk, Th. Janssen, and M. Stokhof, editors, *Formal Methods in the Study of Language*, pages 277 – 322. Mathematisch Centrum Tracts, Amsterdam, 1981.

[104] H. Kamp and U. Reyle. *From Discourse to Logic*. Kluwer, Dordrecht, 1993.

[105] J. Kaplan. Cooperative responses from a portable natural language database query system. In *Computational Models of Discourse*, pages 167–208. MIT Press, Cambridge MA, 1982.

[106] R. Kaplan and J. Bresnan. Lexical-functional grammar : A formal system for grammatical representation. In *The Mental Representation of Grammatical Relations*, pages 173–280. MIT Press, 1982.

[107] J.J. Katz and J.A. Fodor. The structure of a semantic theory. *Language*, 39 :170–210, 1963.

[108] M. Kay. Chart generation. In *Proceedings of ACL'96*, Santa Cruz, USA, 1996.

[109] D. Kayser and H. Abir. A non-monotonic approach to lexical semantics. In *Computational Lexical Semantics*. CUP, 1995.

[110] A. Kehler. *Interpreting Cohesive Forms in the Context of Discourse Inference*. PhD thesis, Harvard University, 1995.

[111] A. Kehler and S. Shieber. Anaphoric dependencies in ellipsis. *Computational Linguistics*, 23(3) :457–467, September 1997.

[112] R. Kibble and R. Power. Optimising referential coherence in text generation. *Computational Linguistics*, 2002.

[113] A. Kinyon. Hypertags. In *Actes Conférence COLING*, Sarrebruck, 2000.

[114] K. Knight. Unification : a multidisciplinary survey. *ACM Computing Surveys*, 21 :93–124, 1989.

[115] M. Kohlhase. Higher-order order-sorted resolution, 1994.

[116] A. Koller, R. Debusman, M. Gabsdil, and K. Striegnitz. Put my galkmid coin into the dispenser and kick it : Computational linguistics and theorem provind in a computer game. *Language and Computation*, 2001.

[117] A. Koller, J. Niehren, and R. Treinen. Dominance constraints : Algorithms and complexity. In *Third International Conference on Logical Aspects of Computational Linguistics*, Grenoble, 1998.

[118] K. Konrad and D. A. Wolfram. Kimba, a model generator for many-valued first-order logics. In *Proc., 16th International Conference on Automated Deduction (CADE)*, LNCS, forthcoming. Springer, 1999.

[119] A. Kratzer. The representation of focus. In Arnim van Stechow and Dieter Wunderlich, editors, *Semantik : Ein internationales Handbuch der zeitgenoessischen Forschung*. Berlin : Walter de Gruyter, 1991.

[120] M. Krifka. A compositional semantics for multiple focus constructions. In J. Jacobs, editor, *Informationsstruktur and Grammatik*. Heidelberg, 1992. Sonderheft 4.

[121] M. Krifka. Focus and/or context : A second look at second occurence expressions. Unpublished Ms. University of Texas, Austin, February 1995.

[122] R. Ladd. *The structure of intonational meaning.* Indiana University Press, Bloomington, 1980.

[123] G. Lakoff. Presupposition and relative well–formedness. In Danny Steinberg and Leon Jakobovits, editors, *Semantics : An Interdisciplinary Reader in Philosophy, Linguistics and Psychology*, pages 329–340. Cambridge University Press, 1971.

[124] D. T. Langendoen. The logic of reciprocity. *Linguistic Inquiry*, 9(2) :177–197, 1978.

[125] Tore Langholm. *Partiality, Truth and Persistence.* CSLI Publications, 1988.

[126] A. Lascarides and N. Asher. Temporal interpretation, discourse relations and common sense entailments. *Linguistics & Philosophy*, 16 :437–493, 1993.

[127] A. Lascarides and A. Copestake. Pragmatics and word meaning. *Journal of Semantics*, 16, 1999.

[128] M. Light, G. Mann, E. Riloff, and E. Breck. Analyses for elucidating current question answering technology. *Journal of Natural Language Engineering*, 2001.

[129] D. Litman and J. Allen. Discourse processing and commonsense plans. In P. Cohen, Jerry Morgan, and Martha Pollack, editors, *Intentions in Communication*, pages 365–388. MIT Press, Cambridge MA, 1990.

[130] S. Lorenz. A tableau prover for domain minimization. *Journal of Automated Reasoning*, 13 :375–390, 1994.

[131] E. Mays. *A Modal Temporal Logic for Reasoning about Changing Data Bases with Applications to Natural Language Question Answering.* PhD thesis, University of Pennsylvannia, 1984.

[132] E. Mays, A. Joshi, and B. Webber. Taking the initiative in natural language data base interactions : Monitoring as response. In *Proceedings of the European Conference on Artificial Intelligence*, pages 255–256, Orsay, France, 1982.

[133] J. McCarthy. Circumscription - a form of non-monotonic reasoning. *Artificial Intelligence*, 13 :27–39, 1980.

[134] J. D. McCawley. *Adverbs, Vowels, and other objects of Wonder.* University of Chicago Press, Chicago, Illinois, 1979.

[135] M. McCord and S. Lappin. Anaphora resolution in slot grammar. *Computational Linguistics*, 16 :197–212, 1990.

[136] W. McCune. Automatic proofs and counterexamples for some ortholattice identities. *Information Processing Letters*, 65 :285–291, 1998.

[137] W. McCune and R. Padmanabhan. *Automated Deduction in Equational Logic and Cubic Curves.* Springer Verlag, 1996.

[138] I. Mel'čuk. Paraphrase et lexique dans la théorie linguistique sens-texte. *Lexique*, 6 :13–54, 1988.

[139] R. Montague. Universal Grammar. In R.H. Thomason, editor, *Formal Philosophy : Selected Papers of R. Montague.* Yale University Press, 1970.

[140] R. Montague. The proper treatment of quantification in ordinary English. In R. Thomason, editor, *Formal Philosophy. Selected Papers.* Yale University Press, New Haven, 1974.

[141] M.P.Marcus, D. Hindle, and M.M.Fleck. Talking about talkning about trees. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, Cambridge, MA, 1983.

[142] T. Müller. *Constraint Propagation in Mozart.* Doctoral dissertation, Universität des Saarlandes, Naturwissenschaftlich-Technische Fakultät I, Fachrichtung Informatik, Saarbrücken, Germany, 2001.

[143] R. Muskens and E. Krahmer. LTAGs and underspecified semantics. In *Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks*, Philadelphia, PA, 1998.

[144] Reinhard Muskens. *Meaning and Partiality.* CSLI, Stanford, 1995.

[145] Reinhard Musskens. Combining Montague semantics and discourse representation. *Linguistics & Philosophy*, 14 :143 – 186, 1996.

[146] F. Namer. Acquisition automatique de sens à partir d'opérations morphologiques en français : études de cas. In *Actes de TALN*, 2002.

[147] P. Patel-Schneider. Dlp system description. In *Proceedings of the International Workshop on Description Logics*, Trento, Italy, 1998.

[148] G. Perrier. Interaction grammars. In *In Proceedings of 18th International Conference on Computational Linguistics (COLING 2000)*, 2000.

[149] Guy Perrier. From intuitionistic proof nets to interaction grammars. Technical report, LORIA, Nancy, France, 1999.

[150] T. Pietrzykowski and D. Jensen. Mechanizing ω-order type theory through unification. Internal Report CS-73-16, Department of Applied Analysis and Computation, University of Waterloo, 1972.

[151] M. Pollack. *Inferring Domain Plans in Question-Answering.* PhD thesis, University of Pennsylvannia, 1986.

[152] Carl Pollard and Ivan Sag. *Head-driven Phrase Structure Grammar.* CSLI and University of Chicago Press, 1994.

[153] V. Poznanski, J. L. Beaven, and P. Whitelock. An efficient generation algorithm for lexicalist MT. In *Proceedings of ACL '95*, 1995.

[154] H. Prüst, R. Scha, and M. van den Berg. Discourse grammar and verb phrase anaphora. *Linguistics & Philosophy*, 17 :261–327, 1994.

[155] S. Pulman. Grammars, parsers and memory limitations. *Language and Cognitive Processes*, 1(3) :197–225, 1986.

[156] S. Pulman. Higher order unification and the interpretation of focus. *Linguistics & Philosophy*, 20 :73–115, 1997.

[157] J. Pustejovsky. *The generative lexicon.* MIT Press, Cambridge, 1995.

[158] O. Rambow, K. Vijay-Shanker, and David Weir. D–Tree Grammars. In *Proceedings of ACL'95*, 1995.

[159] A. Ramsay and H. Seville. Models and discourse models. In C. Monz and M. de Rijke, editors, *Proceedings of Inference in Computational Semantics*, pages 111–125, Amsterdam, 1999.

[160] E. Reiter. A new model of lexical choice for nouns. *Computational Intelligence*, 1991.

[161] E. Reiter and R. Dale. *Building Natural Language Systems.* Cambridge, 2000.

[162] Christian Retoré. Calcul de lambek et logique linéaire. *Traitement Automatique des Langues*, 37(2) :39–71, 1997.

[163] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the Association for Computing Machinery*, 12(1) :23–41, 1965.

[164] M. Rooth. Ellipsis redundancy and reduction redundancy. In S. Berman and A. Hestvik, editors, *Proceedings of the Stuttgart Ellipsis Workshop*, University of Stuttgart, 1992.

[165] M. Rooth. A theory of focus interpretation. *Natural Language Semantics*, pages 75–116, 1992.

[166] J.R. Ross. *Constraints on Variables in syntax.* PhD thesis, MIT, 1967.

[167] E. Sacerdotti. *A Structure for Plans and Behavior.* Elsevier, 1977.

[168] Ivan A. Sag. *Deletion and Logical Form.* Garland Publishing, New York and London, 1980.

[169] P. Saint-Dizier. Sense variation and lexical semantics generative operations. In *Proceedings of the Joint Conference on New Methods in Language Processing and Computational Natural Language Learning*, 1998.

[170] J. Searle. *Speech Acts.* Cambridge University Press, 1969.

[171] S. Shieber. A Uniform Architecture for Parsing and Generation. In *Proceedings of ACL '88*, 1988.

[172] S. Shieber, F. Pereira, G. van Noord, and R. Moore. Semantic-head-driven generation. *Computational Linguistics*, 16(1), 1990.

[173] Stuart Shieber, Fernando Pereira, and Mary Dalrymple. Interaction of scope and ellipsis. *Linguistics & Philosophy*, 19 :527–552, 1996.

[174] M. Silberztein. *Dictionnaires électroniques et analyse automatique de textes : le systèmes INTEX.* Masson, 1993.

[175] J. Slaney. FINDER (Finite Domain Enumerator) : Notes and guide. Technical Report TR-ARP-1/92, Australian National University Automated Reasoning Project, Canberra, 1992.

[176] G. Smolka. The Oz Programming Model. In *Computer Science Today*, volume 1000 of *LNCS*, 1995.

[177] B. Srinivas. *Complexity of lexical descriptions and its relevance for partial parsing.* PhD thesis, University of Pennsylvannia, 1997.

[178] B. Srinivas and A. Joshi. Suppertagging : an approach to almost parsing. *Computational Linguistics*, 1999.

[179] E. Stabler. Avoid the pedestrian's paradox. In *Principle-Based Parsing : Computation and Psycholinguistics.* Kluwer, 1991.

[180] R. Stalnaker. Pragmatic presuppositions. *Semantics and Philosophy*, pages 129–214, 1974.

[181] M. Stede. *Lexical Semantics and Knowledge Representation in Multilingual Generation.* PhD thesis, University of Toronto, 1996.

[182] M. Steedman. Type-raising and directionality in combinatory grammar. In *Proceedings of the 29th ACL*, Berkeley, USA, 1991.

[183] M. Stone. *Modality in Dialogue : Planning, Pragmatics and Computation.* PhD thesis, Department of Computer & Information Science, University of Pennsylvania, 1998.

[184] M. Stone. *Modality in Dialogue : Planning, Pragmatics and Computation.* PhD thesis, Department of Computer & Information Science, University of Pennsylvania, 1998.

[185] M. Stone, C. Doran, B. Webber, T. Bleam, and M. Palmer. Microplanning from communicative intentions : Sentence planning using descriptions (spud). *Submitted to Computational Linguistics*, 2001.

[186] P. Strawson. On referring. *Mind*, 59 :320–344, 1950.

[187] P. Sturt and M. Crocker. Monotonic syntactic processing : a cross linguistic study of attachment and reanalysis. *Language and Cognitive Processes*, 11(5) :449–494, 1996.

[188] C. Tancredi. *Deletion, Deaccenting and Presupposition.* PhD thesis, Massachusetts Institute of Technology, 1992.

[189] K. van Deemter. Generating Referring Expressions : Boolean Extensions of the Incremental Algorithm. *Computational Linguistics*, 28(1) :37–52, 2002.

[190] R. van der Sandt. Presupposition projection as anaphora resolution. *Journal of Semantics*, 9 :333–377, 1992.

[191] J. Veronis and L. Khouri. Etiquettage grammatical multilingue : le projet multex. *TAL*, 1995.

[192] E. M. Vorhees. The trec question answering track. *Natural Language Engineering*, 2001.

[193] B. Webber. *A formal approach to discourse anaphora.* PhD thesis, Harvard University, 1978.

[194] B. Webber, C. Gardent, and J. Bos. Position statement : Inference in question answering. In *Proceedings of the third international conference on language resources and evaluation (LREC)*, Las Palmas, Canary Islands, Spain, 2002.

[195] C. Weidenbach, B. Afshordel, U. Brahm, C. Cohrs, T. Engel, E. Keen, C. Theobalt, and D. Topic. System description : Spass version 1.0.0. In *16th International Conference on Automated Deduction, CADE 16*, pages 314–318, 1999.

[196] P. Whitelock. Shake-and-bake translation. In *Proceedings of COLING '92*, Nantes, France, 1992.

[197] T. Winograd. *Procedures as a representation for data in a computational program for understanding natural language.* PhD thesis, MIT, Cambridge, 1971.

[198] Ludwig Wittgenstein. *Tractatus Logico-Philosophicus.* Annalen der Naturphilosophie, 1921.

[199] F. Xia, C.-H. Han, M. Palmer, and A. Joshi. Automatically extracting and comparing lexicalized grammars for different languages. In *Proc. of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001*, Seattle, Washington, 2001.