Université de Caen - UFR Sciences

THÈSE

présentée pour obtenir le titre de

Docteur de l'Université de Caen

(Arrêté du 30 Mars 1992)

Spécialité **Informatique**

par

Christèle POIRIER

Conception et réalisation d'un analyseur sémantique de documents composites (textes et schémas)

Soutenue le 9 décembre 1996 devant la commission d'examen composée de :

M. Patrice Enjalbert	Professeur à l'Université de Caen	
M. Jacques François	Professeur à l'Université de Nancy II	
M. Daniel Kayser	Professeur à l'Université de Paris-Nord	Rapporteur
M. Gérard Sabah	Directeur de Recherche CNRS au LIMSI	Rapporteur
M. Bernard Victorri	Directeur de Recherche à l'Université de Caen	Directeur
M. Pierre Zweigenbaum	Chercheur à l'AP-HP et Université de Paris VI	

Thèse préparée au Laboratoire d'Informatique de l'Université de CAEN
GREYC - CNRS URA 1526
14032 CAEN CEDEX

CENTRE NATIONAL
DE LA RECHERCHE
SCIENTIFIQUE

Remerciements.

Je tiens à remercier toutes les personnes qui ont contribué à l'aboutissement de ces travaux :

Monsieur Bernard Victorri, Directeur de Recherche CNRS, qui m'a encadrée pendant ces trois années,

Monsieur Patrice Enjalbert qui, par ses conseils, m'a aidée tout au long de cette thèse,

Monsieur le Professeur Daniel Kayser et Monsieur Gérard Sabah, Directeur de Recherche CNRS, qui ont accepté de juger ce travail et d'en être rapporteurs,

Monsieur le Professeur Jacques François et Monsieur Pierre Zweigenbaum, Chercheur à l'AP-HP, pour l'intérêt qu'ils ont porté à ce travail en acceptant d'être membre du jury,

Fabrice Pied pour sa collaboration autant fructueuse que sympathique,

Tous les membres du laboratoire d'informatique qui m'ont aidée et soutenue pendant ces trois ans, mais plus particulièrement et dans le désordre : Jacques Madelaine, Renaud Cazoulat, Loïc Thomazo, Mauro Gaio, Maurice Szmurlo.

Je remercie également le secrétariat d'informatique pour son aide dans les tâches administratives.

Enfin tous ceux que je n'ai pas cité et qui m'ont aidée, soutenue ou conseillée tout au long du déroulement de cette thèse.

Introduction

Le développement des techniques informatiques a permis de mettre à la disposition de toute personne disposant d'un ordinateur un fond de plus en plus important et de plus en plus disparate de documents. Cette profusion de documents a rendu nécessaire le développement de systèmes de recherche de plus en plus performants tant au niveau de la rapidité que de la qualité des résultats rendus. Les systèmes existants, que ce soit la "recherche plein texte" (utilisée par exemple par le logiciel WAIS [1] ou les différents moteurs de recherche permettant d'effectuer des recherches sur Internet) ou la "recherche par mots-clés ou descripteurs" (utilisée en général pour effectuer des recherches dans les banques documentaires) ont plusieurs inconvénients.

En effet, la recherche par mots-clés présente deux inconvénients majeurs : d'une part elle demande un pré-traitement très lourd pour effectuer l'indexation des données (indexation qui est effectuée complètement "manuellement" ou à l'aide de systèmes d'aide à l'indexation [2]), d'autre part il est difficile, voire impossible, de pouvoir décrire le contenu entier d'un document grâce à des mots-clés. Un autre problème, plus annexe, lié à l'interrogation de bases de données, concerne les modalités d'interrogation : la recherche dans une base de données s'effectue en général par l'intermédiaire d'un langage de commandes lié au format de la base de données (nom et format des différents champs, ...) qui est souvent complexe et qui varie suivant la base interrogée. Le développement de systèmes comme "l'anté-serveur" [3] [Victorri *et al.*, 1992] a bien sûr permis de limiter ces problèmes relatifs à la convivialité d'une telle recherche sans pour autant en augmenter les possibilités.

La recherche plein texte, quant à elle, est moins limitée dans ses possibilités que la recherche par mots-clés mais étant donné qu'elle est basée sur la présence, dans le texte, des mots de la requête effectuée et sur leur fréquence d'apparition dans les

1. Wide Area Informations Servers
2. Systèmes improprement appelés le plus souvent "systèmes d'indexation automatique".
3. Logiciel permettant une interrogation en "langage naturel" et en simultané de plusieurs banques documentaires distinctes.

documents, les résultats qu'elle fournit contiennent d'une part beaucoup de "bruit" (documents non pertinents sélectionnés) et d'autre part beaucoup de "silence" (documents pertinents non sélectionnés). Ces mauvais résultats sont essentiellement dus au fait que la recherche plein texte fonctionne uniquement par comparaison de chaînes de caractères : les critères de sélection d'un document sont donc uniquement des critères de forme et ne peuvent tenir compte des phénomènes de synonymie, de paraphrase, d'homographie et de polysémie présents dans les textes. Ces mauvais résultats sont donc une conséquence directe de la non utilisation du sens du document.

La conclusion que l'on peut tirer de ce rapide aperçu des différents systèmes utilisés pour effectuer de la recherche d'informations est que, pour pouvoir classer correctement un ensemble de documents (et donc y effectuer une recherche efficace), il faut s'appuyer non seulement sur la forme des documents mais surtout sur leur contenu. L'idéal serait bien sûr de concevoir des systèmes de compréhension "automatique" du sens des documents capables de traiter tout type de document. Cependant, l'évolution des grands projets de ce type lancés il y a une dizaine d'années montre que cette approche est loin d'être réalisable dans un avenir proche du fait de la nécessité, pour les mettre en place, de devoir disposer de systèmes informatiques de représentation des connaissances encyclopédiques de très grande envergure (systèmes dont l'évolution ne peut que nous rendre pessimistes sur la validité d'une telle approche - voir, par exemple, le projet CYC [Lenat & Guha, 1990]) et de la difficulté à représenter le sens d'un document. Les recherches se sont donc tournées vers des systèmes moins ambitieux mais plus abordables de compréhension limitée de corpus homogènes.

Ces systèmes, en général, rentrent dans le cadre de "l'extraction d'informations" [4] [Cowie & Lehnert, 1996]. Ils ont pour but de ne travailler que sur des ensembles de documents composés uniquement de documents de même forme, réalisés dans les mêmes buts et faisant appel au même domaine (articles scientifiques et techniques, dépêches d'agence de presse, documentations techniques, modes d'emploi, brevets industriels, ...) de façon à réduire la quantité des connaissances à représenter pour pouvoir effectuer une interprétation. Ils ne visent qu'une compréhension partielle du document qui ne représentera qu'une partie du sens des documents et qui sera définie en fonction de l'utilisation que l'on veut en faire : les seules informations considérées comme importantes sont celles qui ont trait au domaine précis auquel appartient le corpus. Dans le cadre de l'extraction d'informations, le travail du système consiste alors

4. En anglais : Information Extraction - IE.

à remplir automatiquement des fiches grâce aux informations extraites du document. Ces fiches pourront ensuite servir à remplir une base de données qui pourra être interrogée. Cette problématique, assez ancienne, a donné lieu à la mise en place de la série des conférences (les conférences MUC[5] [MUC, 1991], [MUC, 1992]). Ces conférences ont pour but de permettre une évaluation et une comparaison des différents systèmes d'extraction d'informations. On peut constater que la plupart des systèmes, tout en divergeant sur les mises en œuvre techniques, mettent en place les mêmes modules à savoir : un module effectuant un pré-traitement au niveau du texte ayant pour but de ne conserver que les segments pertinents, un module effectuant un traitement linguistique constitué d'une analyse syntaxique souvent partielle et d'une analyse sémantique plus ou moins poussée et enfin un module effectuant la mise en forme du résultat c'est-à-dire le remplissage des fiches. L'analyse sémantique effectuée a pour but d'obtenir une représentation sémantique canonique, de résoudre les problèmes d'inférence et de coréférence et de déterminer les liens entre entités et événements.

C'est dans le cadre de ce type de système que ce situe le projet développé au sein de nos deux laboratoires (le GREYC[6] et l'ELSAP[7]) et dont cette thèse constitue une étape. L'intérêt de notre système par rapport aux systèmes actuels d'extraction d'informations se situe non seulement dans le fait que l'analyse sémantique que nous effectuons est plus poussée que celle effectuée dans la plupart des systèmes de ce type, mais aussi dans le fait que nous ajoutons aux étapes décrites ci-dessus une étape complète de construction de la scène décrite dans le document nous permettant une meilleure compréhension de ce dernier et donc une plus grande efficacité. Notre projet consiste donc à effectuer une compréhension limitée de corpus homogènes ayant pour but, non pas de remplir uniquement des fiches, mais aussi de construire de façon précise la scène décrite par le document de façon à pouvoir répondre avec précision à des questions sur le contenu du document.

Il existe de nombreux domaines où un tel système peut être utilisable. Une des applications possibles concerne les comptes-rendus médicaux. Un projet de grande envergure a d'ailleurs été développé sur ce type de corpus. Il s'agit de MENELAS [MENELAS, 1995]. Ce projet, qui regroupe huit partenaires européens (Français, An-

5. Message Understanding Conference.

6. Groupe de Recherche en Informatique, Image et Instrumentation de Caen - UA 1526 du CNRS et Université de Caen - 14032 Caen Cedex.

7. Étude de la Linguistique de la Signification à travers l'Ambiguïté et la Paraphrase - UA 1234 du CNRS et Université de Caen - 14032 Caen Cedex.

glais et Néerlandais) alliant des compétences en informatique médicale et en linguistique, a pour objectifs, d'une part, d'effectuer une indexation des documents, d'autre part, de permettre une interrogation en langage naturel de la base ainsi créée. Un autre projet du même type (création d'une base de données puis interrogation en langage naturel de cette dernière) a été mis en place dans les années 80 mais dans un tout autre domaine : le projet LILOG[8]. Ce projet, lancé en 1985, a conduit la mise en place d'un système d'informations touristiques sur la ville de Düsseldorf (Allemagne) dont les objectifs étaient de lire les brochures touristiques de la ville, de les intégrer dans une base de données et de dialoguer en allemand avec un utilisateur pour le renseigner sur les activités touristiques de la ville : donner non seulement les horaires d'ouverture du musée mais indiquer aussi comment y aller. Ce dernier point nécessite, de la part du système, une compréhension en profondeur des textes et des questions qui lui sont soumises.

Il existe aussi de nombreux corpus homogènes où les documents ne sont pas composés uniquement de textes mais aussi d'images ou de schémas et où le contenu de ces derniers est aussi important que celui du texte dans la compréhension du document (articles scientifiques et techniques, brevets industriels, documentations techniques et modes d'emploi, etc.). Cette caractéristique nous a amenés à étudier non seulement l'aspect linguistique des documents mais aussi leur aspect "imagier". De même qu'au niveau du texte nous considérons qu'il est utopique de vouloir développer un système général capable de "comprendre" toute sorte de documents, nous estimons qu'il est plus réaliste de concevoir un système de "compréhension limitée" des images (au sens large : images ou schémas) présentes dans un document. Nous avons donc décidé de concevoir et de réaliser un système d'analyse de documents composites (documents contenant à la fois du texte et des images) basé sur une compréhension limitée de corpus homogènes utilisant à la fois le texte et l'image et dont le but principal est d'effectuer une construction de la scène décrite par l'ensemble du document.

La conception et la réalisation d'un tel système est un travail de longue haleine qui doit s'effectuer par étapes, de façon à pouvoir, à chaque étape, évaluer les résultats obtenus, valider ou invalider les différentes hypothèses sur lesquelles repose ce travail, et définir en fonction de ces réflexions le contenu précis des étapes suivantes.

La première étape consiste à réaliser une première maquette d'une application dans son intégralité de façon à pouvoir valider la faisabilité d'un tel système. Cela nécessite

8. LInguistic and LOGic methods and tools.

de choisir une application présentant un degré suffisant de difficulté, c'est-à-dire dans laquelle le maximum de problèmes que posent un tel système apparaissent effectivement. En particulier, il est important que toute la chaîne de traitement du document soit mise en place. Ceci étant, il s'agit d'une maquette, ce qui signifie qu'on ne prétend pas y avoir résolu de manière satisfaisante tous les problèmes que l'on fait apparaître. En particulier, il ne sert à rien de vouloir évaluer des performances à ce stade. Ce que l'on demande à la maquette, c'est de traiter un petit nombre de documents représentatifs du corpus : l'objectif est d'abord d'évaluer la faisabilité du système avant de se lancer, dans les étapes suivantes, dans des développements plus considérables.

Afin d'effectuer l'implémentation d'une première maquette, il nous fallait déterminer, parmi les corpus homogènes composés à la fois de textes et d'images, celui qui servirait le mieux nos objectifs : il nous fallait un corpus nous permettant de tester les différentes composantes du système. Pour cela, nous avons choisi un corpus constitué de témoignages d'accidents de la route que les assurés envoient à leur compagnie d'assurance (les documents constituant notre corpus nous ont été aimablement fournis par la MAIF[9]). Les documents constituant un tel corpus sont composés d'un texte (se trouvant au dos du constat) ainsi que d'un schéma (situé soit au recto, soit au verso du constat). Les avantages de ce corpus résident non seulement dans les caractéristiques des textes (ce sont des textes courts mais suffisamment riches pour permettre des analyses et des inférences réelles tout en faisant appel aux connaissances d'un monde relativement clos, le "monde de la route") mais aussi dans l'importance des liens existant entre le texte et le schéma.

Cette thèse a donc consisté d'une part à participer à l'élaboration théorique de l'architecture (dont la principale originalité est de répartir le travail d'interprétation du document en trois sous-systèmes : analyse du schéma, analyse du texte et construction d'une représentation de la scène à partir des données fournies par les deux analyseurs et en s'appuyant sur une description du domaine lié au document et sur les conditions dans lesquelles il a été réalisé), d'autre part à implémenter l'ensemble de l'architecture ainsi qu'une partie de deux des sous-systèmes qui la composent (analyse du schéma et analyse du texte). Comme nous l'avons expliqué dans notre description de la première étape de constitution d'un tel système, l'implémentation réalisée a pour but, non pas de permettre une évaluation des résultats du système, mais d'évaluer sa faisabilité et de déterminer les problèmes existants de façon à donner une base pour des développe-

9. Mutuelle d'Assurance des Instituteurs de France.

ments plus considérables. Cette thèse a donc consisté à mettre en place un atelier de réalisation permettant par la suite d'améliorer et de terminer la réalisation du système d'interprétation de documents composites.

Le mémoire de cette thèse est organisé en quatre parties présentant les différents aspects de mon travail à savoir : principes de l'architecture de l'ensemble du système, analyseur de schémas, analyseur linguistique et présentation de l'implémentation de l'architecture globale à travers la mise en place des communications entre les trois sous-systèmes.

Dans la première partie, nous présenterons l'ensemble du projet. Nous décrirons d'abord le projet initial de traitement automatique de textes et le cadre lié à ce projet avec, entre autres, une présentation très succincte des diverses équipes ayant travaillé ou travaillant sur ce sujet (interprétation des textes de constats d'accidents). Dans un deuxième temps, nous indiquerons les raisons qui nous ont poussés à prendre en compte non seulement le texte mais aussi le schéma et donc à passer d'un système d'interprétation automatique de textes à un système d'interprétation de documents composites (documents composés d'un texte et d'un schéma). Nous présenterons ensuite quelques travaux relatifs à l'interprétation automatique de documents composites ainsi que divers cadres d'application pour un tel système. Nous conclurons cette partie par une présentation en détail de l'architecture que nous proposons.

La deuxième partie de ce mémoire est entièrement consacrée à la présentation de l'analyseur de schémas mis en place dans le cadre de cette thèse. Une première maquette de cet analyseur a été effectuée lors de mon stage de DEA [Poirier, 1993] et le travail effectué lors de ma thèse dans ce domaine a essentiellement consisté à améliorer cette maquette et à intégrer l'analyseur obtenu au système global. Cette partie débute par une présentation des différents types de schémas ayant fait l'objet de travaux de recherche et des particularités des schémas de constats d'accidents par rapport à ces derniers. Nous indiquerons ensuite comment ces particularités nous ont conduits à mettre en place une architecture modulaire puis nous détaillerons l'ensemble des modules ainsi que les choix effectués pour l'implémentation. Nous conclurons cette partie sur les résultats obtenus par cet analyseur ainsi que sur les améliorations qu'il serait possible de lui apporter.

La troisième partie de ce mémoire, quant à elle, porte entièrement sur la conception et la réalisation de l'analyseur linguistique. Nous verrons tout d'abord les particularités de l'analyseur linguistique du fait de l'architecture choisie pour l'ensemble du projet ainsi que les principes mis en place dans le cadre d'une coopération entre analyseur linguistique et modèle du monde et nous verrons comment cette coopération nous permet d'effectuer une analyse du texte “en contexte”. Après une présentation de diverses architectures modulaires existantes, nous indiquerons les choix effectués pour l'architecture de l'analyseur linguistique et son implémentation. Nous détaillerons ensuite les deux modules de cet analyseur que nous avons réalisés en donnant, pour chacun d'eux, son principe de fonctionnement et les choix effectués pour son implémentation. Nous conclurons cette partie par la présentation d'une analyse détaillée et des résultats obtenus dans l'état actuel de réalisation.

Si schéma et texte sont étroitement liés, il peut cependant, au terme de l'analyse des deux parties du document, subsister un certain nombre de différences (incohérences) entre le résultat des deux analyses. Du fait de l'architecture mise en place pour effectuer l'interprétation du document, seul le sous-système chargé d'effectuer la représentation de la scène est à même de s'apercevoir de ces incohérences. La résolution de ces dernières nécessite donc la mise en place de coopérations entre les analyseurs et le sous-système chargé d'effectuer la représentation de la scène. C'est ce principe de coopération ainsi que ses conséquences sur les communications entre les différents sous-systèmes et sur l'implémentation du système que nous présenterons dans la dernière partie de ce mémoire.

Cette quatrième partie débute par un rappel du fonctionnement global du système et par une présentation des différents types d'incohérences que nous avons répertoriés et de la solution que nous proposons pour les résoudre. Nous présenterons ensuite les choix effectués au niveau de l'implémentation et des communications. Nous conclurons cette dernière partie par une description des communications ayant lieu dans le cadre d'une analyse sans incohérence puis dans le cadre d'analyses avec incohérences (une analyse par grand type d'incohérence répertorié précédemment).

Première partie

L'ensemble du projet

Le projet de base (fruit d'une collaboration entre le GREYC et l'ELSAP) concernait uniquement l'interprétation automatique de textes (textes issus d'un corpus de textes de constats amiables d'accidents automobiles). Cependant, dans un document comme le constat amiable (document composé à la fois d'un texte et d'un schéma), l'analyse du seul texte ne suffit pas pour comprendre la totalité du document. En effet, dans ce type de document, schéma et texte sont étroitement liés et apportent à la construction de la scène des informations de natures différentes mais complémentaires (le texte apporte des informations dynamiques sur le déroulement de l'accident, le schéma apporte des informations statiques sur la disposition particulière des lieux et la position des véhicules au moment de l'accident). Cette constatation nous a amenés à mettre en place non pas un système d'interprétation de textes mais un système réalisant à la fois l'interprétation du texte et celle du schéma : un système d'interprétation de documents composites.

Ce sont les objectifs de ce dernier système ainsi que les choix effectués pour l'architecture d'un tel système que nous détaillerons dans cette partie.

Chapitre 1

Vers un système d'interprétation de documents composites

1.1 Le projet d'interprétation de textes

La réalisation de notre système a débuté dans le cadre d'un projet du groupe de travail "Sémantique des Langues Naturelles" portant sur la compréhension automatique de textes (projet ayant reçu le soutien des PRC Communication Homme-Machine et Intelligence Artificielle [Abraham *et al.*, 1992] et [Abraham *et al.*, 1993]). Ce groupe de travail est actuellement constitué d'une dizaine d'équipes de recherche composées aussi bien d'informaticiens que de linguistes qui, comme nous le verrons par le suite, ont abordé le problème de la compréhension de textes de diverses manières en privilégiant certains aspects des textes étudiés [TAL, 1994]. Un des premiers objectifs de ce groupe de travail a donc été de définir un cadre général permettant de confronter les différentes approches. Ce cadre de travail concernait l'application à réaliser et définissait d'une part un corpus commun, d'autre part la tâche à effectuer sur les textes du corpus (compréhension des textes qui pourra être jugée, par exemple sur la capacité du système à répondre à des questions les concernant). L'objectif de ce projet était donc l'étude de la faisabilité d'une telle application et un des critères de comparaison des différentes approches était une réalisation informatique ayant pour but de valider les divers choix.

Le corpus qui a été choisi par le groupe de travail est un corpus de récits d'accidents de la circulation. Ces textes sont des comptes-rendus d'accidents automobiles se trouvant au verso des constats amiables d'accidents automobiles et rédigés a posteriori par un des conducteurs. Ils décrivent les circonstances de l'accident et viennent en complément des indications portées au recto. Ces textes ont été choisis pour plusieurs raisons. D'une part ils existent en grand nombre et sont facilement accessibles. D'autre part ils sont suffisamment riches pour permettre des analyses et des inférences réelles mais de taille suffisamment restreinte pour envisager un traitement automatique. De plus, ils font appel aux connaissances d'un monde relativement clos qui est "le monde de la route". Le corpus que nous avons utilisé est composé de 87 textes (textes fournis par la MAIF - cf. Annexe B.1) dont 17 (les textes de la série A) ont été plus particulièrement étudiés.

Du fait de la nature des textes, différents niveaux de compréhension sont envisageables. Parmi ces différents niveaux se dégagent deux courants principaux : une compréhension factuelle qui consiste à extraire des textes ce qui s'est passé (les objets intervenant dans la scène, leurs relations spatiales, les différents événements, leur ordre de succession ...) ou une compréhension plus "pragmatique" dans laquelle l'aspect argumentatif des textes, dont le but principal est de se disculper vis-à-vis de l'assurance, est pris en compte. Au sein de ce groupe de travail, c'est au premier niveau qu'a été consacrée la majorité des efforts mais le second niveau a cependant été exploré par l'équipe du LIPN [10] et Dominique Estival.

Dominique Estival de l'ISSCO [11] et Françoise Gayral du LIPN présentent dans [Estival & Gayral, 1994] les différentes caractéristiques des textes de constats d'accidents (textes courts et narratifs introduisant au moins deux participants et dans lesquels l'auteur tente de se justifier) ainsi que l'influence de ces caractéristiques sur la rédaction et l'interprétation de ces textes. Elles dégagent entre autres les différentes formes que peut prendre l'argumentation dans ce genre de texte et l'apport de l'aspect argumentatif dans l'interprétation. Leur travail représente une bonne introduction aux problèmes que posent ces textes.

J. Francois [12] (en collaboration avec D. Coulon du CRIN [13]) a étudié, d'un point

10. Laboratoire d'Informatique de Paris-Nord - UA 1507 du CNRS et Université de Paris-Nord, Institut Galilée - 93 430 Villetaneuse.
11. International Society for the Study of Comparative Oncology, Université de Genève, 54 route des Acacias, CH-1227 Genève.
12. Université Nancy 2 - UFR de Lettres - 23, bld Albert 1er, BP 3397, 54 015 Nancy cedex.
13. Centre de Recherche en Informatique de Nancy - UA 262 et Universités de Nancy - 54 506

de vue linguistique pur, l'ensemble des textes du corpus sur plusieurs problèmes linguistiques ([François, 1996] et [Coulon & François, 1996]) et en particulier celui des métonymies du sujet dans ce style de texte. Pour mener à bien cette étude, J. François a construite une base de données (disponible sous Excel) dans laquelle figurent, pour chacun des textes du corpus, le découpage en propositions et pour chacun des niveaux de profondeur syntaxique, le connecteur de chacune des propositions ainsi que son sujet et son verbe. Pour chacun des verbes du texte est indiqué la compatibilité entre sujet et verbe ce qui permet de repérer les métonymies du sujet. Cette base de données a été amplement utilisée par l'ensemble des équipes travaillant sur les constats d'accidents.

L'approche choisie par l'équipe du CAMS [14] (animée par J.P. Desclés), prend ses sources dans le modèle global de la Grammaire Applicative et Cognitive [Descles, 1990b] et est basée sur la définition et la mise en place de schèmes représentant la signification d'un prédicat en fonction de son contexte (les *schèmes sémantico-cognitifs* [Descles, 1990a]). Ces schèmes permettent, entre autres, de représenter, dans le lexique, les différentes significations d'un même verbe et par conséquent de rendre compte des effets polysémiques. L'hypothèse principale de cette équipe est que les différents sens d'un verbe se dérivent d'un sens "premier" en fonction du contexte. Le travail de cette équipe a porté non seulement sur les verbes de mouvement et d'activité (représentation des verbes de mouvement et d'activité au travers de *schèmes sémantico-cognitifs* [Abraham, 1995] et [Abraham, 1994]) mais aussi sur le traitement automatique du temps et de l'aspect présenté à travers un exemple dans [Desclés *et al.*, 1994].

M. Aurnague de l'ERSS [15] et P. Sablayrolles de l'IRIT [16] ont essentiellement étudié l'aspect spatio-temporel de la langue et plus spécialement les marqueurs du mouvement. Dans [Aurnague *et al.*, 1994], ils présentent avec J. Jayez [17] un travail commun sur la représentation du contenu sémantique d'un certain nombre d'expressions spatiales et spatio-temporelles dans le domaine des constats d'accidents. Dans cet article, ils présentent tout d'abord les résultats d'une étude de diverses expressions de mouvement dans le cadre du corpus de constats d'accidents de la route. Ils présentent ensuite les grandes lignes d'une architecture basée sur un découpage du texte en épisodes inspiré

Vandoeuvre les Nancy Cedex.

14. Centre d'Analyse et de Mathématiques sociales, 54 et 96, bld Raspail, 75 006 Paris.

15. Équipe de Recherche en Syntaxe et Sémantique - CNRS UA 1033, Université de Toulouse-Le Mirail, Maison de la recherche, 5 allée Antonio Machado, 31 058 Toulouse.

16. Institut de Recherche en Informatique de Toulouse - CNRS UA 1399, Université Paul Sabatier 118, rue de Narbonne 31 062 Toulouse.

17. CEntre de LInguistique THéorique (CELITH), 54 bld Raspail, 75 006 Paris.

de [Hwang & Schubert, 1992].

Dans [Gayral *et al.*, 1994], l'équipe du LIPN présente comment l'analyse du problème de l'interprétation automatique de textes de constats d'accidents les a conduits à la notion de "mondes alternatifs" (notion à rapprocher de la sémantique des mondes possibles de [Kripke, 1963], de la notion de chronique de [McDermott, 1982] ou encore des espaces mentaux de [Fauconnier, 1984] et des univers de croyance de [Martin, 1983]) et à l'étude de l'aspect argumentatif des textes. Cette notion est venue du besoin de représenter, en plus des événements ayant eu réellement lieu, l'ensemble des situations mentionnées dans le texte mais qui ne se sont pas réalisées mais qui cependant permettent de rendre compte de l'articulation des différents segments du texte et de l'avancée du récit. Les mondes alternatifs permettent donc de représenter, au fur et à mesure de la lecture du texte, les diverses situations envisagées, attendues, voulues ou niées et mentionnées dans les textes par l'intermédiaire de marqueurs comme : "*voulant dépasser un semi-remorque*", "*j'ai voulu m'engager*", "*je ne m'attendais pas à*" ... Un raisonnement par comparaison entre le monde réel et ces mondes alternatifs permet d'effectuer des raisonnements plus fins, en particulier aux niveaux argumentatif et causal, en expliquant des relations de causalité (réelles ou supposées) dans l'enchaînement des actions du narrateur et des autres acteurs.

L'objectif de l'application envisagée par les équipes du GREYC et de l'ELSAP dans le cadre de ce groupe de travail se situait au niveau de la compréhension factuelle des textes [Enjalbert & Victorri, 1994] et plus spécialement au niveau de l'étude et de la représentation des actions de conduite et de localisation. Nous n'avons pas voulu prendre en compte, dans un premier temps, l'aspect argumentatif des textes, notre but étant essentiellement de construire une représentation cohérente et opératoire de la scène décrite. De plus, nous souhaitions pouvoir vérifier l'interprétation effectuée non seulement par l'intermédiaire de questions sur le texte mais aussi par une visualisation (si possible animée) de la scène construite.

Lors de l'étude portant sur la réalisation d'une telle application, nous nous sommes aperçus que, dans bien des cas, le texte ne permettait pas à lui seul de construire à coup sûr une représentation qui soit satisfaisante. Afin d'illustrer notre propos, nous donnons maintenant un certain nombre d'exemples représentatifs des problèmes liés au fait de n'interpréter que le texte.

Prenons, tout d'abord, le texte suivant :

Revenant à mon domicile, le chauffeur du véhicule B me précédant a perdu le contrôle de son véhicule à cause d'un brutal verglas. Je n'ai pas pu à mon tour, contrôler mon véhicule qui après 20 mètres a percuté le véhicule de Mme Louvet. Je tiens à signaler qu'il n'y avait pas de verglas ailleurs et que nous avons été beaucoup de véhicules à déraper sur cette rue. Rien ne pouvait laisser prévoir 1 tel verglas. [Texte B28 du corpus]

Dans ce texte, les données linguistiques ne permettent pas de déterminer précisément le nombre de véhicules mis en cause car rien ne permet de savoir si les termes *véhicule B* et *le véhicule de Mme Louvet* font référence à un même véhicule.

Regardons ensuite, le texte suivant :

Je roulais sur la partie droite de la chaussée quand un véhicule arrivant en face dans le virage a été complètement déporté. Serrant à droite au maximum, je n'ai pu éviter la voiture qui arrivait à grande vitesse. [Texte A8 du corpus]

Dans ce cas, le problème diffère légèrement du précédent. En effet, à la lecture de ce texte, il semble logique de ne considérer que deux véhicules (1 lié au locuteur, 1 autre référencé à la fois par *la voiture* et *un véhicule*). Cependant, les données linguistiques n'excluent pas totalement la présence de trois véhicules. En effet, on peut très bien considérer que les expressions *la voiture* et *un véhicule* font référence à deux véhicules différents.

Ces deux exemples montrent des problèmes concernant le nombre de véhicules mis en cause. Il existe aussi, dans un certain nombre de textes, des problèmes de localisation ou de positionnement d'un véhicule par rapport à un autre. Dans le cas du texte qui suit par exemple, il n'y a aucune indication sur la position du véhicule A par rapport à celle du véhicule B qui, a priori est conduit par le locuteur.

Je roulais à droite avenue de Libération en direction de la rue René Coty où je devais me rendre. Arrivant à cette rue, qui prend à gauche, je me suis rangé à droite le plus possible pour m'assurer qu'aucun véhicule n'arrivait. Je n'ai vu aucun véhicule proche ni devant ni derrière, et j'ai commencé à tourner à gauche. Ai-je mal apprécié la vitesse du véhicule A? Toujours est-il que la conductrice a freiné dans les dernières secondes, mais n'a pas pu éviter le choc. J'avais indiqué mon changement de direction. [Texte B54 du corpus]

De plus, il s'est avéré que la plupart des textes sont difficiles à comprendre pour un lecteur ne connaissant pas les lieux (on peut en voir un exemple dans le cas du texte suivant). En effet, la description de ces derniers est le plus souvent faite grâce à des noms de rue ce qui rend toute interprétation difficile à qui est ignorant de la disposition particulière des lieux.

Circulant Bd Couturier, le véhicule B était stationné sur ce même Boulevard quand celui-ci a effectué une marche arrière dans la rue Constantinide, puis a redémarré brusquement vers le bd Couturier pour tourner à gauche et m'a percuté au moment où j'arrivais sur le carrefour. Le véhicule B arrivait de la partie gauche de sa chaussée et a coupé sa manœuvre pour tourner à gauche. [Texte B24 du corpus]

Dans le cas des documents que nous étudions, il existe normalement non seulement un texte mais aussi un schéma (cf. figure 1.1) qui pourrait nous aider à résoudre les différents problèmes mentionnés ci-dessus. Dans un constat amiable, le texte permet au conducteur du véhicule de préciser les circonstances de l'accident (nombre de véhicules en cause, mouvements des véhicules ...), le schéma lui permet d'indiquer la disposition particulière des lieux ainsi que la position des véhicules au moment de l'accident. Le texte donne donc des indications dynamiques sur le déroulement de l'accident alors que le schéma donne des informations statiques (accompagnées quelquefois d'informations "pseudo-dynamiques" par l'intermédiaire de flèches indiquant les déplacements). De plus, certains textes font des références explicites au schéma (*En me rapprochant de l'intersection représentée sur le schéma*, extrait du texte B29 du corpus; *(prière de voir croquis p.1!)* extrait du texte B57 du corpus) montrant bien que ces deux parties du constat, bien que de nature différente, sont étroitement liées.

Pour toutes ces raisons, nous avons choisi d'effectuer la construction de la scène en utilisant non seulement les données fournies par le texte mais aussi celles fournies par le schéma. Le cadre de cette interprétation se situe alors, non plus dans celui d'une simple interprétation du texte, mais plutôt dans celui d'une "interprétation de documents composites" (documents composés à la fois d'une partie texte et d'une partie graphique).

Nous nous sommes cependant trouvés confrontés à un problème concernant ces schémas : même si le constat comporte toujours un schéma, celui-ci peut se trouver uniquement au recto du constat. Ne disposant uniquement que des photocopies du verso des constats nous ne disposions que de très peu de schémas. Pour pouvoir tester notre architecture et mettre au point l'analyseur de schémas nous avons donc dû nous procurer un certain nombre de schémas. Dans un premier temps, nous avons nous même réalisé les schémas correspondant aux textes que nous souhaitions utiliser. Dans un second temps, nous avons demandé à un certain nombre de personnes (les membres de notre laboratoire) de réaliser les schémas correspondant à quatre textes que nous souhaitions étudier. L'avantage de cette seconde formule est de disposer de plusieurs schémas possibles pour un même texte et de pouvoir étudier les différences de représentation.

Les documents que nous avons utilisés ont été sélectionnés essentiellement en fonction des textes c'est-à-dire en fonction de la possibilité qu'un lecteur arrive à reconstituer ce qui s'est passé. Un critère de sélection tel que la clarté du schéma ou la complémentarité texte-schéma nous aurait sûrement amené à en choisir d'autres.

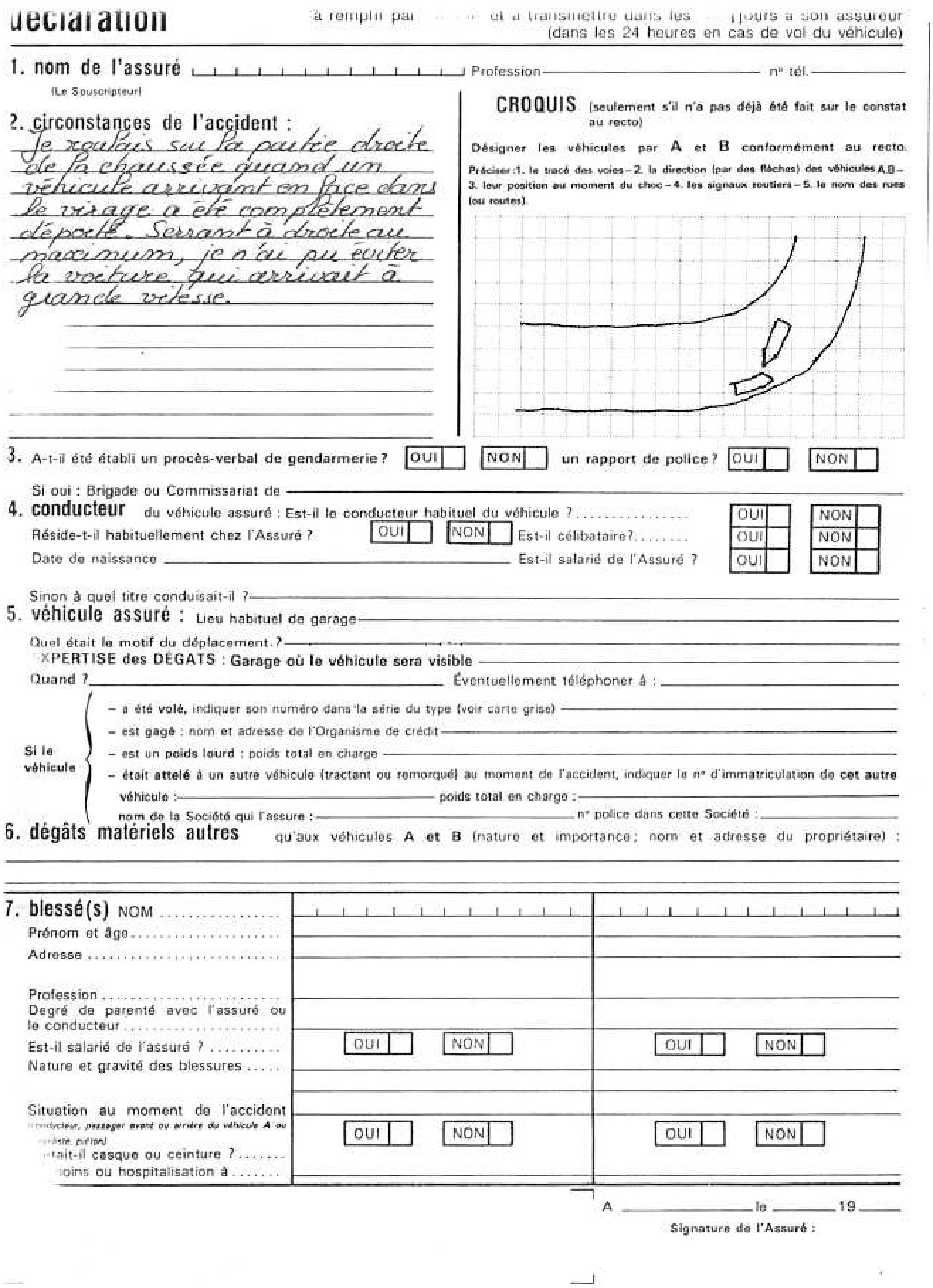
1.2 Le projet d'interprétation de documents composites

L'interprétation de documents composites que nous effectuons consiste à interpréter simultanément les deux éléments du document (le texte et le schéma) et à intégrer informations textuelles et informations graphiques pour construire une représentation de la scène décrite à la fois par le texte et le schéma. Elle est à distinguer de "l'analyse de documents composites" effectuée par des systèmes comme Herode[18] [Krönert *et al.*, 1987] ou le projet des équipes Movi et Read du CRIN [Belaïd & Tombre, 1992]. En effet, l'analyse de documents composites telle qu'elle est effectuée dans ces différents systèmes, consiste à reconnaître automatiquement les différentes parties composant un document papier dans le but de les rendre utilisables sur support informatique. Cette opération consiste en général à extraire la structure logique du document (c'est-à-dire à organiser le document en une hiérarchie d'objets logiques tels que résumé, titre, chapitre, paragraphe, figure, table, etc.) ainsi que sa structure physique (c'est-à-dire déterminer les objets physiques comme les pages, les colonnes, etc.). L'analyse de documents composites est donc essentiellement située dans le domaine de l'analyse d'images et, en général, ne fait appel à la linguistique que pour des problèmes de reconnaissance de caractères (la linguistique étant le plus souvent limitée à de la morphologie).

18. Handling the Electronic Representation of Office Documents based on ECMA 101 standard - projet développé dans le cadre d'un programme de recherche Esprit.

L'application que nous développons a pour objectif d'interpréter les deux composantes du document sans se soucier de les extraire de ce dernier, c'est-à-dire sans avoir à reconnaître la structure du document pour en extraire les parties textes et les parties graphiques et sans avoir à effectuer de reconnaissance de caractères sur les parties textes comme le présuppose la mise en place d'une architecture permettant de passer directement du document papier à une interprétation de ce dernier. Notre application vise donc à construire une représentation cohérente et opératoire de la scène décrite à partir de l'analyse de deux "objets" distincts : d'une part le résultat de la numérisation du schéma (et non du document entier), d'autre part le texte donné sous forme de fichier texte ne nécessitant aucune reconnaissance graphique (pas de reconnaissance de caractères).

Cette double interprétation nous permettra non seulement de compléter et d'affiner notre interprétation de la scène par rapport à celle effectuée à partir du texte seul mais aidera aussi le système lors de la visualisation de la scène construite (objectif secondaire du système visant à représenter le déroulement temporel de l'accident sur un écran). En effet, le schéma tout en représentant l'étape finale de la visualisation (il correspond normalement à la situation lors de l'accident), peut être un point de départ pour la mise en place de la visualisation : il indique non seulement la situation lors de l'accident mais apporte aussi des informations sur la disposition particulière des lieux.


déclaration à remplir par ... et à transmettre dans les ... jours à son assureur (dans les 24 heures en cas de vol du véhicule)

1. nom de l'assuré ______ Profession ______ n° tél. ______
(Le Souscripteur)

2. circonstances de l'accident :
Je roulais sur la partie droite de la chaussée quand un véhicule arrivant en face dans le virage a été complètement déporté. Serrant à droite au maximum, je n'ai pu éviter la voiture qui arrivait à grande vitesse.

CROQUIS (seulement s'il n'a pas déjà été fait sur le constat au recto)
Désigner les véhicules par A et B conformément au recto.
Préciser : 1. le tracé des voies – 2. la direction (par des flèches) des véhicules A B – 3. leur position au moment du choc – 4. les signaux routiers – 5. le nom des rues (ou routes).

3. A-t-il été établi un procès-verbal de gendarmerie ? OUI NON un rapport de police ? OUI NON
Si oui : Brigade ou Commissariat de ______

4. conducteur du véhicule assuré : Est-il le conducteur habituel du véhicule ? .... OUI NON
Réside-t-il habituellement chez l'Assuré ? OUI NON Est-il célibataire ? .... OUI NON
Date de naissance ______ Est-il salarié de l'Assuré ? OUI NON
Sinon à quel titre conduisait-il ? ______

5. véhicule assuré : Lieu habituel de garage ______
Quel était le motif du déplacement ? ______
EXPERTISE des DÉGÂTS : Garage où le véhicule sera visible ______
Quand ? ______ Éventuellement téléphoner à : ______
Si le véhicule
– a été volé, indiquer son numéro dans la série du type (voir carte grise) ______
– est gagé : nom et adresse de l'Organisme de crédit ______
– est un poids lourd : poids total en charge ______
– était attelé à un autre véhicule (tractant ou remorqué) au moment de l'accident, indiquer le n° d'immatriculation de cet autre véhicule : ______ poids total en charge : ______
nom de la Société qui l'assure : ______ n° police dans cette Société : ______

6. dégâts matériels autres qu'aux véhicules A et B (nature et importance ; nom et adresse du propriétaire) :

7. blessé(s) NOM ....
Prénom et âge ....
Adresse ....
Profession ....
Degré de parenté avec l'assuré ou le conducteur ....
Est-il salarié de l'assuré ? .... OUI NON OUI NON
Nature et gravité des blessures ....
Situation au moment de l'accident (conducteur, passager avant ou arrière du véhicule A ou B, cycliste, piéton)
Portait-il casque ou ceinture ? .... OUI NON OUI NON
Soins ou hospitalisation à ....

A ______ le ______ 19 ______
Signature de l'Assuré :

FIG. 1.1 - *Verso d'un constat amiable*

Chapitre 2

Le système d'interprétation de documents composites

Dans ce chapitre, nous présenterons tout d'abord quelques applications pour un tel système ainsi que des systèmes déjà développés. Nous détaillerons ensuite l'architecture que nous envisageons en exposant tout d'abord les principes et ensuite, les caractéristiques des différents composants. Nous présenterons, enfin, divers systèmes et théories basés sur des principes similaires.

2.1 Les applications

L'interprétation de documents composites telle que nous l'avons définie précédemment peut être utilisée dans différents domaines comme l'interprétation de récits plus ou moins longs auxquels sont associées une ou plusieurs images (c'est le cas du système que nous avons développé ou de celui développé par [Abe *et al.*, 1981]), l'interprétation d'images avec leur légende (comme le réalise PICTION), l'interprétation de descriptions d'itinéraires, l'extraction d'informations à partir de documents multimédia, etc.

Le système de [Abe *et al.*, 1981] a pour but d'effectuer l'interprétation de récits décrits à la fois par un texte et par des images. Les images utilisées sont des images simples en couleur (images semblables à celles d'un dessin animé très simplifié); le texte est un ensemble de phrases simples qui sont associées à l'une ou l'autre des images. Ce système utilise les textes et les images pour construire un modèle du monde décrit par l'histoire qui pourra ensuite être interrogé.

Ce système est composé de deux modules : un module de traitement automatique du langage et un module de traitement automatique de l'image qui se partagent une même base de connaissances sous forme de réseau sémantique. Le module de traitement automatique de l'image dispose en plus d'un modèle des objets à reconnaître mentionnant, pour chacun des objets (et des parties d'un objet), sa couleur et sa taille. Dans ce système, les deux modules ne travaillent pas indépendamment mais interagissent de façon à tirer le maximum des déductions effectuées par chacun des systèmes : les informations extraites du texte sont initialement utilisées pour contraindre l'interprétation de l'image (contraindre la zone de recherche d'objets dans le texte) et celles extraites de l'image sont utilisées pour lever un certain nombre d'ambiguïtés dans l'interprétation du texte. L'inconvénient dans ce système est que si un objet est présent dans une image mais non mentionné dans le texte lui correspondant alors il ne sera pas détecté par le module de traitement automatique de l'image.

D'autres systèmes comme PICTION[19] développé au CEDAR[20] utilisent la légende d'une image pour effectuer l'interprétation de celle-ci ([Srihari & Burhans, 1994] et [Srihari *et al.*, 1994]). Dans PICTION, la légende est utilisée pour identifier des personnes sur une photographie de groupe. Ce système est capable, à partir d'une photographie digitalisée et du texte de la légende correspondante, de localiser, d'identifier et de donner des informations portant sur les objets (ou personnages) présents sur la photographie. PICTION utilise pour cela un système de repérage des visages capable de localiser les zones de l'image susceptibles de correspondre à un visage. Les contraintes issues de la légende sont en général suffisantes pour éliminer les zones incorrectes extraites par le système de repérage et pour effectuer une identification des visages.

Ce système est composé de trois modules principaux : un module de traitement automatique du langage, un module de traitement automatique de l'image et un module de contrôle. Ce système utilise une base de connaissances (donnée sous forme de réseau sémantique qui est utilisé à la fois par le module linguistique et par le module d'imagerie) regroupant à la fois des connaissances linguistiques et perceptives. Le module linguistique prend en entrée le texte de la légende et produit une représentation qui inclut à la fois des informations factuelles et une description, sous forme de contraintes (positionnement d'un visage par rapport à un autre, "typage" des visages -

19. PICture and capTION.
20. Center of Excellence for Document Analysis and Recognition - Buffalo, NY 14228-2567 USA.

homme/femme, etc.), de l'image. Le module de contrôle utilise ces diverses contraintes portant sur l'image pour diriger le module d'imagerie dans son interprétation et son étiquetage des zones susceptibles de correspondre à un visage.

Ces deux systèmes utilisent les données extraites par un des modules du système pour contraindre et limiter l'analyse effectuée par le second composant. Dans le système de [Abe *et al.*, 1981], le texte est utilisé pour contraindre les zones de recherche d'un objet et le schéma aide à désambiguïser le texte. Dans PICTION, le texte est utilisé pour diriger le module d'analyse d'images dans son interprétation des zones détectées par le système de repérage des visages et pour aider à la détection des zones qui ne correspondent pas à des visages. L'inconvénient de tels systèmes est qu'une erreur dans une des analyses peut entraîner une erreur dans l'autre analyse. Pour cette raison, nous avons préféré effectuer séparément les deux analyses puis comparer leurs résultats. De plus, les deux analyses étant relativement compliquées nous avons souhaité les développer séparément afin de pouvoir les pousser relativement à fond sans avoir à gérer des communications entre les deux systèmes. Un tel fonctionnement demande bien entendu la mise en place d'un principe de résolution des incohérences (celui-ci sera détaillé dans la section 11.2, page 196) et nécessite d'envisager la possibilité de retours arrière au niveau des analyses. Ce principe de fonctionnement est rendu possible par l'architecture globale que nous avons choisie et qui sera détaillée dans la section 2.2.

Dans le cas de la description d'itinéraires, il n'existe pas, à notre connaissance, de systèmes utilisant à la fois le texte et l'image. Cependant, comme l'a montré [Fraczak, 1996], une description d'itinéraire peut être non seulement textuelle mais aussi graphique et, dans ce cas, il est nécessaire d'effectuer non seulement l'interprétation du texte mais aussi celle du schéma. Au niveau de la description d'itinéraires, il existe cependant divers travaux qui ont montré l'importance d'avoir à la fois le texte et l'image. Ces travaux ne concernent pas l'interprétation de descriptions d'itinéraires mais la génération automatique de telles descriptions. Le projet VITRA GUIDE [Herzog *et al.*, 1993], développé en Allemagne, a pour but de générer automatiquement des descriptions d'itinéraires en utilisant non seulement des descriptions en langage naturel mais aussi des descriptions visuelles 2D ou 3D (visualisation de l'itinéraire sur une carte par exemple). Ce projet est issu du projet VITRA [21] dont le but est de permettre une interaction en langage naturel entre le système et l'utilisateur en ayant pour objectif principal de répondre aux questions de l'utilisateur concernant les rela-

21. VIsual TRAnslator.

tions spatiales et les trajectoires. Les concepteurs de VITRA se sont aperçus que, pour que les descriptions d'itinéraires soient mieux adaptées aux besoins des utilisateurs, il fallait non seulement donner une description en langage naturel, mais aussi une description visuelle de l'itinéraire. Tout en ayant un objectif différent du nôtre, l'équipe du projet VITRA en est arrivée aux mêmes conclusions que nous, à savoir que dans beaucoup de cas (dont la description d'itinéraires), les images sont aussi importantes que le texte.

De même, l'extraction d'information d'un document multimédia nécessite d'une part une reconnaissance des différentes parties qui le constituent (reconnaissances des parties graphiques et des parties textuelles avec reconnaissance des structures logiques et physiques du document), d'autre part une interprétation de ces différentes parties afin de déterminer les informations pertinentes du document. C'est dans cette dernière partie qu'un système d'interprétation de documents composites comme le nôtre, pourra être utilisé.

2.2 Architecture du projet global

2.2.1 Les principes

Le principe que nous avons mis en place pour effectuer l'analyse des constats amiables d'accidents automobiles consiste à découper, en trois parties bien distinctes (analyse du texte, analyse du schéma et enfin construction de la scène à partir des deux analyses), le travail d'interprétation du document. Ce découpage est basé d'une part sur la structure du document (document composé de deux parties : un texte et un schéma), d'autre part sur une conception particulière de la sémantique. En effet, nous considérons qu'un certain nombre d'informations sont apportées par le texte (ou le schéma) et que d'autres doivent être déduites des conditions d'énonciation et des connaissances sur le monde auquel fait référence le document.

Nous considérons, en effet, que l'énonciateur vise à "mettre en scène" des événements en en spécifiant à la fois le contenu et le point de vue à partir duquel la scène doit être observée et que le lecteur reconstruit cette scène et sa mise en perspective en utilisant le contenu du document comme autant d'instructions qui le guident dans cette opération de représentation. Cependant, le document "ne dit pas tout". En effet, tout ce qui est supposé connu par les deux interlocuteurs (but de l'énonciation, connaissances communes, etc.) ne fait pas partie du contenu du document. Par conséquent, l'ensemble des instructions fournies par le texte et le schéma est incomplet et doit donc être complété grâce à des informations externes au document. En effet, dans un texte, un mot comme *je* n'est pas ambigu et fait référence au locuteur, cependant, seules les conditions d'énonciation permettent de déterminer avec précision le référent de ce *je* (nous reviendrons en détail sur les aspects linguistiques de cette distinction section 6.1). De même, au niveau de l'analyse du schéma, celle-ci est susceptible de détecter un virage et de fournir une indication sur sa courbure mais elle ne peut pas déterminer la nature de ce virage (virage "serré" on non). Cette nature qui, lors de la construction de la scène, peut avoir son importance pour expliquer un véhicule déporté par exemple, ne peut être déterminée qu'à partir de connaissances encyclopédiques stipulant à partir de quel degré de courbure un virage peut être considéré comme serré.

C'est cette conception de la sémantique qui nous a amenés à séparer le travail d'interprétation du texte ainsi que celui du schéma en deux parties distinctes : d'une part une analyse de la partie considérée (soit le texte, soit le schéma), d'autre part une interprétation des informations extraites grâce à des connaissances sur le monde et à des connaissances sur la nature du document (conditions d'énonciation). Nous avons donc choisi de limiter la tâche de l'analyseur linguistique à l'extraction des contraintes imposées par le texte (entités mentionnées par le texte ainsi que leurs propriétés et les relations qu'elles entretiennent) et de laisser à un troisième sous-système (que nous avons appelé sous-système de construction de la scène) le soin d'effectuer l'interprétation du texte proprement dite en se basant sur ces contraintes et en utilisant des connaissances extra-linguistiques. Ces contraintes, qui définissent ce que nous avons appelé le niveau "sémantique linguistique", ne se réduisent pas à de l'information que devrait prendre en compte le processus de construction de la scène mais constituent un ensemble d'instructions auxquelles ce processus doit se conformer. Il y a donc finalement une interaction assez subtile entre le texte et les données extra-linguistiques : le texte apporte des règles de construction de la signification et ces règles s'appliquent, dans un cadre dépendant des conditions d'énonciation, à des entités et des événements intégrés dans une appréhension cohérente du monde dont on parle. De même, la tâche de l'analyseur de schémas est limitée à l'extraction des contraintes géométriques portant sur les divers éléments composant le schéma, ces contraintes, comme les contraintes générées par le texte, seront ensuite intégrées au niveau de la construction de la scène.

2.2.2 Architecture proposée

Nous obtenons alors un système où le travail d'interprétation du document est réparti entre 3 tâches bien distinctes : analyse géométrique du schéma, analyse linguistique du texte et construction de la scène, ces tâches étant réparties entre 3 sous-systèmes indépendants (cf. figure 2.1 - les flèches du schéma représentent le flot de données principal et non le contrôle, nous verrons section 6.2, page 91 et section 11.2, page 196 qu'il existe aussi des flots de données secondaires). Cette architecture nous permet non seulement de tenir compte des rôles des différentes données mises en jeu (informations linguistiques, informations géométriques et connaissances sur le domaine) mais aussi de respecter la structure même du document analysé (document composé de deux parties distinctes).

Le premier sous-système (*Analyse du schéma*) effectue une analyse géométrique du schéma en s'appuyant d'une part sur des règles de "construction" des routes basées sur des connaissances perceptives[22], d'autre part sur un système de catégorisation des véhicules basé sur les caractéristiques morphologiques des contours susceptibles de représenter un véhicule. Le résultat est un ensemble de *contraintes géométriques* (données sous forme d'équations) qui permettent de déterminer la situation lors de l'accident ainsi que la la disposition particulière des lieux (on verra, au chapitre 4 page 39, le format de ces contraintes).

L'analyse effectuée par le second sous-système (*Analyse Linguistique*) vise essentiellement à extraire des textes les faits "objectifs" (nous n'avons pas traité l'aspect argumentatif des textes). Elle a pour but de mettre en évidence les objets intervenant dans la scène, leurs relations spatiales, les différents événements cités ainsi que leur ordre de succession... Pour extraire ces faits, elle intègre une analyse syntaxique et un certain niveau d'analyse sémantique. L'analyse sémantique "linguistique" s'appuie, essentiellement, sur les mécanismes de la langue pour déterminer un certain nombre de valeurs (qui peuvent être temporelles, aspectuelles, actancielles, relatives à la détermination, etc.). Ce sont ces valeurs qui, jointes aux informations lexicales, permettent de construire un ensemble de relations que doivent nécessairement vérifier les entités et les événements auxquels le texte fait référence. Le résultat est donc un ensemble de *contraintes sémantiques* (chacun de ces ensembles est lié à un des *procès* du texte) qui déterminent en partie la construction de la signification (une description détaillée du

22. Par "connaissances perceptives", nous entendons connaissances sur la perception visuelle

format adopté pour ces contraintes est donné section 6.3, page 93).

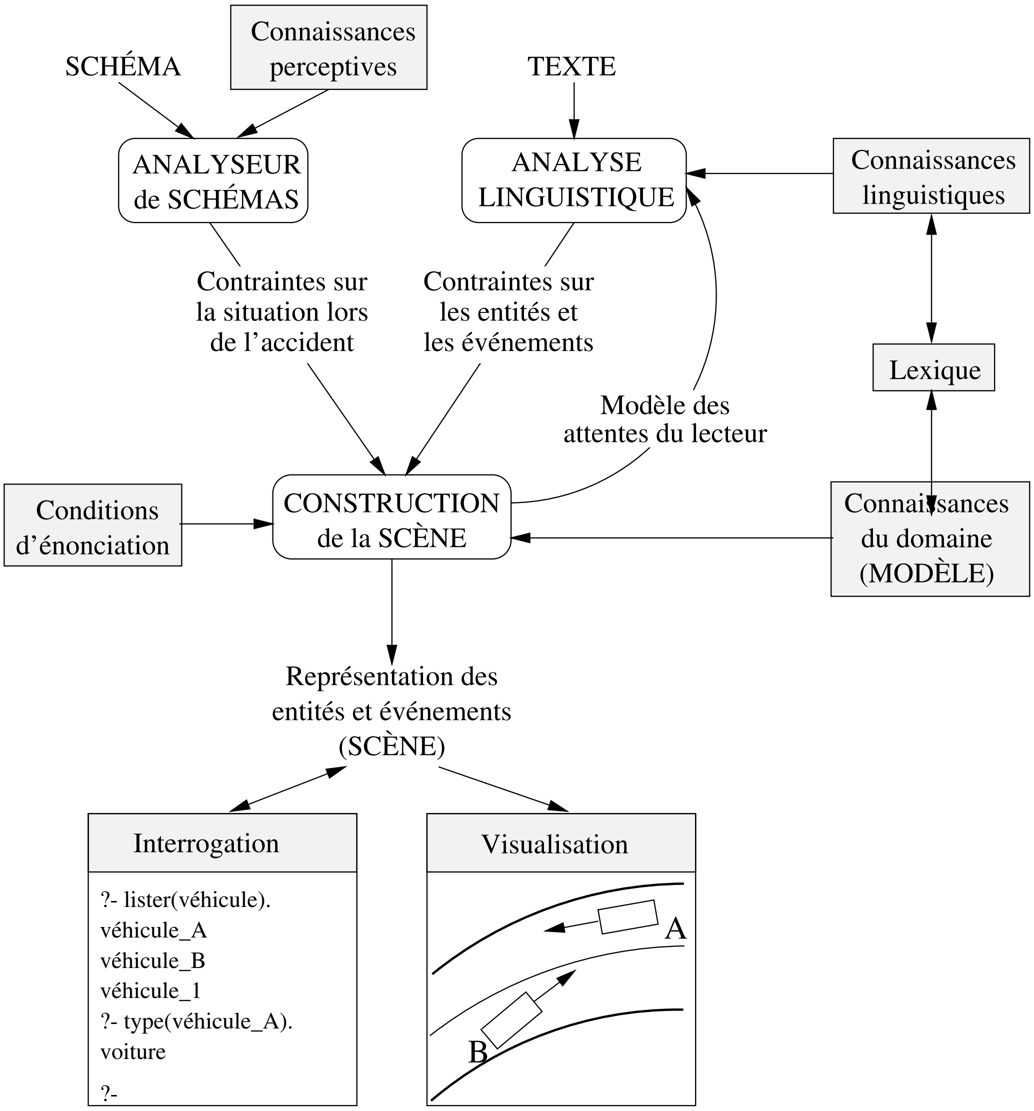


FIG. 2.1 - *Architecture du projet global (les flèches du schéma représentent le flot de données principal et non le contrôle)*

Le troisième sous-système (*Construction de la scène*) a pour but de construire une représentation de la scène décrite par le texte et le schéma en utilisant trois types de données : les connaissances du domaine, les conditions d'énonciation et de schématisation et les contraintes imposées par les deux sous-systèmes d'analyse.

Les connaissances du domaine que nous utilisons sont regroupées sous le terme de *modèle* (du monde de la route) et regroupent à la fois des représentations et des connaissances à proprement parler (cf. [Enjalbert & Victorri, 1994] et [Pied, 1994]). Les représentations utilisées sont essentiellement des entités géométriques (repères, rubans, courbes ...) permettant d'avoir un point de vue géométrique sur les lieux, des entités temporelles pour représenter les "instants" ou les plages d'instants (intervalle d'énonciation, intervalle lié au procès ...), des entités abstraites (les événements) pour représenter l'action décrite par le procès (un choc, un déplacement ...), des entités matérielles pour représenter les objets mobiles (les véhicules, les personnes ...) ainsi qu'un certain nombre d'entités permettant de représenter les "objets" pouvant intervenir dans une scène (routes, carrefours ...). Les connaissances utilisées regroupent des connaissances générales sur les entités géométriques, temporelles et matérielles ainsi que des connaissances plus spécifiques de situation de conduite automobile. Dans les documents que nous avons à interpréter, la référence aux objets (routes, véhicules ...) et phénomènes physiques (déplacements, chocs ...) étant particulièrement prégnante, nous avons donc choisi de modéliser ces objets et connaissances grâce à la géométrie analytique et à la physique classique (cinématique et dynamique) qui proposent un cadre de modélisation riche, bien maîtrisé et mathématisé. Ce modèle, qui ne peut, bien sûr, constituer qu'une représentation très réductrice des connaissances du domaine, définit les limites de compréhension du système. Ainsi, comme nous ne comptons pas, actuellement, modéliser les intentions du locuteur, rien de ce qui se rapporte à celles-ci ne sera pris en compte dans la construction de la scène, même si l'analyse linguistique parvient à les mettre en évidence.

Les conditions d'énonciation et de schématisation permettent de prendre en compte les informations sur la personne ayant rempli le constat (conducteur d'un des véhicules impliqué dans un accident, identifié comme le véhicule A ou B, etc.) ainsi que les conditions générales sur la scène à construire (se situant dans le passé par rapport au moment de l'énonciation, généralement sur une route, avec en général 2 véhicules, etc.). C'est aussi à ce niveau que nous pourrons spécifier les intentions probables du locuteur (se disculper vis-à-vis de sa compagnie d'assurance, etc.).

Les contraintes imposées par les deux autres sous-systèmes (analyseur linguistique et analyseur de schémas) sont d'une part les contraintes sémantiques imposées par l'analyse linguistique, d'autre part les contraintes géométriques imposées par l'analyse du schéma. Ces dernières portent essentiellement sur l'emplacement, au moment de l'accident, des différentes entités intervenant dans la scène.

Le rôle du sous-système de construction de la scène consiste alors, en quelque sorte, à compléter et à interpréter ces dernières informations à l'aide des autres sources de connaissances dont il dispose afin de construire une représentation cohérente et intelligible de la scène. Une fois la scène construite, le système peut exécuter les tâches demandées en sortie : répondre à des questions sur les véhicules (position, vitesse, direction, etc.) ou effectuer une visualisation schématique voir dynamique de la scène. Ces différentes tâches ont pour but de nous permettre une vérification de la cohérence de la scène construite et donc une vérification de l'interprétation effectuée. Ce sous-système ne faisant pas l'objet de cette thèse il ne sera pas plus amplement détaillé dans ce mémoire. Une description plus fine se trouve cependant dans [Enjalbert & Victorri, 1994] et [Pied, 1994].

La distribution des tâches entre les trois sous-systèmes que nous avons adoptée a une part d'arbitraire mais il est toujours possible d'effectuer un "retour arrière" : le sous-système de construction de la scène peut demander à l'un ou l'autre des sous-systèmes d'analyse des renseignements complémentaires. En effet, comme nous l'avons vu précédemment, les liens entre connaissances linguistiques et connaissances extra-linguistiques d'une part, entre connaissances perceptives et connaissances du domaine d'autre part, sont nombreux et devront donner lieu à de nombreuses interactions entre les différents sous-systèmes de notre architecture. La coopération entre analyseur linguistique et sous-système de construction de la scène est effectuée d'une part par l'intermédiaire du lexique, d'autre part par l'intermédiaire de ce que nous avons appelé le "modèle des attentes du lecteur" (les principes de cette coopération seront plus amplement détaillés en section 6.2, page 91). Les principes mis en place pour permettre ces interactions lors de l'implémentation des différents sous-systèmes seront détaillés en partie IV.

2.3 Quelques travaux similaires

La conception de la sémantique pour laquelle nous avons opté se retrouve dans un certain nombre de théories psychologiques portant sur la compréhension du discours comme notamment celles de [Van Dijk & Kintsch, 1983] et de [Johnson-Laird, 1983] qui considèrent que le traitement des messages langagiers consiste en la création de deux principaux types de représentation mentale : "le modèle de texte" qui représente le contenu linguistique du discours et le "modèle de la situation" qui est élaboré à partir du modèle du texte en utilisant d'autres types de connaissances. Un tel découpage est aussi utilisé en modélisation de la description d'itinéraires [Fraczak, 1996]. Cette conception de la sémantique est aussi à la base de l'architecture proposée par [Crouch & Pulman, 1993] pour une application très différente (interface avec un système de planification).

Lidia Fraczak s'est attachée à la modélisation des connaissances nécessaires pour effectuer l'interprétation de textes décrivant des itinéraires dans le but de développer des systèmes informatiques et notamment des systèmes d'aide à la navigation. Son étude, basée sur trois corpus de descriptions d'itinéraires en langage naturel, l'a amenée à considérer deux niveaux distincts dans l'analyse et la modélisation : le niveau linguistique et le niveau conceptuel. Le premier niveau décrit l'organisation discursive de la description d'itinéraire. Le second niveau concerne la structure conceptuelle sous-jacente (indépendante de la langue) et est basé sur un modèle des connaissances sur la structure de l'itinéraire prototypique et sur ses composantes. Ce second niveau peut servir de support à une "représentation référentielle" de l'itinéraire créée pendant le traitement automatique et qui peut être donnée sous la forme de descriptions langagières ou graphiques.

Le système de [Crouch & Pulman, 1993] est composé de trois modules (une interface en langage naturel, un évaluateur de langage d'interrogation de plan et un planificateur). L'interface en langage naturel a pour but de produire, à partir du texte en anglais, des formes logiques qui seront ensuite traduites dans le langage d'interrogation de plan utilisé par l'évaluateur et fournies à ce dernier. Ce second module a pour but de répondre à un certain nombre de questions en demandant au planificateur de construire des plans. L'architecture de ce système est donc relativement proche de celle que nous avons mise en place mais la distinction que nous effectuons entre éléments de sens provenant du texte et intégration dans le modèle du monde n'est pas aussi nette : des connaissances extra-linguistiques sont utilisées par le premier module (interface en langage naturel).

Une des théories linguistique de représentation du discours (la Théorie de la Représentation du Discours proposée par Hans Kamp [Kamp, 1981] et [Kamp & Reyle, 199] pour résoudre entre autres les problèmes d'anaphore et notée DRT [23]) présente aussi un tel découpage. En effet, si la plupart des théories sémantiques passent directement du niveau syntaxique d'une phrase à son interprétation dans un modèle, la DRT utilise

23. Discourse Representation Theory.

un niveau intermédiaire dont les structures de représentation sont appelées les DRS[24]. L'interprétation d'un discours dans le cadre de la DRT se fait en deux étapes : d'abord construction de la DRS associée au texte, puis interprétation de cette DRS dans un modèle. Le plus souvent, la DRS associée au texte (appelée souvent "DRS englobante") inclut, au niveau de ses conditions, d'autres DRS reliées par une condition particulière qui peut être une implication, une disjonction, une négation, etc.. Ces conditions particulières correspondent à différent aspects de la langue particulièrement étudiés dans le cadre de la DRT. Les DRS représentent le niveau sémantique du texte et sont composées d'une part des référents de discours (les entités apparaissant dans le texte), d'autre part des propriétés (ou conditions) applicables entre les référents de discours de la DRS (les propriétés des entités et les relations qu'elles entretiennent). Ces structures peuvent être considérées comme étant l'équivalent de nos contraintes sémantiques.

Cette théorie a donné lieu à un certain nombre d'extensions comme par exemple, la SDRT[25] de Nicholas Asher [Asher, 1993] (extension tenant compte de la structure du discours en intégrant un ensemble de relations de discours). Dans cette théorie, la structure de base est la SDRS[26] qui est une structure composée d'un ensemble de DRS ainsi que d'un ensemble de relations entre ces DRS. Ces DRS représentent en fait des segments du discours et les relations entre les DRS reflètent les relations de discours entre les segments correspondants. Le formalisme de la SDRT a été utilisé, entre autres, par Pierre Sablayrolles [Sablayrolles, 1995] pour représenter la sémantique spatio-temporelle des verbes de mouvement.

La DRT est aussi à la base de l'architecture mise en place dans le cadre du projet allemand de compréhension de textes LILOG[27] [Herzog & Rollinger, 1991]. Dans ce système, l'analyse linguistique est réalisée à l'aide de grammaires d'unification grâce au système STUF[28] (STUF est un atelier qui permet d'implémenter aussi bien des grammaires catégorielles que des grammaires HPSG) et les résultats de cette dernière sont donnés dans un langage logique intermédiaire, les EDRS[29]. Celles-ci sont alors traduites dans le langage de représentation des connaissances L_{lilog} utilisé par le module logique. Celui-ci a deux tâches principales. D'une part, étendre la base de connaissances à partir des informations extraites du texte et des connaissances dont il dispose déjà,

24. Discourse Representation Structures.
25. Segmented Discourse Representation Theory.
26. Segmented Discourse Representation Structure.
27. LInguistic and LOGic methods and tools.
28. STuttgart Unification Formalism.
29. Extended Discourse Representation Structure.

d'autre part, répondre aux questions posées par l'utilisateur. Comme nous pouvons le voir, les fonctionnalités du système mis en place dans le cadre de ce projet sont assez proches de nos préoccupations.

Le projet allemand VERBMOBIL (projet de traduction automatique allemand/anglais et japonais/anglais qui regroupe une centaine de chercheurs issus de 29 institutions différentes - [Dorna & Emele, 1996]) utilise lui aussi une représentation sémantique intermédiaire. Cette représentation, basée sur les DRS, est nommé LUD[30][Bos *et al.*, 1996]. Ce formalisme a été choisi car il est plus adapté que les DRS pour représenter la sous-spécification de certains phénomènes de la langue. L'idée de base qui les a amenés à mettre en place ce langage est que beaucoup d'expressions du langage ne sont pas directement représentables par des DRS mais sont représentables par des représentations décrivant plusieurs DRS. Les représentations LUD correspondent donc à l'équivalent de plusieurs DRS et sont utilisées pour construire les DRS correspondant au texte. Les représentations LUD sont relativement proches des UDRS[31] de [Reyle, 1993] (les UDRS sont composées d'un ensemble de DRS reliées par un ordre partiel) excepté qu'elles sont, en principe, indépendantes du langage objet (dans le cas présent, la DRT) contrairement aux UDRS.

Nous verrons ultérieurement (section 6.3, page 97), après avoir donné plus de précisions sur les analyseurs et plus spécialement sur l'analyseur linguistique, les différences qui peuvent exister entre le principe de découpage appliqué par la DRT et le nôtre.

30. description Language for Underspecification Discourse representation.
31. Underspecified Discourse Representation Structures.

Deuxième partie

L'ANALYSEUR DE SCHÉMAS

L'analyseur de schémas que nous avons mis en place a pour but de déterminer les contraintes géométriques imposées par le schéma figurant sur le constat.

Nous nous sommes attachés, dans cette partie, à présenter un certain nombre de travaux portant sur l'analyse de schémas et à préciser les particularités de nos schémas par rapport à ceux couramment étudiés. C'est à partir de ces caractéristiques que nous avons spécifié l'architecture de cet analyseur.

Après avoir précisé les caractéristiques de l'architecture que nous avons choisie, nous détaillerons les différentes parties qui composent cet analyseur et nous donnerons des précisions sur l'implémentation du système. Nous conclurons cette partie par une présentation critique des résultats obtenus par la maquette d'analyseur réalisée et des améliorations souhaitables pour pouvoir effectuer une meilleure analyse des schémas de constats d'accidents.

Chapitre 3

Les schémas de constats d'accidents

3.1 L'interprétation de schémas

Un certain nombre de projets concernant l'interprétation de schémas ont déjà été mis en place. Ils concernent divers domaines comme l'interprétation de schémas électriques [Bley, 1984], l'interprétation de plans cadastraux ([Boatto *et al.*, 1992] et [Antoine, 1991]), l'interprétation de dessins techniques ([Nagasamy & Langrana, 1990], [Haralick & Queeney, 1982], [Fahns *et al.*, 1988], [Collin, 1992], [Vaxivière, 1995]) etc.

Ces différents types de schémas sont à la fois proches et éloignés des schémas que nous avons à étudier. Ils sont proches par le fait que les uns comme les autres sont des simplifications de la réalité. Ils sont éloignés parce que, contrairement aux autres types de schémas mentionnés, les schémas de constats d'assurance ne sont pas normalisés. En effet, que ce soit un plan cadastral, un schéma électrique ou un dessin technique, les schémas sont effectués en fonction d'une norme (nationale ou internationale), par contre pour les schémas des constats d'assurance il n'existe aucune norme pour représenter les objets qu'ils contiennent (véhicules, ronds-points, routes, panneaux ...) et de plus, l'ensemble des objets qu'il est possible de trouver dans un schéma n'est qu'imparfaitement délimité. Les seuls renseignements dont nous disposons sont ceux inscrits sur le constat amiable à savoir que l'utilisateur doit indiquer, normalement, le tracé des routes, la direction des véhicules, leur position au moment du choc, les signaux routiers ainsi que le nom des rues ou routes dessinées (instructions données aux auteurs au niveau du croquis, cf. figure 1.1, mais qui, le plus souvent, ne sont pas suivies).

L'interprétation de schémas que nous effectuons peut être considérée comme relativement proche de la saisie automatique de documents : au lieu de transformer l'image numérisée en un document électronique conforme à une certaine norme, nous la transformons en des primitives compréhensibles par le sous-système chargé de la construction de la scène. Celui-ci travaillant avec des primitives complexes, nous devons par conséquent aller plus loin dans l'interprétation que le projet développé dans [Tombre, 1987] afin d'obtenir un résultat approprié à notre problème. En effet, dans ce projet, K. Tombre effectue uniquement une extraction des différentes composantes connexes de l'image sans en effectuer une reconnaissance complète : il ne cherche par à déterminer la nature (carré, cercle, élément complexe composé de plusieurs composantes connexes élémentaires, ...) des composantes extraites. La seule chose qui est effectuée au niveau des composantes complexes consiste à envisager une méthode de reconnaissance des polygones particuliers (les rectangles, les triangles, ...) basée sur le nombre de segments et les relations entre segments. Cependant, cette méthode n'a été qu'envisagée et non implémentée et testée.

3.2 Les caractéristiques de notre analyseur

L'objectif premier de ce travail est de déterminer une architecture pour un analyseur de schémas de constats d'accidents et de valider cette dernière par l'intermédiaire d'une première réalisation. Cette réalisation n'a pas comme but de traiter l'ensemble des problèmes liés aux schémas de constats mais de vérifier si une telle application est réalisable et s'il est possible d'extraire des schémas les éléments principaux (c'est-à-dire les routes, les véhicules, etc.) et ce, en n'utilisant que des connaissances perceptives.

Pour mettre au point cette architecture, et comme nous ne disposions que de très peu de constats réels (7 dont deux correspondant à la transcription, par deux scripteurs différents, du même accident), nous avons tout d'abord réalisé 6 schémas représentant différentes situations d'accident à partir de textes du corpus linguistique : un accident entre deux véhicules sur une route ayant plusieurs files (schéma correspondant au texte A7 du corpus, annexe B.1), un accident dans un virage (texte A8), un accident dans un carrefour (texte A4), un accident entre trois véhicules (texte A3 où nous avons remplacé la moto par une voiture), un accident au bord d'un carrefour (texte A1) et enfin un accident dans un rond-point (texte B18). Afin d'élargir le corpus et de valider l'architecture sur un plus grand nombre de schémas, nous avons demandé à une quinzaine de personnes (quelques chercheurs de notre laboratoire) d'effectuer les schémas correspondant à quatre textes que nous leur avons fournis. Cette expérience nous a permis d'enrichir le corpus de 64 nouveaux schémas.

Les objets contenus dans les schémas à étudier et que nous souhaitons extraire sont essentiellement les routes, les véhicules, les carrefours et la signalisation (qui peut être de diverses formes mais essentiellement sous forme de panneau ou de type de ligne - continue, discontinue). Les schémas qui nous intéressent sont composés de deux types d'éléments : d'une part, les éléments fermés (représentés par des contours connexes fermés) et d'autre part, les éléments ouverts (les segments non intégrés dans un contour connexe fermé). Après observation des schémas dont nous disposions, nous nous sommes aperçus que les routes étaient en général représentées à l'aide d'éléments ouverts (ce sont des ensembles de segments organisés suivant certaines règles) tandis que les véhicules étaient essentiellement représentés par des éléments fermés. Les panneaux, quant à eux, peuvent être composés d'éléments des deux types.

L'architecture que nous avons mise en place pour cet analyseur de schémas est basée sur ces observations et donc sur un découpage de l'image en éléments fermés et éléments ouverts.

Chapitre 4

La réalisation

L'architecture utilisée pour cet analyseur de schémas est une architecture modulaire composée de 4 modules (cf. figure 4.1) qui, pour certains, sont divisés en sous-modules.

Les quatre modules principaux sont les suivants :

1. *Traitements bas niveau classiques et tri des éléments suivant leur catégorie* : il effectue les traitements de bas niveau couramment utilisés pour le traitement d'images, à savoir la squelettisation, la détection de contours et le lissage. Ces traitements ne font aucun travail d'interprétation d'image mais permettent juste d'effectuer une mise en forme de l'image pour l'exécution des traitements suivants. Ce module effectue aussi la partition des éléments en deux groupes (éléments fermés et éléments ouverts).

2. *Extraction des routes à partir des éléments ouverts* : ce module travaille uniquement sur les éléments ouverts de l'image à partir desquels il doit déterminer les routes existantes, leur position dans l'image ainsi que les relations existant entre les différentes routes (déterminer, par exemple, si deux routes se croisent). Extraire les routes existantes consiste non seulement à indiquer les routes présentes dans l'image mais aussi à donner les différentes parties composant ces routes (parties courbes et parties droites) dans le but de signaler d'éventuels virages.

3. *Catégorisation des éléments fermés* : ce module travaille uniquement sur les éléments fermés de l'image et permet de les catégoriser, c'est-à-dire de préciser pour chaque élément son type (voiture, camion, centre de rond-point, haut de panneau, etc.).

Les modules 2 et 3 travaillent donc sur les résultats fournis par le module 1 et peuvent être amenés à communiquer entre eux afin de pouvoir traiter le cas d'objets composés d'éléments de plusieurs types et de lever certaines ambiguïtés sur le type des objets.

4. *Mise en forme des résultats* : il formalise les résultats des modules 2 et 3 de façon à les rendre compréhensibles par le sous-système chargé d'effectuer la construction de la scène.

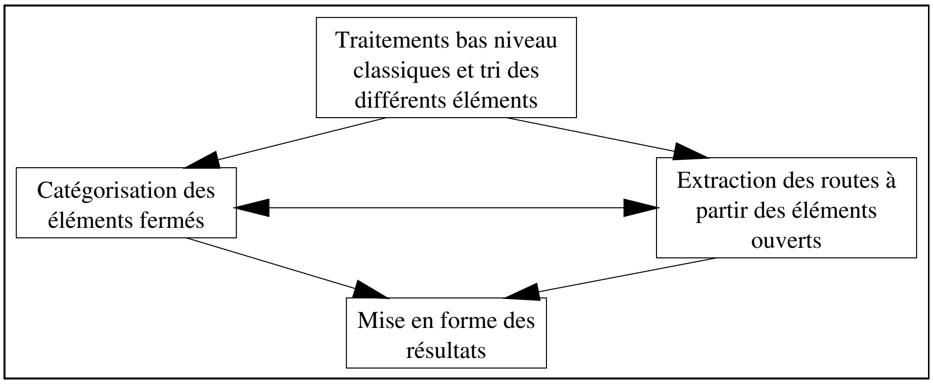


FIG. 4.1 - *Architecture de l'analyseur de schémas*

Nous allons maintenant, avant de préciser l'implémentation effectuée, détailler, module par module, les principes mis en place.

4.1 Traitements bas niveau

Ce module utilise un certain nombre d'opérateurs de traitement bas niveau couramment utilisés en imagerie et a pour but essentiel de partitionner l'image en deux groupes d'éléments : les éléments ouverts et les éléments fermés.

Les schémas n'ayant que deux couleurs bien distinctes (une pour le fond et une pour le dessin en lui-même), les images utilisées sont donc des images binaires et il n'est pas nécessaire de les binariser pour pouvoir effectuer les traitements suivants.

Les trois premières opérations effectuées sont : la squelettisation, la détection de contours et le lissage de contours. Ces trois opérations nous permettent de préparer l'image pour les traitements ultérieurs.

La squelettisation est un procédé classique en traitement d'images qui consiste à remplacer une région par son "squelette" (ensemble des points intérieurs à la région et équidistants de deux de ses bords). Elle nous sert ici à éliminer les surépaisseurs au niveau du tracé et donc à transformer tout trait d'une épaisseur supérieure à 1 en un trait d'épaisseur égale à 1 (cf. figure 4.2).

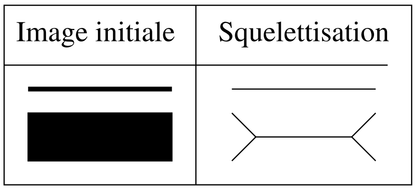


FIG. 4.2 - *Squelettisation*

La détection de contours (ou suivi de contours) consiste à construire, à partir d'une image bitmap, une image vectorielle en regroupant les segments connexes. Cette opération permet d'obtenir une description plus simplifiée et plus synthétique des tracés présents dans l'image. Pour effectuer cette opération, nous avons utilisé un algorithme en deux passes : la première identifie les nœuds, c'est-à-dire les points qui se trouvent à la jonction de plusieurs traits ainsi que ceux qui se trouvent à une extrémité (cf. figure 4.3); la seconde étape est un suivi de contours "à main droite" classique qui consiste à parcourir l'image en laissant le tracé à main droite et à créer les segments correspondants, ceux-ci sont regroupés en ensembles (que nous appelons des polysegments) en fonction des nœuds (1 polysegment regroupe l'ensemble des segments se trouvant entre deux nœuds - cf. figure 4.3).

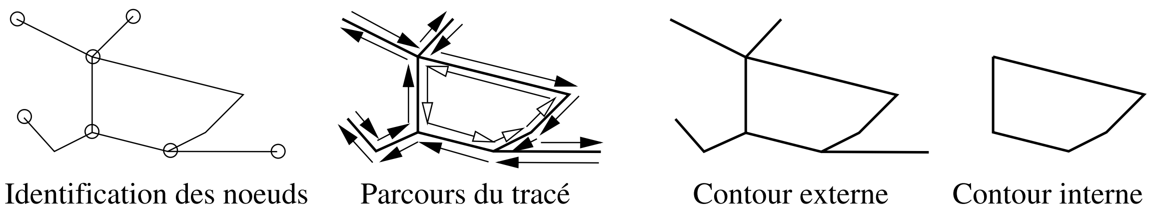


FIG. 4.3 - *Détection des contours*

L'inconvénient d'un tel suivi de contours est qu'il donne beaucoup de segments (les diagonales sont souvent représentées par un ensemble de segments formant un escalier). C'est pour remédier à cet inconvénient que nous avons effectué ensuite un lissage.

Pour effectuer ce lissage (ou approximation polygonale), nous utilisons une variante de la méthode proposée par P. Garnesson [Garnesson, 1991]. Cette méthode est basée sur un principe de minimisation d'une fonction d'énergie le long des polysegments (ensemble regroupant les segments se trouvant entre deux nœuds correspondant soit à une extrémité, soit à une jonction). Le principe est simple : il considère que chacun des sommets du polysegment est doté d'une énergie qui traduit la résistance de ce point à être éliminé. Le point le moins résistant est alors éliminé et le processus se répète tant qu'il existe des points dont l'énergie est en dessous d'un seul de résistance donné. La fonction d'énergie utilisée par P. Garnesson est la distance du point au segment reliant ses deux points voisins. Nous avons utilisé une variant de cette fonction d'énergie qui prend en compte non seulement la distance au segment reliant les deux points voisins mais aussi l'angle non orienté au point considéré. Cette fonction (comme le montre la figure 4.4) permet de mieux conserver les angles.

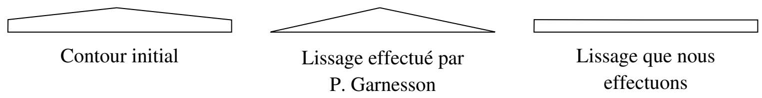


FIG. 4.4 - *Différence entre le lissage de [Garnesson, 1991] et le nôtre*

Nous présentons brièvement ci-dessous l'algorithme utilisé (algorithme dont nous pouvons voir une étape sur la figure 4.5) :

```
- calculer les d_i : distances (Point_i, [Point_{i-1}, Point_{i+1}])
- calculer les a_i : angles (Point_{i-1}, Point_i, Point_{i+1})
- Tant qu' il existe un d_i inférieur au seuil s Faire
  | - supprimer le point Point_j correspondant au plus petit angle et
  |   dont le d_j est inférieur à s
  | - recalculer les distances d_{j-1} et d_{j+1}
  | - recalculer les angles a_{j-1} et a_{j+1}
```

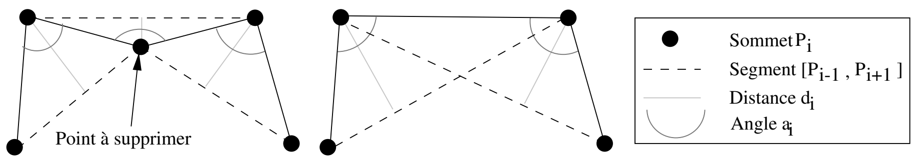


FIG. 4.5 - *Une étape du lissage*

Après ces différentes étapes, nous pouvons commencer les traitements spécifiques à notre application.

Le seul traitement spécifique que nous considérons comme étant de bas niveau est le découpage de l'image en deux groupes d'éléments : les éléments fermés et les éléments ouverts. Ce traitement se divise en deux parties : extraction des éléments fermés puis soustraction de ces éléments à l'image initiale lissée afin d'obtenir les éléments ouverts.

Les schémas étant dessinés à la main, il arrive qu'il y ait des erreurs de tracé. La plus gênante de ces erreurs étant le fait que deux traits qui devraient se toucher sont malheureusement séparés par un léger espace (sans doute dû à un lever de crayon un peu rapide - cf. figure 4.6). De plus, ces erreurs, loin d'être supprimées par la numérisation, sont au contraire amplifiées.

Une conséquence directe de ce genre d'erreur est le fait que des éléments qui, au départ étaient pratiquement fermés, se trouvent avec un espace qui fait qu'ils ne sont plus considérés comme tels. Par conséquent, avant d'extraire les éléments fermés, nous sommes obligés d'effectuer, de manière automatique, une "légère" fermeture de contours (cf. figure 4.6). Cette fermeture consiste à agrandir chaque segment de l'image lissée suivant sa direction principale jusqu'à "toucher" un autre segment, et ce, uniquement de quelques pixels (d'où le terme de "légère" fermeture) de façon à ne pas fermer des éléments qui n'ont pas à l'être. Le nombre de pixels utilisés pour le prolongement n'est pas choisi au hasard mais est fonction de la numérisation effectuée précédemment.

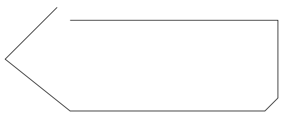
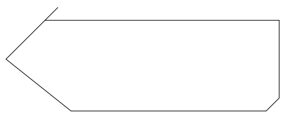
Élément incomplètement fermé	Élément après fermeture
	

FIG. 4.6 - *Exemple de fermeture*

L'extraction des éléments fermés se base sur le fait qu'un élément fermé a deux contours : un contour interne et un contour externe contrairement aux éléments ouverts qui n'ont qu'un seul contour. De plus, dans le cas des éléments fermés, seul les contours internes nous intéressent. En effet, comme le montre la figure 4.7, dans le cas où deux éléments se touchent, pour avoir les deux éléments il faut prendre les contours internes et non le contour externe.

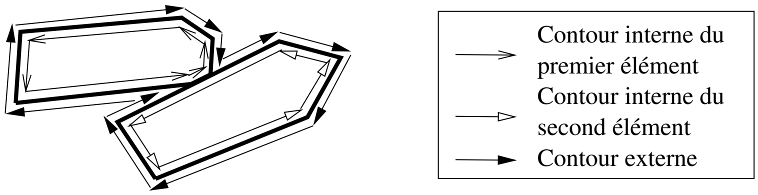


FIG. 4.7 - *Contours internes et externe quand deux éléments se touchent*

Pour obtenir les éléments fermés il faut donc déterminer, parmi l'ensemble des contours, ceux qui sont des contours internes. Pour cela, nous nous basons sur le fait qu'un contour interne "tourne" dans le sens trigonométrique et que tout point du contour "voit" l'ensemble du contour sous un angle de 2π contrairement à un contour externe pour lequel cet angle est de -2π ou au contour d'un élément ouvert pour lequel cet angle est de 0. L'angle orienté sous lequel un point P voit un contour est égal à la somme des angles orientés sous lesquels ce point voit chacun des segments du contour. Cependant, si on prend comme point P un des sommets du contour, celui-ci n'a aucune "vue" sur les segments auxquels il appartient (cf. angle α du premier dessin de la figure 4.8). C'est pour cette raison que nous prenons, pour détecter les contours internes, non seulement les sommets du contour mais aussi les points qui sont situés sur les segments (cf. second dessin de la figure 4.8 : l'angle sous lequel le point P voit le contour est égal à la somme $\alpha_1 + \alpha_2 + \alpha_3$ soit 2π).

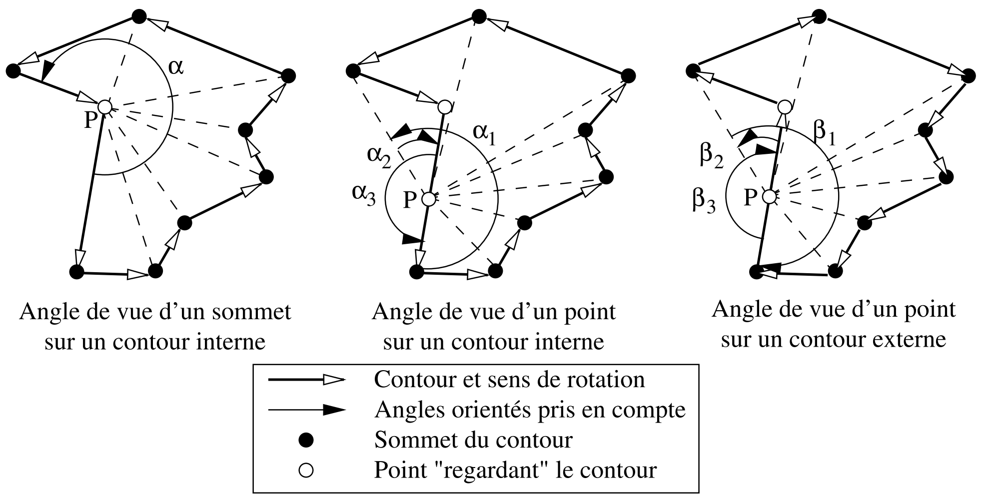


FIG. 4.8 - *Angles de vue d'un contour*

Pour effectuer la recherche des éléments fermés nous utilisons donc les algorithmes suivants :

Construction de la liste des contours internes

- **Pour chacun** des contours **Faire**
  - **Si** le contour est interne **Faire**
    - ajouter le contour à la liste des contours internes

Déterminer si un contour est interne

- **Pour chacun** des points P du contour (même ceux qui ne sont pas des extrémités de segments) **Faire**
  - calculer l'angle orienté sous lequel le point P voit le contour
  - **Suivant** la valeur de l'angle :
    - $2 * \pi$: le contour est interne
    - $-2 * \pi$: le contour est externe
    - 0 : le contour est ouvert

Le problème de cet algorithme de recherche des éléments fermés est qu'il est entièrement lié à l'algorithme de détection de contours. En effet, si on change cet algorithme en un suivi de contours "à main gauche" par exemple, un contour interne sera caractérisé par un angle de vue de -2π et un contour externe par un angle de vue de 2π; tout changement au niveau de l'algorithme de détection de contours risque d'entraîner une modification de l'algorithme de recherche des éléments fermés.

Pour obtenir les éléments ouverts de l'image, nous considérons que tout segment appartenant à l'image de départ lissée mais pas à l'image formée par les éléments fermés est un élément ouvert.

A l'issue des traitements de bas niveau nous obtenons donc deux images vectorielles qui à elles deux donnent l'image lissée (cf. figure 4.9). On peut alors aborder les traitements de haut niveau, c'est-à-dire l'analyse de l'image proprement dite.

4.2 Extraction des routes

Ce module a pour but d'extraire, de l'image composée des éléments ouverts, les routes existantes. Les schémas sur lesquels nous travaillons étant une représentation

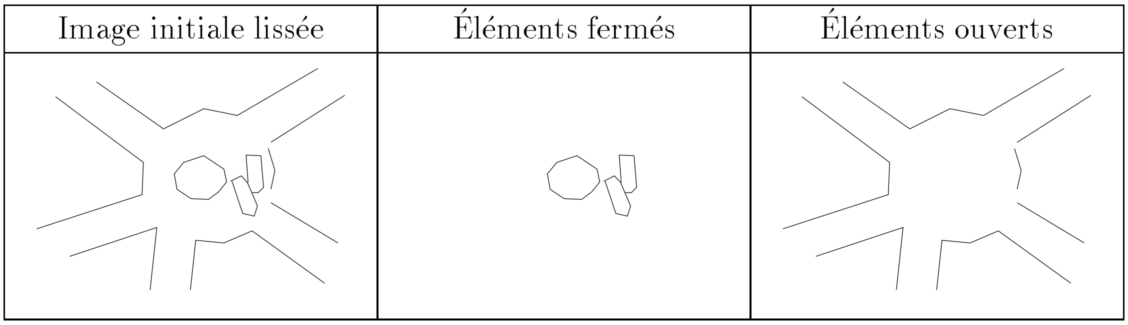


FIG. 4.9 - *Résultat des traitements bas niveau*

du réel effectuée à la main, il nous était impossible d'effectuer l'extraction et la reconnaissance des routes à partir de modèles prédéfinis comme cela est réalisé dans différents systèmes comme celui mis en place dans le cadre du projet PROMETHEUS[32] [Broggi & Bertè, 1995] où la détection des routes est effectuée à l'aide de la superposition à l'image d'un modèle de route. Nous avons donc choisi de nous baser sur un ensemble de règles générales qui permettent de déterminer si un ensemble de données de base (primitives élémentaires) forme un objet tout en gardant la possibilité de pouvoir revoir notre jugement en cas d'incohérence.

Pour pouvoir déterminer les routes existantes, il nous fallait tout d'abord donner des règles nous permettant de caractériser une route, c'est-à-dire une définition de la route. Ce module travaillant sur une image vectorielle, la définition que nous avons utilisée est basée sur la notion de segment.

Nous allons donc, dans un premier temps, donner la définition de la route que nous avons utilisée puis nous préciserons le fonctionnement de ce module en se basant sur cette définition.

4.2.1 Qu'est-ce qu'une route?

A partir des observations que nous avons effectuées sur les schémas au niveau des routes, nous avons extrait un ensemble de définitions que nous allons donner maintenant. Afin de les clarifier, nous nous appuierons sur la figure 4.10.

32. Program for a European Traffic with Highest Efficiency and Unprecedented Safety - projet ayant pour but de mettre en place un système de reconnaissance des routes dans un cadre d'assistance à la conduite automobile: les images traitées sont donc des images réelles correspondant à ce que voit le conducteur.

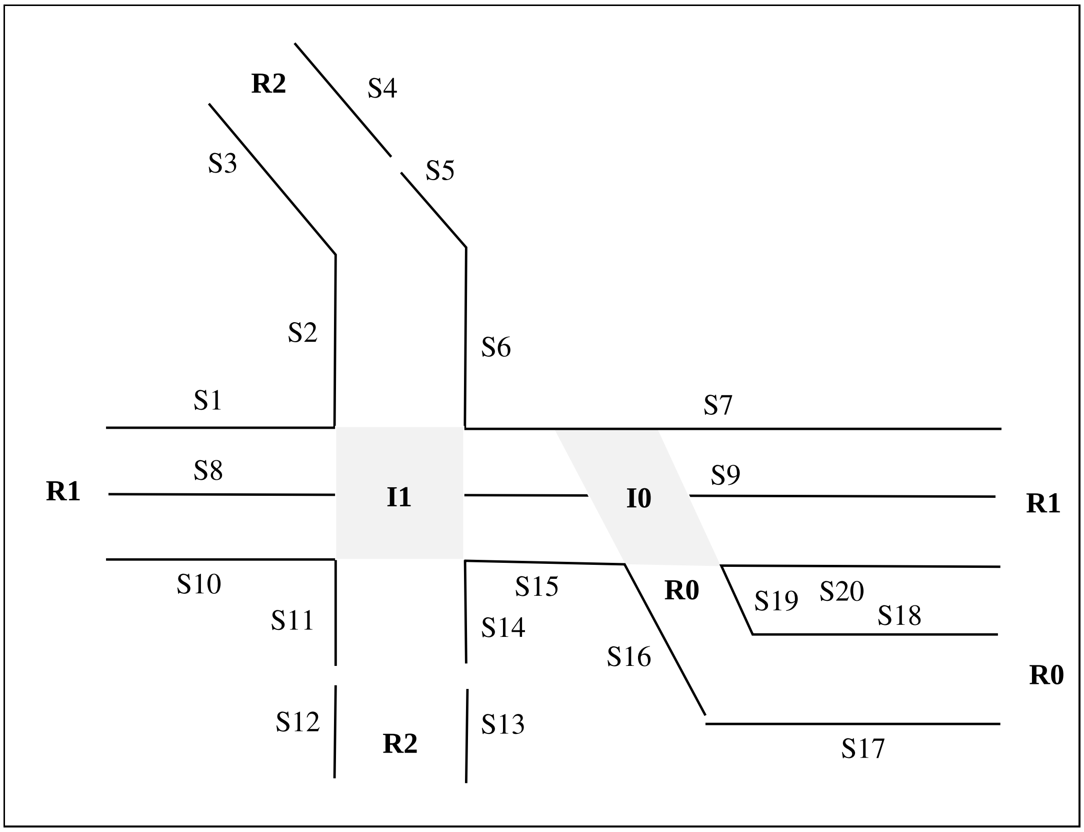


Fig. 4.10 - *Différentes configurations de route*

1) une **ligne** est composée d'un segment ou d'un ensemble de segments qui se prolongent (sur la figure 4.10, les segments S_4 et S_5 forment une ligne).

2) une **bande** est composée de deux lignes qui sont parallèles et face à face (une même ligne peut donc appartenir à plusieurs bandes - sur la figure 4.10, S_1 forme une bande avec S_8 et une autre avec S_{10}).

3) On appelle **bande minimale** une bande telle que l'ensemble des segments formant la bande soit minimal pour obtenir une bande ayant les deux bords approximativement de même longueur. Sur la figure 4.10, la bande B_1 composée de S_{11} et S_{14} est une bande minimale mais pas la bande B_2 composée de S_7 et S_{15} car ses deux bords sont de longueurs différentes et car S_{15} peut être complété avec S_{20} pour donner une bande minimale. De même, la bande B_3, composée pour un de ses bords de (S_{11}, S_{12}) et pour son autre bord de (S_{13}, S_{14}), n'est pas minimale car il existe un sous-ensemble de B_3 (B_1) dont les deux bords sont approximativement de même longueur et qui forme donc une bande minimale.

4) un **morceau de route** est composé de bandes minimales ayant un bord en commun et dont les bords ne sont pas reliés par un ensemble de segments. Sur la figure 4.10, les segments S_1, S_8 et S_{10} appartiennent à un même morceau. Ce morceau a comme bords S_1 et S_{10}. Par contre, S_{18} et S_{20} n'appartiennent pas à un même morceau parce qu'ils sont reliés par S_{19} (cela nous évite de considérer un terre-plein comme étant un morceau de route).

5) un **ensemble de morceaux de route** est composé soit d'un seul élément, soit de plusieurs éléments qui se touchent par leurs deux bords. Sur la figure 4.10, les segments S_3, S_4 et S_5 forment un morceau de route, les segments S_2 et S_6 en forment un autre. Ces deux morceaux appartiennent au même ensemble car leurs bords se touchent.

6) une **route** est une liste d'ensembles de morceaux de route qui sont deux à deux alignés mais pas forcément contigus. Sur la figure 4.10, la route R_2 est formée de 2 ensembles de morceaux de route : d'une part celui formé des morceaux ((S_4, S_5), (S_3)) et ((S_2), (S_6)) et celui formé des morceaux ((S_{11}, S_{14})) et ((S_{12}), (S_{13})).

À ces définitions nous avons ajouté la définition suivante (qui découle des précédentes) :

7) il existe une **intersection** entre deux routes R_i et R_j quand un des ensembles de morceaux de R_i touche un des ensembles de morceaux de R_j par un seul de ses bords.

Cet ensemble de définition nous permet d'obtenir sur la figure 4.10, les routes R_0, R_1 et R_2 ainsi que les intersections I_0 et I_1 (zones grisées).

Pour mettre en place ces différentes définitions, nous avons utilisé des relations basées sur des contraintes perceptives. Ces relations sont les suivantes : *connexité*, *parallélisme* et *prolongement*. Elles sont définies pour chacune des catégories d'objets définissant une route et sont toutes basées sur les relations entre segments définies ci-dessous. La relation de *connexité* correspond au fait que deux segments sont connexes (c'est-à-dire proches l'un de l'autre), la relation de *parallélisme* correspond au fait que deux segments sont parallèles et face à face et la relation de *prolongement* correspond au fait que deux segments sont dans le prolongement l'un de l'autre.

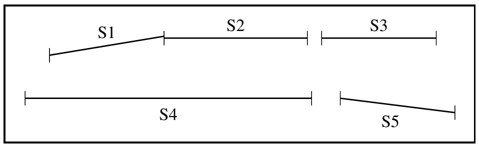


FIG. 4.11 - *Relations entre segments*

Un segment S_i a une relation de *connexité* avec un segment S_j si la plus petite des distances entre les extrémités des deux segments est inférieure à un certain seuil. Celui-ci doit permettre de tenir compte des erreurs de tracé mais sans prendre en compte des segments trop éloignés. Donc, comme dans le cas de la fermeture de contours, le seuil dépend de la numérisation. Cette relation permet de considérer, sur la figure 4.11, que S_1 et S_2 sont connexes, S_2 et S_3 sont connexes mais S_4 et S_5 sont trop éloignés pour être connexes.

Les relations de *prolongement* et de *parallélisme* utilisées sont celles définies dans [Gaio, 1994] et [Gaio *et al.*, 1994]. Mauro Gaio a établit, d'une part à partir de données psycho-expérimentales, d'autre part à partir d'expériences sur des figures à illusion perceptive puis sur des images de synthèse, que, pour prolonger un segment (que l'on considère comme étant le segment solliciteur d'un prolongement) en un autre, il fallait non seulement que l'angle entre les deux segments de droite soit inférieur à 16° mais aussi que le segment susceptible d'être un prolongement soit dans une certaine zone définie en fonction du segment solliciteur d'un prolongement. Cette zone (appelée "zone d'action du segment solliciteur") est définie par les formules suivantes :

$$|y| < a\lambda \left(\frac{x-\lambda}{\lambda}\right)^{1/3} + b \ si \ x \geq \lambda \tag{4.1}$$

$$|y| < a\lambda \left(\frac{x}{\lambda}\right)^{1/3} + b \ si \ x \leq 0 \tag{4.2}$$

où λ est la longueur en pixels du segment qui sollicite un prolongement, x et y sont les coordonnées, dans le repère du segment solliciteur, de n'importe quel point du segment candidat à un prolongement. L'expérimentation sur des images de synthèse a permis à M. Gaio de fixer la valeur de **a** à 0.05 et celle de **b** à 4 pixels.

Cette définition de la relation de prolongement permet de déterminer, comme prolongement possible pour le segment O_1E_1 de la figure 4.12 (la figure est donnée dans le repère défini à partir de O_1E_1 et les unités sont en pixels), le segment O_3E_3. Par contre, le segment O_4E_4, étant hors de la zone d'action de O_1E_1 et malgré un angle inférieur à 16° ne peut être un prolongement de ce dernier. De même, le segment O_2E_2, formant un angle supérieur à 16° ne peut pas être un prolongement pour O_1E_1. Dans le cas de la figure 4.11, S_1 se prolonge en S_2, S_2 se prolonge en S_1 et en S_3, S_4 se prolonge en S_5. Par contre, S_5 est trop petit pour inclure S_4 dans sa zone de d'action.

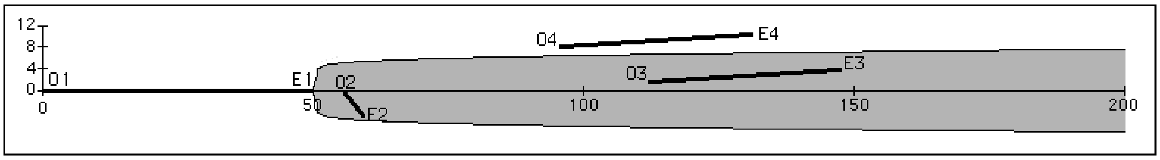


FIG. 4.12 - *Zone d'action du segment* O_1E_1 *suivant la définition de M. Gaio (la figure est donnée dans le repère défini à partir de* O_1E_1 *et les unités sont en pixels)*

En ce qui concerne la relation de parallélisme, Mauro Gaio considère que deux segments ont une relation de parallélisme si ces segments sont face à face et si l'angle entre les deux segments de droite est inférieur à 3°. Sur la figure 4.11, S_1 et S_4 sont parallèles, de même que S_2 et S_4. Par contre, les segments S_3 et S_5 forment un angle trop important pour pouvoir être considérés comme parallèles. De même, les segments S_3 et S_4 ne peuvent être parallèles car ils sont trop décalés.

Les relations de connexité et de parallélisme sont symétriques (si S_i est en relation avec S_j alors S_j est en relation avec S_i) mais la relation de prolongement ne l'est pas du fait de la définition donnée pour la zone d'action.

4.2.2 Fonctionnement

Le fonctionnement de ce module, directement basé sur la définition d'une route, est le suivant :

```
-  Initialisation : créer une ligne par segment existant
-  Rechercher les bandes minimales
-  Regrouper les bandes en morceaux de route
-  Réunir les morceaux connexes en ensembles
-  Regrouper les ensembles en routes
-  Déterminer les intersections entre routes
```

Les parties que nous allons détailler plus amplement maintenant sont celles qui, à notre avis, présentent le plus d'intérêt : la recherche des bandes minimales et la recherche des intersections.

La recherche des bandes

Nous avons choisi d'effectuer la recherche des bandes minimales en relâchant progressivement les contraintes sur les relations de prolongement et de parallélisme entre deux segments. Le fait de relâcher progressivement les contraintes permet de prendre en premier ce qui est le plus significatif et évite de prendre trop de bandes en considération (la recherche, pour une ligne donnée, continue avec des contraintes relâchées uniquement si aucune bande n'a été trouvée pour cette ligne).

En ce qui concerne la relation de parallélisme, la contrainte porte sur l'angle maximum que doivent former les deux segments pour être mis en relation. L'angle retenu dans [Gaio, 1994] est de 3°. Cependant, comme l'a montré Etemadi cet angle peut être porté à 10° [Etemadi *et al.*, 1991]. Le relâchement de contraintes sur la relation de parallélisme tient compte de ces remarques et consiste à augmenter progressivement l'angle de tolérance de 0° à 10°.

Pour la relation de prolongement, les contraintes portent à la fois sur l'angle de tolérance entre les deux segments susceptibles d'être mis en relation et sur la zone dans laquelle un segment quelconque peut établir une relation de prolongement. Dans [Gaio, 1994], l'angle retenu est de 16° et la zone d'action est définie par les équations 4.1 et 4.2 (page 49) Nos images étant moins précises que celles utilisées par M. Gaio, la zone d'action qu'il utilise était pour nous trop restrictive : un certain nombre de segments n'étaient pas considérés comme prolongements d'un autre à cause d'une faible partie sortant de cette zone. Pour cette raison, nous avons décidé de l'agrandir. Pour cela, par expérimentation sur nos schémas, nous avons fixé 0.1 (au lieu de 0.05) comme valeur maximale de a. De plus, suivant M. Gaio, un segment doit être entièrement compris dans la zone d'action pour former un prolongement acceptable. Toujours pour des raisons d'imprécision de nos schémas (imprécision due à la numérisation), nous avons redéfini une partie de cette zone : nous considérons que tout segment dont une des extrémités se trouve dans la zone de 4 pixels de haut et respectant la condition d'orientation, est susceptible d'être un prolongement même si son autre extrémité sort de la zone d'action globale du segment solliciteur (cf. figure 4.13).

Dans la figure 4.13 nous indiquons les différentes zones d'action du segment $\texttt{O}_1\texttt{E}_1$ (la figure est donnée dans le repère défini à partir de $\texttt{O}_1\texttt{E}_1$ et les unités sont en pixels). Sur cette figure, nous pouvons voir que les segments $\texttt{O}_3\texttt{E}_3$ et $\texttt{O}_2\texttt{E}_2$, qui ne sont pas considérés comme des prolongements possibles pour $\texttt{O}_1\texttt{E}_1$ d'après M. Gaio, sont, pour nous, des prolongements possibles pour $\texttt{O}_1\texttt{E}_1$. Le relâchement de contraintes sur la relation de prolongement tient compte des remarques faites précédemment et consiste à augmenter progressivement la valeur de a de 0 à 0.1.

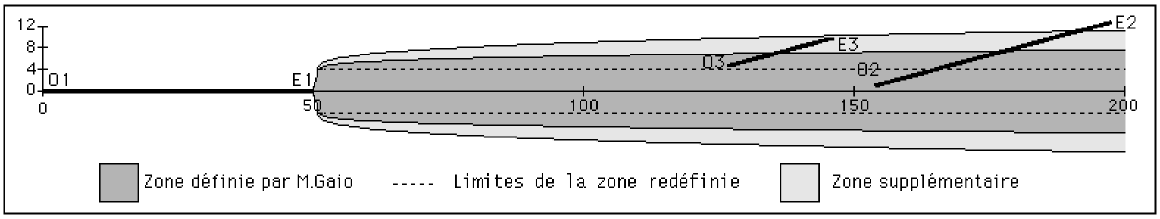


FIG. 4.13 - *Zone d'action du segment* $\texttt{O}_1\texttt{E}_1$ *telle que nous l'utilisons (la figure est donnée dans le repère défini à partir de* $\texttt{O}_1\texttt{E}_1$ *et les unités sont en pixels)*

La recherche des bandes s'effectue à l'aide de l'algorithme suivant :

Recherche des bandes minimales

- **Pour chacun** des niveaux N de contraintes sur le parallélisme et **Tant qu'** il existe au moins une bande dont les deux bords sont de longueurs très différentes **Faire**
  - déterminer les bandes de niveau N
  - **Pour chacun** des niveaux M de contraintes sur le prolongement **Faire**
    - déterminer les prolongements possibles de niveau M
    - **Pour chacune** des bandes $\mathcal{B}_1$ **Faire**
      - **Si** $\mathcal{B}_1$ a deux bords de longueurs très différentes **Alors**
        - déterminer b le plus petit des bords de $\mathcal{B}_1$
        - déterminer B le plus grand des bords de $\mathcal{B}_1$
        - déterminer les bandes $\mathcal{B}_2$ et $\mathcal{B}_3$ (situées aux extrémités de $\mathcal{B}_1$) ayant en commun avec $\mathcal{B}_1$ le bord B et telles que leur autre bord soit un prolongement possible de b de niveau M
        - fusionner $\mathcal{B}_2$ et $\mathcal{B}_3$ avec $\mathcal{B}_1$

Cet algorithme nécessite de savoir déterminer, d'une part, les bandes de niveau N, d'autre part, les prolongements possibles de niveau M et utilise les algorithmes suivants :

```
Déterminer les bandes de niveau N
- Pour chacune des lignes I Faire
  | - Pour chacune des lignes J Faire
  |   | - Si I et J sont différentes
  |   |     et I et J ne forment pas déjà une bande de niveau ≤ N
  |   |     et sont parallèles au niveau N Alors
  |   |     | - former une bande à partir de I et de J

Déterminer les prolongements possibles de niveau M
- Pour chacune des lignes I Faire
  | - Pour chacune des lignes J Faire
  |   | - Si I et J sont différentes
  |   |     et I n'a pas déjà de prolongement de niveau ≤ M
  |   |     et J est un prolongement de I au niveau M Alors
  |   |     | - mémoriser J comme prolongement possible de I
```

La recherche des intersections

La recherche des intersections obéit au principe suivant : tout d'abord, déterminer tous les points de jonction entre deux bords de route (il existe un point de jonction entre deux lignes si ces deux lignes n'appartiennent pas à la même route et ont une relation de connexité) puis les regrouper par intersection en "suivant" les points de jonction par l'intermédiaire de leurs vis-à-vis (le vis-à-vis d'un point P étant le point se trouvant en face P, sur l'autre bord de la route - cf. figure 4.14). La recherche des points de jonction (comme la construction des intersections) ne s'effectue qu'en regardant les extrémités de chacun des ensembles composant une route. En effet, les morceaux de route se trouvant dans un même ensemble étant liés par la relation de connexité, ils ne peuvent donc pas être séparés par une intersection. La recherche des points de jonction (comme la construction des intersections) s'effectue donc à partir du premier et du dernier morceau de chacun des ensembles de chacune des routes.

Deux lignes peuvent avoir une relation de connexité sans pour autant avoir un point commun (cf. figure 4.15). Pour cette raison, un point de jonction entre la ligne L et la ligne L_1 est en fait composé de deux points : un point P sur L et un point P_1 sur L_1.

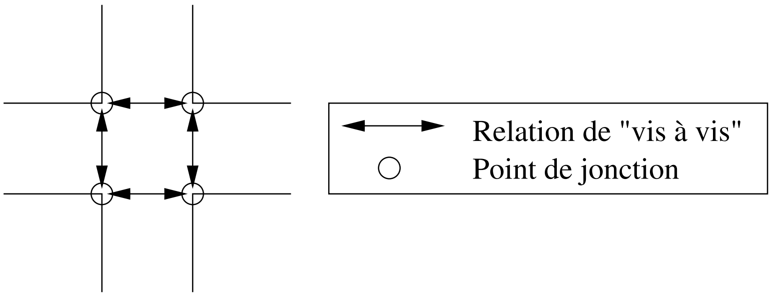


FIG. 4.14 - *Notion de vis-à-vis*

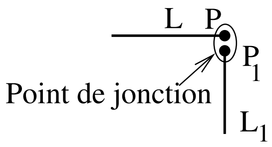


FIG. 4.15 - *Représentation d'un point de jonction*

La recherche des intersections s'effectue donc à l'aide des algorithmes suivants :

Déterminer les points de jonction

- **Pour chacune** des routes **Faire**
  - **Pour chacun** des ensembles de la route **Faire**
    - déterminer les points de jonction pour chacun des bords du premier morceau de l'ensemble
    - déterminer les points de jonction pour chacun des bords du dernier morceau M_d de l'ensemble
    - **Si** il existe une jonction en P_1 et telle que P_2 le vis-à-vis de P_1 n'ait pas de jonction **Alors**
      - créer une jonction sur P_2 ayant comme vis-à-vis P_1 (le vis-à-vis d'un point de jonction doit être un point de jonction)

Déterminer les points de jonction pour la ligne L

- **Tant qu'** il existe des lignes L_1 à distance nulle de L et n'appartenant pas à la même route que L **Faire**
  - **Si** L_1 appartient à une route **Alors**
    - ajouter une jonction à L

Déterminer les intersections

- **Pour chacune** des routes **Faire**
  - **Pour chacun** des ensembles de la route **Faire**
    - déterminer les intersections au niveau du premier morceau de l'ensemble
    - déterminer les intersections au niveau du dernier morceau de l'ensemble

Déterminer les intersections au niveau d'un morceau de route M

- **Si** il existe une jonction libre J (non intégrée dans un carrefour) sur M **Alors**
  - créer une nouvelle intersection I
  - ajouter J à la liste des jonctions de I
  - déterminer, à partir de J, les autres points de I

Déterminer, à partir d'une jonction J*, les autres points de l'intersection* I

- déterminer la jonction J_1 étant "en face" de J
- déterminer, à partir de J_1, les autres points de I

4.3 Catégorisation des éléments fermés

La catégorisation des éléments fermés a pour but de déterminer le type des objets, autres que les routes, existant sur l'image. Les schémas présents dans les constats d'assurance n'étant pas normalisés, les différents auteurs des schémas n'utilisent pas tous la même représentation pour un même objet (cf. Annexe A.1, exemples de types “généraux” de dessins de voiture que nous avons recueillis sur un ensemble de 64 schémas). De plus, du fait de la réalisation de ces dessins (“à la main”) et des différents traitements bas-niveau réalisés avant d'effectuer la catégorisation, une reconnaissance des objets complexes du style de celle envisagée dans [Tombre, 1987] (reconnaissance de polygones particuliers comme les rectangles, les carrés, les losanges, les triangles etc ... basée sur le nombre de segments et les relations entre les segments composant l'élément) ne pouvait convenir. En effet, comme le montre la figure 4.16, un élément qui pourrait être assimilé à un rectangle n'est pas nécessairement composé de 4 segments 2 à 2 parallèles et de même longueur. Nous nous sommes donc orientés vers d'autres systèmes de catégorisation et avons, dans un premier temps,choisi d'effectuer cette catégorisation à l'aide d'un réseau connexioniste à rétropropagation d'erreur. Cette technique nous semblait appropriée à notre problème du fait qu'elle nous permettait de prendre en compte non seulement les caractéristiques "locales" des segments composant l'élément mais aussi des caractéristiques morphologiques "globales" de l'élément.

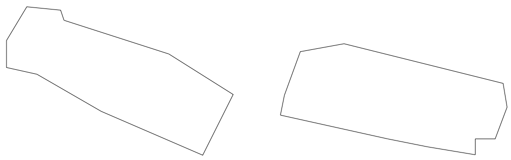


FIG. 4.16 - *Exemples d'éléments de type "rectangle"*

Le premier réseau mis en place lors de mon DEA ([Poirier, 1993] et [Poirier, 1994]) effectuait un découpage de l'ensemble des éléments en deux catégories : *véhicule* ou *cercle* (un cercle correspondant en général à un centre de rond-point) et travaillait directement sur les éléments fermés extraits de l'image. Il n'utilisait que deux caractéristiques morphologiques qui étaient : l'élongation et la compacité.

L'*élongation* est une caractéristique qui permet d'apprécier l'allure globale d'un élément. Pour un rectangle, elle correspond au rapport de la longueur du rectangle avec sa largeur. Pour étendre ce concept à des éléments de forme quelconque, il faudrait pouvoir déterminer le rectangle englobant un élément et dans ce cas, nous pourrions alors assimiler l'élongation du rectangle englobant à celle de l'élément proprement dit. La procédure que nous avons choisie pour déterminer le rectangle englobant passe par l'évaluation de la matrice centrale d'inertie de l'élément. Les vecteurs propres de cette matrice correspondent aux directions des axes principaux d'inertie suivant lesquels l'objet présente le plus fort et le plus faible moment d'inertie [Postaire, 1987]. Nous avons pu ainsi définir un rectangle dont les côtés sont parallèles à ces vecteurs puis nous en avons calculé l'élongation.

La *compacité*, définie à partir de la surface et du périmètre par : $\boxed{c = 4\pi * s/p^2}$, exprime le rapport entre la surface définie par le contour de l'élément fermé et celle d'un cercle ayant le même périmètre. Elle varie donc, de la valeur 1 pour un élément de forme circulaire à la valeur 0 pour un élément en forme de segment de droite; elle permet ainsi de situer la forme d'un élément entre ces deux formes extrêmes.

Ce réseau comportait deux unités d'entrée (une par caractéristique), deux unités de sortie ainsi qu'une unité cachée.

Cette première catégorisation posait tout d'abord le problème de la catégorisation des petits éléments fermés présents dans l'image : ces éléments, s'ils sont isolés, ne sont pas à prendre en compte (ce sont des résidus des traitements bas niveau), par contre, comme le montre la figure 4.17, si ces éléments sont en contact avec d'autres (grands ou petits) ils peuvent former un cercle ou un véhicule. La solution mise en place pour résoudre ce problème consiste à fusionner les petits éléments fermés avec ceux auxquels ils sont en contact et à catégoriser le résultat de cette fusion.

FIG. 4.17 - *Véhicules formés de petits éléments ou d'un grand et d'un petit (éléments extraits d'un schéma réel)*

Le second problème rencontré lors de l'utilisation de cette première catégorisation vient du fait qu'une voiture peut ne pas être représentée par un unique élément fermé : elle peut être représentée par un triangle et un rectangle. Ce problème m'a amenée à introduire une nouvelle classe (celle des triangles; classe qui pourra être aussi utilisée pour classer les panneaux de signalisation) et à effectuer la fusion décrite ci-dessus non plus uniquement pour les petits éléments fermés mais pour tous les éléments fermés. La mise en place de la fusion des éléments quelle que soit leur taille nécessite la reconnaissance des "objets de type inconnu" (ces objets appartiennent à la catégorie "inconnu"), c'est-à-dire des objets qui ne sont ni des voitures ni des cercles. Cette reconnaissance, comme la montre la figure 4.18, est nécessaire pour éviter de fusionner deux voitures qui se touchent. Dans ce cas, le niveau de fusion pris en compte (qui détermine le classement de l'objet) pour un élément est le dernier (s'il existe) qui a donné une classification différente de celle représentant les "objets inconnus".

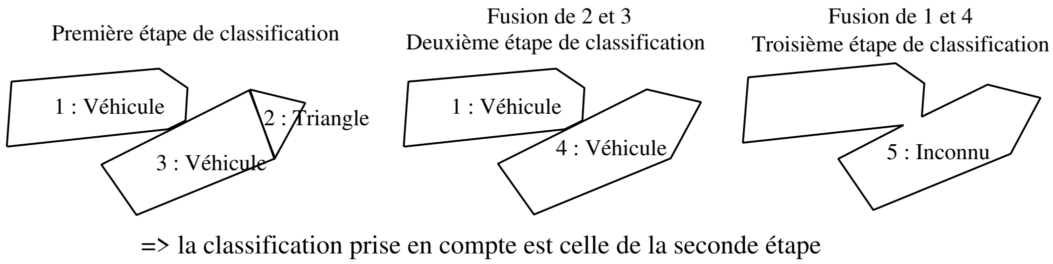


FIG. 4.18 - *Intérêt et utilisation de la classe "inconnu"*

La classe véhicule telle qu'elle était déterminée par la première classification mise en place était trop vaste : elle contenait aussi bien les véhicules orientés (rectangle avec une pointe) que ceux qui ne l'étaient pas (rectangle sans pointe) ce qui entraînait, pour un véhicule dessiné sans orientation précise, la transmission, au sous-système "construction de la scène", d'une orientation erronée. Comme il est souhaitable de ne fournir à ce sous-système que des données certaines, j'ai choisi de modifier la signification associée à la classe véhicule (celle-ci correspond uniquement aux véhicules dont l'orientation est certaine) et d'ajouter une classe supplémentaire contenant les véhicules sans orientation (la classe des rectangles).

Pour pouvoir mettre en place une catégorisation qui nous permette de classer les objets dans les catégories suivantes : petit, cercle, rectangle, triangle, véhicule (rectangle avec une pointe) et inconnu, nous devions utiliser de nouvelles caractéristiques morphologiques. Du fait de l'irrégularité de nos dessins, nous ne pouvions prendre en compte uniquement des données comme le nombre de segments ou les relations entre segments. Par contre, nous avons déterminé un certain nombre de caractéristiques qui peuvent être utilisées (en plus de l'élongation et de la compacité déjà utilisées) : la dispersion, le rapport entre la surface délimitée par le contour de l'élément et la surface du rectangle englobant, les profils de l'élément, le nombre de segments de l'élément et enfin la surface de l'élément.

La *dispersion* permet de caractériser la distribution de l'ensemble des points qui constituent l'objet surfacique par rapport au moment d'inertie. Elle s'exprime par : $\boxed{s/(\pi * detI)}$ (I étant la matrice centrale d'inertie de l'élément).

Le *rapport de similitude* qui correspond au rapport entre la surface délimitée par le contour de l'élément et la surface du rectangle englobant et qui permet de caractériser l'élément par rapport à un rectangle : plus la valeur de ce rapport est proche de 1, plus l'élément ressemble à un rectangle.

Les *profils* de l'élément permettent de déterminer si celui-ci a ou non une "pointe"; cette indication nous permet entre autre de déterminer si l'élément a une orientation propre ou non (cela permet de faire la distinction entre véhicule orienté et véhicule non orienté). Les valeurs prises en compte sont une caractérisation des profils gauche et droit de l'élément pris en compte (un profil plat est proche de 0, un profil pointu vers l'extérieur est proche de 0.5 alors qu'un profil pointu vers l'intérieur est proche de -0.5). Ces profils sont pris par rapport à l'axe d'inertie suivant lequel l'élément présente le plus fort moment d'inertie (cf. figure 4.19).

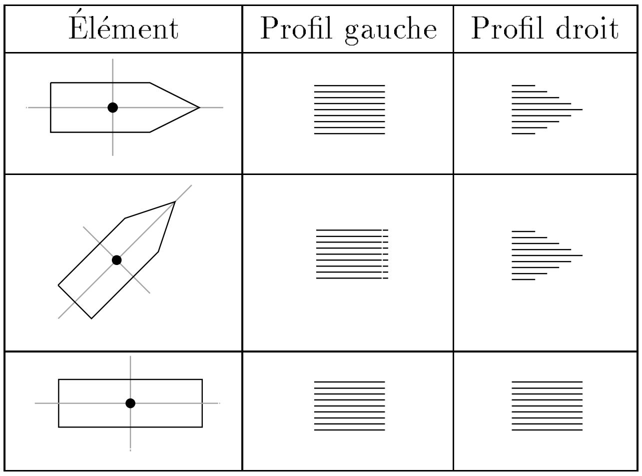


FIG. 4.19 - *Exemples de profil*

Le *nombre de segments* de l'élément est calculé à partir du contour correspondant à l'élément à caractériser. Ce contour est alors fortement lissé afin de ne garder que les changements de direction importants.

Pour mettre en place une telle classification, nous avons dans tout d'abord augmenté la taille du réseau à rétropropagation d'erreur utilisé dans la première classification. Ce nouveau réseau comportait 7 cellules d'entrée (autant que de caractéristiques utilisées), 6 cellules de sorties (autant que de catégories désirées) ainsi qu'un nombre variable de cellules cachées (variable suivant les essais réalisés) allant de 5 à 8. L'échantillon d'apprentissage était composé d'environ 300 éléments qui étaient soit "fabriqués" (éléments typiques comme un rectangle bien formé, etc.), soit extraits du corpus de schémas et essentiellement des schémas de constats réels et des schémas que nous avions nous même dessinés. Le premier problème que nous avons rencontré a concerné l'apprentissage : celui-ci était très long (plusieurs heures, quel que soit le réseau utilisé) pour un résultat peu concluant (seulement 80% des éléments étaient bien "appris"). Ces différents réseaux ont été testés sur un échantillon de test composé d'environ 700 éléments extraits du corpus obtenus dans le cadre de l'expérience réalisée au GREYC et les résultats obtenus étaient peu concluants : seulement 60% des éléments étaient correctement classés et aucune classe n'était entièrement reconnue de façon satisfaisante (alors qu'il est relativement facile pour un être humain de savoir, en fonction de sa surface, si un élément est petit, le réseau était incapable de faire une telle distinction). La complexité du réseau (et surtout celle de l'interprétation des résultats obtenus en fonction des poids des différents liens) et de l'amélioration d'un tel réseau nous a conduits à envisager une autre méthode de classification.

La méthode de classification pour laquelle nous avons opté[33] consistait à utiliser les arbres de décision [Quinlan, 1986], [Crémilleux, 1996] et [Crémilleux & Robert, 1997]. Pour cette seconde méthode de classification, les échantillons d'apprentissage et de test utilisés ont été les mêmes que ceux qui avaient servi à la mise au point du second réseau (300 éléments "fabriqués" ou extraits du corpus de schémas réels et des schémas que nous avions nous même dessinés pour l'apprentissage; 700 éléments extraits des schémas réunis lors de l'expérience pour l'échantillon de test). Comme le montre la table 4.1, les résultats obtenus par cette seconde méthode sont globalement satisfaisants (seulement 5,6 % d'erreur de classification) et de plus, ce système de classification nous a permis, entre autres, de nous apercevoir que deux de nos caractéristiques (l'élongation et la dispersion) étaient a priori inutiles. Il est vrai, qu'en ce qui concerne notre corpus, la dispersion et la compacité font quasiment double emploi et que les différentes classes peuvent être composées d'éléments d'élongation variable.

		Classification obtenue						Pourcentage
		(a)	(b)	(c)	(d)	(e)	(f)	d'erreur
classification voulue	(a) : inconnu	100	3			6		8,2
	(b) : rectangle		174	11		4		7,9
	(c) : véhicule		6	51		8		21,5
	(d) : cercle				21	1		4,5
	(e) : triangle					67		0
	(f) : petit						240	0
Pourcentage d'erreur		0	4,9	17,7	0	22,1	0	5,6

TAB. 4.1 - *Résultats de la classification*

Les résultats obtenus par cette classification montrent cependant que les erreurs sont relativement importantes pour les classes rectangle, véhicule et triangle (22% des éléments classés comme triangle n'en sont pas). En regardant l'arbre de décision généré lors de la phase d'apprentissage (cf. Annexe A.3), on s'aperçoit que les classes posant problème utilisent beaucoup de critères et essentiellement les profils gauche et droit de l'élément contrairement aux classes "petit" et "cercle" qui n'utilisent qu'un seul critère chacune (la surface pour la classe "petit", la compacité pour la classe "cercle"). Il semble donc qu'un arbre de décision, tout en donnant de meilleurs résultats qu'un réseau à rétropropagation, ne soit pas totalement adapté à notre problème. Cependant, comme l'ont montré [Katsuhiro & Shogo, 1995] et [Tanaka *et al.*, 1995], chacune

33. Nous remercions Bruno Cremilleux qui nous a conseillés sur ce point.

des méthodes de classification que nous avons utilisées (réseau connexioniste à rétro-propagation d'erreur et arbre de décision) a ses propres avantages. Dans notre cas, l'avantage de l'arbre de décision par rapport au réseau de neurones est d'être capable de n'utiliser qu'un seul critère pour déterminer une classe (cas des classes "petit" et "cercle"), l'avantage d'un réseau à rétropropagation d'erreur est d'être capable d'utiliser simultanément un grand nombre de caractéristiques pour effectuer la classification et donc de classer correctement des éléments même quand la classification dépend de tout un ensemble de caractéristiques qui jouent toutes autant que les autres.

Il faudra donc mettre en place une nouvelle classification des éléments fermés grâce à un système hybride composé d'une part d'un arbre de décision et d'autre part d'un réseau de neurones. Ce nouveau système hybride devra utiliser les avantages de chacun des systèmes de classification en utilisant un arbre de décision pour "dégrossir" la classification (et, entre autres, déterminer les éléments des classes "petit" et "cercle") et un réseau de neurones pour affiner cette classification.

4.4 Mise en forme des résultats

Pour permettre à l'analyseur de schémas de pouvoir communiquer avec le sous-système de construction de la scène, nous avons choisi un formalisme de sortie commun aux deux sous-systèmes. Ce formalisme est basé sur un modèle géométrique dont une première description est donnée dans [Enjalbert & Victorri, 1994]. Ce formalisme a pour but de permettre au sous-système chargé d'effectuer la construction de la scène de traiter des problèmes de localisation ainsi que de compléter les valeurs par défaut. Les résultats fournis par le module d'extraction des routes et celui de catégorisation des éléments fermés sont sous forme de groupes de segments et ne sont donc pas directement exploitables pour effectuer la construction de la scène (ils ne sont pas exprimés dans un modèle géométrique). Il nous fallait donc "traduire" ces résultats afin de les rendre exploitables par le sous-système "construction de la scène".

4.4.1 Formalisme de sortie

Pour effectuer la traduction nous avons mis en place un formalisme de description capable de transmettre au sous-système "construction de la scène" l'ensemble de nos connaissances sur le schéma. Pour faire cette traduction nous nous basons sur un repère de base (appelé par la suite "repère absolu") qui correspond au repère mathématique (en bas à gauche de l'image).

Les routes

Nous avons choisi de représenter une route par la courbe équidistante de ses deux bords (nous appellerons cette courbe la *directrice* de la route), l'origine de cette courbe (qui sera considérée comme *l'origine* de la route) et la *largeur* de la route en tous points de la directrice. Une route peut être composée de plusieurs parties de forme différente (droite ou courbe), l'équation de la directrice est donc une suite d'équations correspondant à ces différentes parties. Pour chacune d'elles il faut préciser : l'équation du morceau de directrice correspondant, son origine et son extrémité. Pour représenter une directrice nous avons choisi d'utiliser des équations paramétrées en fonction de l'abscisse curviligne (symbolisée dans nos équations par s). Dans ce cas, l'origine (O) et l'extrémité (E) d'une partie de route sont représentées par leurs abscisses curvilignes (respectivement s_o et s_e) calculées à partir de O_r, l'origine de la route.

L'équation de directrice associée à une partie droite est une équation de droite paramétrée en fonction de s et est donc de la forme : $x = s * \cos(\beta_0) + x_{O_r}$, $y = s * \sin(\beta_0) + y_{O_r}$ où β_0 est l'angle orienté entre l'axe des abscisses du repère absolu et le segment considéré (cf. figure 4.20) et x_{O_r} et y_{O_r} sont les coordonnées de l'origine de la route dans le repère absolu. Pour caractériser une partie droite, il suffit donc de disposer de l'abscisse curviligne de son origine (s_o), celle de son extrémité (s_e) ainsi que l'angle β_0 (ce sont ces éléments qui seront fournis au sous-système de construction de la scène).

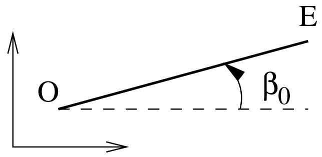


FIG. 4.20 - *Représentation d'une droite*

Nous avons choisi d'approximer les parties courbes par des arcs de cercle dont l'équation paramétrée est de la forme : $x = r * \cos(\beta) + c_x$, $y = r * \sin(\beta) + c_y$ où c_x et

c_y sont les coordonnées du centre du cercle dans le repère absolu, r le rayon du cercle et β l'angle orienté entre l'axe des abscisses du repère absolu et le point de l'arc de cercle d'abscisse curviligne s (β varie donc en fonction de s; cf. figure 4.21). β pouvant être obtenu à partir de β_o (angle, par rapport à l'axe des abscisses, du point d'abscisse curviligne s_o et pouvant être déterminé soit à partir de l'origine de la route, soit à partir du morceau de route précédent) et de s ($\beta = \frac{(s-s_o)}{r} + \beta_o$), pour caractériser une partie courbe, il suffit de donner l'abscisse curviligne de son origine (s_o), celle de son extrémité (s_e), les coordonnées du centre du cercle (c_x et c_y) ainsi que son rayon (r).

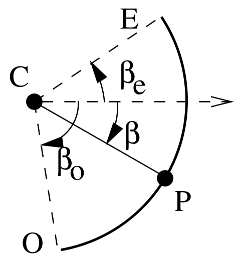


FIG. 4.21 - *Représentation d'une courbe*

Les ensembles formant une route peuvent être séparés par des “espaces vides” (correspondant en général aux intersections). Afin de décrire la route dans sa totalité nous avons choisi d'associer à ces “espaces vides” une équation. Pour caractériser une interruption de tracé, nous donnons, comme dans le cas d'une partie droite (parce que deux ensembles d'une même route sont reliés par une relation de prolongement), l'abscisse curviligne de son origine (s_o), celle de son extrémité (s_e) ainsi que l'angle β_0 caractérisant sa direction.

La largeur d'un morceau de route peut être approximée par une équation de la forme : $l = a * s + b$ et est donc caractérisée par les coefficients a et b.

Entre les deux bords d'une route il peut exister sur le schéma un certain nombre de lignes qui peuvent matérialiser soit une séparation de file de circulation, soit un terre-plein (la différence sera effectuée lors de la construction de la scène en fonction des connaissances relatives au monde de la route). Deux renseignements concernant ces lignes sont nécessaires pour le sous-système chargé d'effectuer la construction de la scène afin de faire la distinction entre les lignes délimitant les files et celles délimitant les terre-pleins : le type de la ligne (continue - discontinue, donné sous la forme Ligne *type*) ainsi que sa position par rapport à la directrice de la partie concernée. La distance entre une ligne donnée et la directrice de la route pouvant varier sur une même partie de route (voie qui s'élargit par exemple), pour donner la distance d'une ligne à la directrice nous utilisons une équation de la même forme que celle servant à préciser la largeur de la route. Cette distance est une distance "orientée" et l'orientation dépend de la direction de la directrice : la distance entre la directrice et une ligne pour une abscisse curviligne donnée est positive si le point situé à cette abscisse a une ordonnée positive dans le repère orthonormé direct associé à la directrice (cf. figure 4.22).

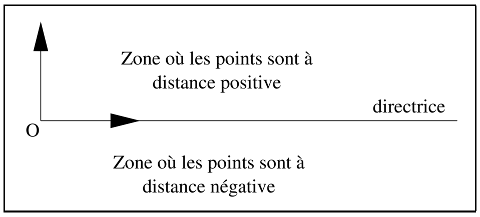


FIG. 4.22 - *Calcul de la distance à la directrice*

Les intersections

Une intersection peut être vue de deux manières différentes : unidimensionnelle et bidimensionnelle. L'aspect unidimensionnel consiste à préciser, pour chaque route arrivant ou sortant de l'intersection, la position de l'intersection par rapport à la route (en précisant la valeur de l'abscisse curviligne à laquelle on entre ou on sort de l'intersection). Pour donner une intersection sous son aspect unidimensionnel, nous indiquons donc, lors de la définition des routes, l'abscisse curviligne à laquelle débute l'intersection I_n ainsi que celle à laquelle elle se termine.

L'aspect bidimensionnel consiste à donner l'ensemble des points formant l'intersection et les routes auxquelles ils appartiennent et ce en tournant dans le sens trigonométrique autour de l'intersection. Pour donner une intersection sous son aspect bidimensionnel, nous précisons donc le nom de l'intersection (I_n), les coordonnées de son centre (c_x et c_y), ainsi que la liste des points qui la composent. Pour chacun de ces points nous donnons, non seulement ses coordonnées dans le repère lié à l'intersection (repère ayant comme origine le centre de l'intersection et comme axes ceux du repère absolu), mais aussi les deux routes sur lesquelles se trouve ce point. Dans certains cas, nous avons été amenés à ajouter des points à l'intersection (points P_2 et P_3 de la figure 4.23) : ces cas sont assez simples, ils correspondent à des ensembles de morceaux de route n'ayant pas de vis-à-vis dans l'intersection prise en considération. Dans ce cas, les deux routes sur lesquelles se trouve le point sont identiques.

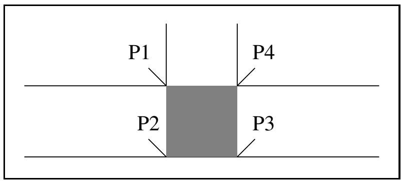


FIG. 4.23 - *Points formant une intersection*

Les véhicules

A partir des schémas, on peut extraire la position, éventuellement l'orientation ainsi que la catégorie du véhicule (voiture, camion, ...- extraite par le module de catégorisation des éléments fermés). Concernant un véhicule, la position absolue n'est pas très intéressante, il est plus utile d'avoir la position relative à la route (et/ou à l'intersection) sur laquelle il se trouve.

Les équations représentant les routes étant des équations paramétrées par s, il faut, pour repérer un véhicule par rapport à une route, l'abscisse curviligne à laquelle il se trouve ainsi que la distance à la directrice. La distance, comme pour les lignes se trouvant entre les deux bords d'une route, est une distance "orientée". Pour repérer un véhicule, il suffit d'un point de ce véhicule. Nous avons choisi de prendre le centre de gravité. La position du véhicule étant donnée par rapport à la directrice, nous avons choisi de donner aussi la direction du véhicule par rapport à celle de la directrice. Pour chaque véhicule nous indiquons : sa catégorie (voiture, camion, ...), la route sur laquelle il se trouve, l'abscisse curviligne à laquelle se trouve son centre de gravité, sa distance par rapport à la directrice, sa direction par rapport à cette dernière ainsi qu'une valeur indiquant si cette direction tient lieu d'orientation ou non. Cette dernière valeur nous permet de spécifier si le véhicule est à considérer comme orienté (c'est-à-dire s'il a été reconnu comme tel lors de la classification) ou non. Ces différents paramètres sont représentés par la figure 4.24.

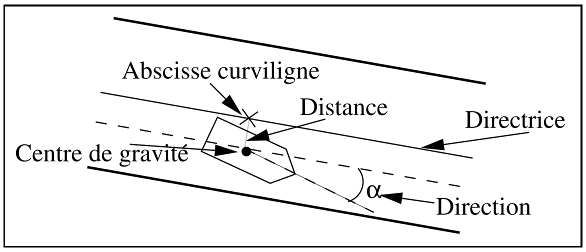


FIG. 4.24 - *Éléments permettant de positionner un véhicule par rapport à une route*

La position d'un véhicule par rapport à l'intersection sur laquelle il se trouve est nécessaire dans le cas où toutes les routes arrivant à l'intersection (ou en sortant) ne le traversent pas (il est alors impossible de positionner le véhicule par rapport aux routes qui ne traversent pas l'intersection). Dans le cas des ronds-points on peut même être incapable de positionner le véhicule par rapport à une route (cf. figure 4.25 : le véhicule n'appartient à aucune des trois routes).

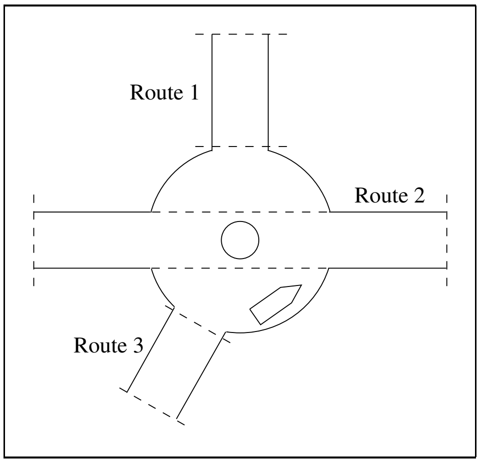


FIG. 4.25 - *Positionnement d'un véhicule*

Pour positionner un véhicule par rapport à une intersection nous utilisons la représentation bidimensionnelle de l'intersection et nous le positionnons par rapport au centre de l'intersection. Pour chaque véhicule se trouvant sur une intersection, nous indiquons : sa catégorie, l'intersection sur laquelle il se trouve, les coordonnées de son centre de gravité dans le repère lié à l'intersection, sa direction dans ce même repère ainsi que, comme dans le cas précédent, une valeur indiquant si cette direction tient lieu d'orientation ou non.

Il peut exister des schémas où les données sont incomplètes (schémas sur lesquels il manque quelques lignes de signalisation voire une route complète; cf. figure 4.26). Dans ce cas, l'analyseur de schémas doit tout de même fournir des indications sur la position du véhicule car celles-ci permettront au sous-système chargé d'effectuer la construction de la scène de déterminer une route probable. En effet, le fait de savoir où se trouve un véhicule dans le schéma peut permettre au sous-système de construction de la scène de déterminer la zone dans laquelle se trouve une éventuelle route et l'aider ainsi à construire sa scène et une représentation visuelle de celle-ci. L'analyseur de schémas fournit alors, pour chacun de ces véhicules, les coordonnées de son centre de gravité, sa direction (ces différentes valeurs étant données dans le repère absolu), sa catégorie ainsi que, comme dans les cas précédents, une valeur indiquant si la direction tient lieu d'orientation ou non.

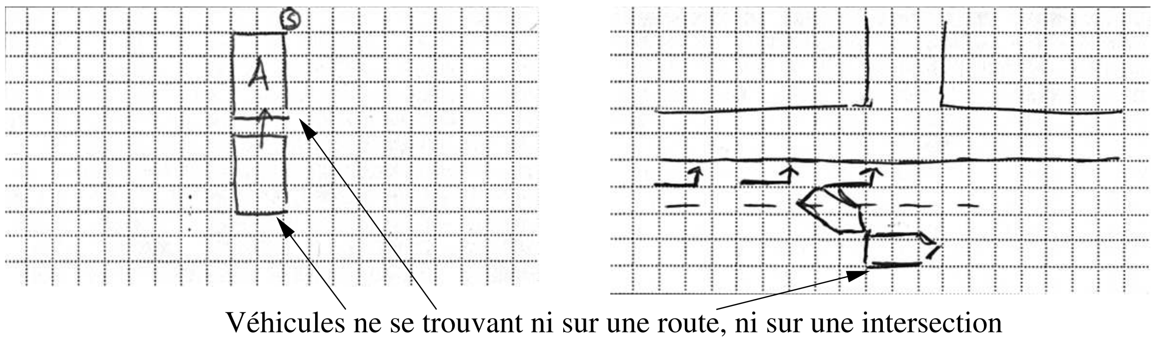


FIG. 4.26 - *Schémas où les routes sont incomplètes*

tableau récapitulatif

La table 4.2 donne un récapitulatif du formalisme de sortie de l'analyseur de schémas tel qu'il vient d'être défini.

Morceau de route droit	:	$s_o < s < s_e \quad Droite(\beta_0)$
Morceau de route courbe	:	$s_o < s < s_e \quad Courbe((c_x, c_y), r)$
Interruption de tracé	:	$s_o < s < s_e \quad Interruption(\beta_0)$
Largeur d'un morceau de route ou indication d'une distance	:	$Largeur(a, b)$
Indication d'une ligne	:	Ligne *type*
Intersection (aspect unidimensionnel)	:	s_o *début/fin* de l'intersection I_n
Intersection (aspect bidimensionnel)	:	Intersection I_n Centre (c_x, c_y) Points $\wp$
Point d'une intersection (élément de $\wp$)	:	$((\mathrm{R}\ \theta), \mathtt{R_i}, \mathtt{R_j})$
Véhicule placé sur une route	:	$Catégorie(\mathrm{R}, s, d, \beta, o)$
Véhicule placé sur une intersection	:	$Catégorie(\mathrm{I}, (r, \theta), \alpha, o)$
Véhicule non positionné	:	$Catégorie((x, y), \alpha, o)$

TAB. 4.2 - *Récapitulatif du formalisme*

4.4.2 Interprétation

L'interprétation effectuée dans le cadre de ce module, consiste à "traduire" les résultats obtenus par le module d'extraction des routes et celui de catégorisation des éléments fermés dans le formalisme décrit ci-dessus. Elle se fait en trois étapes : construction des courbes relatives aux routes, construction des représentations des intersections puis calcul de la position des véhicules par rapport aux routes et/ou aux intersections.

Les routes

Pour effectuer l'interprétation des routes, il faut tout d'abord déterminer les parties droites et les parties courbes de la route. Les routes sont fournies par le module d'extraction des routes sous la forme d'un ou plusieurs ensembles de morceaux de route et sont donc représentées par des ensembles de segments droits : toute partie courbe existant sur un schéma sera partitionnée en morceaux de route qui auront été regroupés en un seul ensemble (ces différents morceaux sont de directions différentes mais sont connexes). La première étape de l'interprétation consiste donc à déterminer les ensembles susceptibles de représenter une partie courbe d'une route.

Pour déterminer si un ensemble forme une courbe, nous nous sommes basés sur le fait que, dans le cas d'un cercle, il existe une relation linéaire entre s et β ($s = R{*}\beta{+}s_o$ avec R rayon du cercle, β angle "parcouru" sur le cercle et s_o abscisse curviligne pour $\beta = 0$). Par conséquent un ensemble de morceaux de route peut être considéré comme courbe s'il existe une relation linéaire entre s et β (s et β étant calculés sur les milieux des segments équidistants des bords des composants de l'ensemble considéré; cf. figure 4.27). Pour déterminer s'il existe une relation linéaire entre s et β, nous calculons le coefficient de corrélation entre s et β : si la valeur absolue de celui-ci est proche de 1 alors la relation entre s et β est quasi linéaire et l'ensemble considéré peut être approximé par un cercle dont le rayon est obtenu grâce à la droite de régression de s en fonction de β.

Quand l'ensemble considéré est reconnu comme étant courbe, il faut ensuite déterminer le centre du cercle qui va servir à l'approximer (c'est le point se trouvant à une distance R du début et de la fin de la courbe). Pour cela, il faut choisir deux points situés sur les segments initiaux pour fixer l'origine et l'extrémité de la courbe. Nous considérons, pour faire ces choix, qu'avant et après chaque courbe il y a une partie

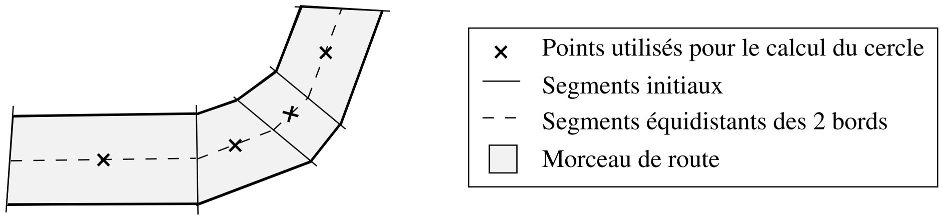


FIG. 4.27 - *Recherche de la présence d'un cercle*

droite (ce qui n'est pas très gênant puisqu'en général avant et après un virage une portion droite est dessinée). Nous prenons donc un point sur le premier segment comme origine de la partie courbe (point situé entre le milieu et l'extrémité en fonction de la longueur du segment : plus le segment est long, plus le point choisi sera proche de l'extrémité) et un point sur le dernier segment (point situé entre l'origine et le milieu en fonction de la longueur du segment). Le cercle ainsi placé n'est pas exactement tangent au reste de la courbe à ses extrémités, mais le fait qu'il passe obligatoirement par des points appartenant à d'autres directrices évite d'avoir des décalages trop importants par rapport aux directrices des autres parties de la route (décalages qui existent quand on se base sur les tangentes pour placer le cercle - cf. figure 4.28).

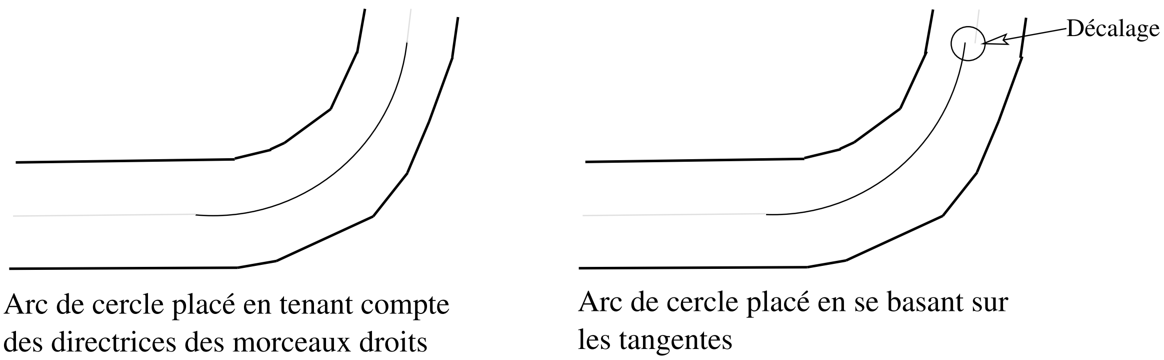


FIG. 4.28 - *Positionnement de l'arc de cercle*

Si l'ensemble considéré n'est pas reconnu comme étant une partie courbe, il faut alors considérer séparément chacun des morceaux qui le composent (car ceux-ci n'entretiennent pas de relation de prolongement) et associer à chacun d'eux une équation. Par définition un morceau de route contient des bandes connexes et sensiblement de même orientation (elles sont reliées par la relation de prolongement) et peut donc être assimilé à une partie droite. Par conséquent, à chaque morceau de route n'appartenant pas à une courbe, nous associons une équation de partie droite.

Chaque route extraite par le module d'extraction des routes est nommée afin de pouvoir être ultérieurement référencée, ce nom est maintenu par l'interprétation et est communiqué sous la forme "Route(*numéro*)" afin de faciliter les références.

Les intersections

Le module d'extraction des routes extrait du schéma les intersections et les donne sous la forme d'ensembles de points de jonction entre routes. Une intersection doit être donnée en sortie sous deux formes : unidimensionnelle et bidimensionnelle. Pour effectuer la représentation unidimensionnelle, il suffit de calculer l'abscisse curviligne de chaque point de l'intersection par rapport aux routes mises en cause. Pour effectuer la représentation bidimensionnelle, il suffit d'organiser l'ensemble des points de jonction de l'intersection dans le sens trigonométrique.

Les véhicules

L'interprétation des véhicules se fait en deux étapes :

1) Déterminer le centre de gravité et la direction (par rapport au repère absolu) du véhicule. Cette direction correspond à l'orientation du vecteur d'origine G (centre de gravité) et d'extrémité P (le point du véhicule le plus éloigné de G dans le repère associé au véhicule et construit à partir des vecteurs propres de la matrice d'inertie);

2) Rechercher les routes et les intersections sur lesquelles se trouve le véhicule et le positionner par rapport à ces éléments.

Comme pour les routes, il faut donner la dénomination des différents véhicules pour maintenir la cohérence, ceci est effectué par "Véhicule(*numéro*)".

Les parties courbes des routes posent un léger problème : du fait de l'approximation effectuée lors du calcul des parties courbes, la position de la route calculée ainsi que sa largeur ne sont pas en tout point exactes à celles de la route dessinée. A cause de ce problème, il peut arriver qu'un véhicule se trouvant sur la route soit considéré comme étant en partie en dehors (cf. figure 4.29, le véhicule dessiné est en partie en dehors de la route calculée alors qu'il est complètement sur la route dessinée). Pour résoudre ce problème, nous avons choisi de positionner les véhicules en tenant compte, non seulement de leur position sur le schéma initial, mais aussi du rapport largeur de la route dessinée/largeur de la courbe calculée (si la distance véhicule dessiné - "milieu" de la route dessinée est égale à 40% de la largeur de la route dessinée alors le véhicule sera positionné comme étant à une distance de la directrice calculée égale à 40% de la largeur de la route calculée).

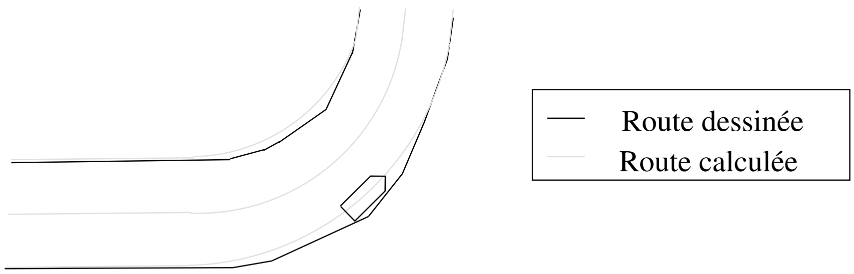
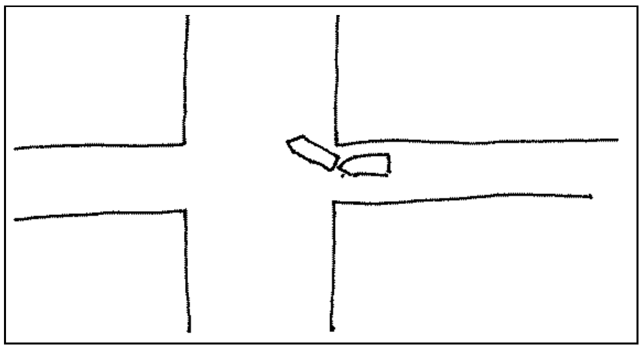


FIG. 4.29 - *Positionnement d'un véhicule dans une courbe*

Exemple

Nous allons maintenant donner et commenter le résultat d'une analyse tel qu'il est fourni par l'analyseur, c'est-à-dire, sous forme d'équations et de positionnements de véhicules. Le résultat fourni dans la table 4.3 est celui obtenu après analyse du schéma suivant :



Résultat	Commentaires
`Taille Image 550 290`	
`Nb routes : 2`	
`Route(0)`	*Définition de la première route*
`Origine (534, 151.5)`	*Origine de la route (repère absolu)*
`0<s<242 Droite(3.16) Largeur(0.005, 50.5)`	*Première partie : partie droite*
`242 Début de l'intersection I0`	*Début de l'intersection I0*
`242<s<379 Interruption(3.17)`	*Interruption de tracé (intersection)*
`379 Fin de l'intersection I0`	*Fin de l'intersection I0*
`379<s<532 Droite(3.18) Largeur(0.025, 50.5)`	*Seconde partie : partie droite*
`Route(1)`	*Définition de la seconde route*
`Origine (224.5, 5.5)`	*Origine de la route (repère absolu)*
`0<s<112 Droite(1.57) Largeur(0.044, 129)`	*Première partie : partie droite*
`112 Début de l'intersection I0`	*Début de l'intersection I0*
`112<s<168 Interruption(1.57)`	*Interruption de tracé (intersection)*
`168 Fin de l'intersection I0`	*Fin de l'intersection I0*
`168<s<282 Droite(1.54) Largeur(-0.026, 144)`	*Seconde partie : partie droite*
`Intersection I0 Centre 224 145`	*Intersection (aspect bidim.)*
`Points [ ((75.39 0.38) , R0, R1)`	*Ensemble des points composant*
`((75.39 2.76) , R1, R0)`	*l'intersection (les coordonnées*
`((74.24 3.58) , R0, R1)`	*sont données dans le repère de*
`((71.16 5.93) , R1, R0)]`	*l'intersection)*
`Vehicule(0)`	*Définition du premier véhicule*
`Voiture(R0, 212, -6.43 0.006)`	*Positionnement par rapport à R0*
`Vehicule(1)`	*Définition du second véhicule*
`Voiture(R0, 261, -19.26, 5.74)`	*Positionnement par rapport à R0*
`Voiture(R1, 160, -49, 1.057)`	*Positionnement par rapport à R1*
`Voiture(I0, 53.2024, 0.39, 2.628)`	*Positionnement par rapport à I0*

TAB. 4.3 - *Résultat de l'analyse (tel qu'il est fourni par l'analyseur)*

4.5 Implémentation

4.5.1 Extraction des routes

Pour pouvoir effectuer l'extraction des routes nous avions besoin de définir un certain nombre de structures permettant de représenter les routes et leurs composants (ensembles de morceaux de route, morceaux de route, etc.), les intersections (avec les points de jonction) et les équations. Les structures utilisées pour définir une route sont toutes basées sur la notion de ligne. Cependant, une ligne peut être soit un simple segment (représenté par ses deux extrémités), soit un ensemble de lignes résultant de la fusion de lignes reliées par la relation de prolongement (fusion réalisée lors de la fusion de bandes au moment de la recherche des bandes minimales). Une telle structure est facilement réalisable grâce à un langage objet du fait de la notion d'héritage : il faut avoir une classe représentant une abstraction générale de la notion de ligne et qui sert de base à deux classes représentant respectivement les lignes simples et les lignes composées. Nous avons donc utilisé, pour développer notre analyseur, un langage objet. Notre choix s'est porté sur le `C++`, d'une part pour des raisons de génie logiciel [Stroustrup, 1989], d'autre part parce que nous disposions d'une librairie d'opérateurs de traitements bas niveau développée dans ce langage (librairie développée au GREYC pour les besoins de la thèse de M. Gaio [Gaio, 1994] et qui contient tous les outils bas niveau - squelettisation, suivi de contours, etc. - dont nous avions besoin).

La classe `Ligne` que nous avons définie est composée de 5 attributs : deux points représentant les extrémités de la ligne, une liste de jonctions permettant de mémoriser les jonctions se trouvant à l'une ou l'autre des extrémités de la ligne et deux listes de pointeurs sur des lignes permettant de mémoriser d'une part, les lignes entretenant une relation de connexité, d'autre part les lignes entretenant une relation de prolongement. Cette classe représente à la fois la ligne générale et une ligne composée d'un seul segment en considérant comme extrémités de la ligne, les extrémités du segment. Ceci simplifie le graphe d'héritage car on n'utilise alors que deux classes : `Ligne` et `LigneComp`.

La classe `LigneComp` (qui hérite de `Ligne`) est composée, en plus des attributs de la classe `Ligne`, d'une liste de pointeurs sur des lignes représentant l'ensemble des segments regroupés. Cette liste est un objet de la classe `ListeDePtLigne` qui est implémentée, comme toutes les listes définies ultérieurement, à partir des listes GNU [34] et qui nous permet de regrouper en un même objet à la fois des instances de la classe `Ligne` et des instances de la classe `LigneComp`. Cette structure récursive nous permet, par la suite, de fusionner non seulement des lignes simples mais aussi des résultats de fusions précédentes. Dans le cas des instances de la classe `LigneComp`, les deux points correspondent au segment équivalant à l'ensemble des segments considérés (ce segment représente une approximation de la ligne abstraite). Ce segment nous permet d'effectuer plus rapidement un certain nombre de calculs entre autres au niveau des relations (prolongement, connexité, parallélisme) et est utilisé lors de la phase de mise en forme des résultats pour déterminer les équations liées aux divers morceaux de route. Il est calculé à partir de la droite de régression associée à l'ensemble des points pris en compte. Ce segment est, bien sûr, une approximation de l'ensemble des lignes mais du fait des relations entre ces différentes lignes (relations de prolongement), la différence avec le tracé réel est relativement faible et ne gêne en rien l'interprétation.

Une bande étant composée de deux lignes, nous avons défini la classe `Bande` comme ayant deux attributs qui sont des pointeurs sur des instances de la classe `Ligne` et qui représentent ses deux bords. Un morceau de route peut être considéré comme un ensemble ordonné de lignes dont deux lignes ont une fonction particulière (les bords). Cependant, pour permettre un accès plus rapide à ces deux lignes particulières, nous avons préféré les "sortir" de la liste. Nous avons donc défini une classe `MorceauRoute` ayant trois attributs : deux pointeurs sur des instances de la classe `Ligne` qui représentent les deux bords et un objet de la classe `ListeDePtLigne` qui représente les lignes se trouvant entre ces deux bords. Un ensemble de morceaux de route est représenté par une instance de la classe `EnsembleMorceauxRoute` qui est une liste d'instances de la classe `MorceauRoute`. Une route est représentée par une instance de la classe `Route` qui est une liste d'ensembles de morceaux de route à laquelle nous associons une origine et une liste d'équations représentant les différentes parties de la directrice associée à la route (ces deux derniers attributs seront mis à jour et utilisés lors de la mise en forme des résultats). Pour chacune de ces classes, nous avons défini un ensemble de méthodes permettant de déterminer les relations de connexité, parallélisme et prolongement. Les routes sont donc représentées par l'ensemble des structures et le graphe d'héritage donnés dans la figure 4.30.

34. Gnu's Not Unix - ensemble de librairies et de logiciels gratuits disponibles sur Internet et fonctionnant sur toute machine Unix [Gnu, 1992].

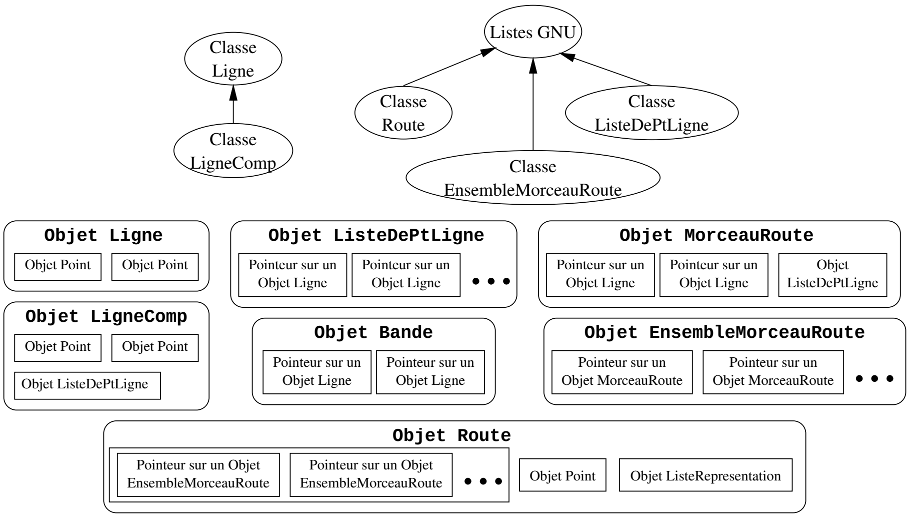


FIG. 4.30 - *Structure de données utilisée pour définir les routes*

Comme nous l'avons vu précédemment (cf. figure 4.15, page 54), un point de jonction entre deux lignes est en fait composé de deux points ayant chacun un vis-à-vis différent. Pour cette raison, nous avons défini, pour représenter les jonctions entre routes, d'une part une classe `PointDouble` permettant de représenter un point et son vis-à-vis (qui est lui même un pointeur sur une instance de la classe `PointDouble`) et de mémoriser la route sur laquelle se trouve ce point, d'autre part une classe `PointLigne` ayant comme attributs un pointeur sur une instance de la classe `PointDouble` et un pointeur sur une instance de la classe `Ligne` qui permet de mémoriser la ligne sur laquelle se trouve le point et enfin une classe `Jonction` ayant comme attributs deux instances de la classe `PointLigne`. Cette structure (dont la représentation est donnée figure 4.31) nous permet de savoir, pour toute jonction, les deux lignes mises en cause ainsi que les routes auxquelles elles appartiennent et les vis-à-vis des deux points considérés.

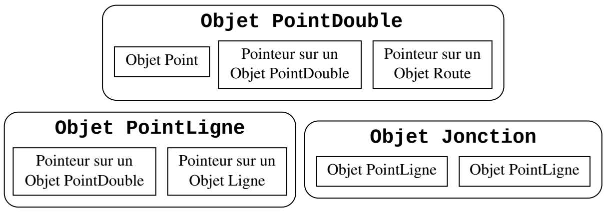


FIG. 4.31 - *Représentation des objets utilisés pour définir les jonctions*

4.5.2 Catégorisation des éléments fermés

Pour mettre en place le réseau à rétropropagation d'erreur utilisé, dans un premier temps, pour effectuer la classification des éléments fermés, nous avons utilisé les outils développés par R. Cazoulat [Cazoulat, 1991]. La seconde classification (arbre de décision) a été implémentée grâce à C4.5[35] [Quinlan, 1993]. Cependant, nous ne voulions pas intégrer le système de classification au reste du programme de façon à pouvoir le tester en dehors du reste de l'application et à pouvoir en changer facilement. C'est pour cette raison, que le module de catégorisation des éléments fermés de l'analyseur de schémas est un programme à lui tout seul qui communique avec le reste de l'application par un pipe Unix. De plus, ce mode de fonctionnement nous permet de pouvoir exécuter en parallèle l'extraction des routes et la classification des éléments fermés.

Les résultats fournis par le module de catégorisation (que ce soit un réseau ou un arbre de décision), sont mémorisés, au niveau du reste de l'application par l'intermédiaire d'instances de la classe `ContourClasse`. Cette classe a comme attributs, d'une part, un pointeur sur le contour à classer, d'autre part le résultat de la classification (un numéro représentant la classe). Cette structure de données sera ensuite utilisée par le module de mise en forme des résultats pour déterminer les véhicules et effectuer le positionnement de ces derniers.

4.5.3 Mise en forme des résultats

Comme nous l'avons vu dans la section 4.4 page 61, la mise en forme des résultats consiste à associer une ou plusieurs équations géométriques à chacune des routes déterminées, à compléter et à décrire les intersections incomplètes et à positionner les véhicules sur les routes et/ou les intersections. Pour effectuer l'implémentation de cette partie, nous avons associé à chaque route un ensemble d'équations (un objet de la classe `ListeRepresentation`) permettant de décrire chacun de ses morceaux. Pour cela nous devons donner, pour chaque partie de la route, non seulement l'équation de la directrice associée mais aussi l'équation de la largeur en chacun des points de la directrice et les équations donnant les positions des lignes se trouvant entre les deux bords de la route. Nous avons donc défini une classe de base `Representation` dont les attributs sont : un pointeur sur une équation représentant la directrice (pointeur

35. Ensemble de programmes permettant d'implémenter des arbres de décision.

sur une instance de la classe `Equation`), une équation correspondant à la largeur de la route (instance de la classe `EqLargeur`) et une liste d'équations donnant la position de chacune des lignes se trouvant entre les deux bords du morceau de route par rapport à la directrice (instance de la classe `ListeDeEqLargeur`).

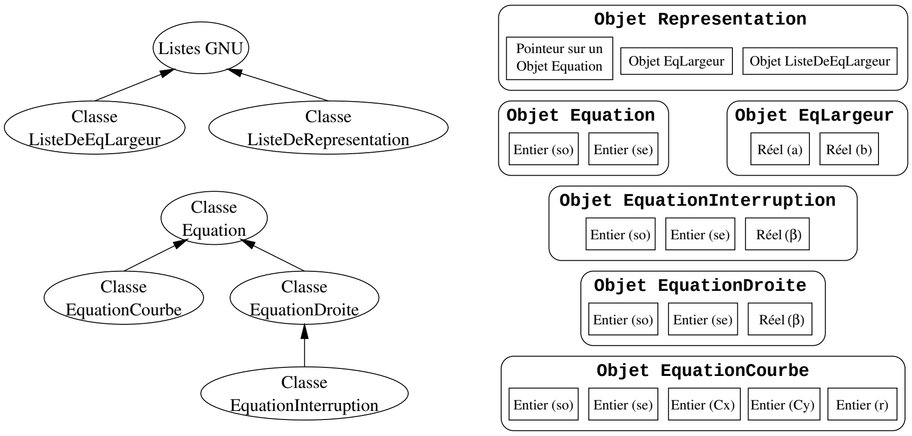


FIG. 4.32 - *Représentation des objets utilisés pour définir les équations*

La classe `Equation` nous permet de représenter les directrices (cette classe est une classe abstraite et a 3 "classes filles" correspondant aux différents types de directrice possibles : `EquationDroite, EquationCourbe et EquationInterruption` - la classe `EquationInterruption` héritant non pas directement de `Equation` mais de `EquationDroite` du fait de la similitude des deux types d'équation). Les attributs de ces classes correspondent aux diverses caractéristiques des équations (angle β_0 pour une équation de droite, coordonnées du centre et rayon pour une équation de courbe). La classe `Equation` est une classe n'ayant aucune instance mais ayant comme attribut deux entiers représentant les abscisses curvilignes de début et de fin de morceau de route (ces deux entiers sont les deux seuls éléments communs à chacune des équations). Ces diverses classes disposent toutes des méthodes `AbscisseCurviligne` et `EstInclus`. La méthode `AbscisseCurviligne` prend en paramètre d'un point dont les coordonnées sont données dans le repère absolu et ainsi que l'origine de la route et retourne l'abscisse curviligne à laquelle se trouve le premier point. La méthode `EstInclus` prend en paramètre un point dont les coordonnées sont données dans le repère absolu et ainsi que l'origine de la route et retourne vrai si ce point est compris entre l'abscisse curviligne de début de l'équation et celle de fin (cette méthode fait appel à la méthode `AbscisseCurviligne`).

Pour représenter les équations de largeur nous avons défini une seule classe : `EqLargeur` qui représente aussi bien les largeurs de morceaux de route que les distances entre les lignes se trouvant entre les deux bords du morceau de route et la directrice et qui a comme attribut deux nombres représentant les coefficients a et b de l'équation. Nous utilisons donc la représentation des équations et les arbres d'héritage donnés dans la figure 4.32.

4.5.4 Architecture

Du fait de la séparation du module de catégorisation des éléments fermés du reste du programme, nous obtenons un système composé d'un programme principal (écrit en `C++`) qui effectue les traitements bas niveau puis l'extraction des routes tout en lançant en parallèle la catégorisation des éléments fermés (cf. figure 4.33). Une fois ces traitements effectués, il transcrit ces résultats en un formalisme compréhensible par le sous-système "construction de la scène".

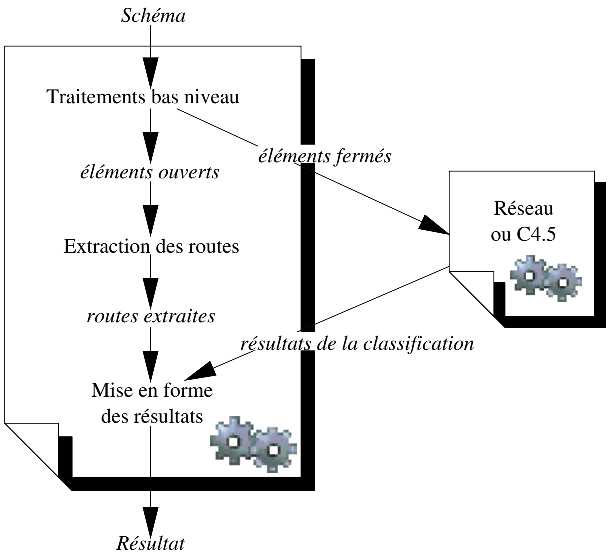


FIG. 4.33 - *Implémentation de l'analyseur de schémas*

Chapitre 5

Résultats obtenus

L'analyseur de schémas que nous venons de présenter avait pour but, non pas de résoudre tous les problèmes liés à l'analyse des schémas de constats d'accidents, mais de déterminer la faisabilité d'un tel analyseur.

Cet analyseur a été testé sur l'ensemble du corpus de schémas dont nous disposions (corpus 1 : 6 schémas dessinés par nos soins ; corpus 2 : 7 schémas réels ; corpus 3 : 64 schémas récoltés lors d'une expérience réalisée au GREYC). Les bons résultats obtenus lors de ce test (cf. figures 5.1, 5.2 et 5.4) tant au niveau de la reconnaissance des routes droites (80% d'entre elles sont correctement reconnues) qu'au niveau des véhicules (88% d'entre eux sont bien classés - cf. table 4.1, page 60) nous permettent d'affirmer que les objectifs visés par cette première réalisation ont été atteints. Nous pensons donc que le système tel qu'il est actuellement apporte une validation des idées générales et prouve la faisabilité d'un tel analyseur.

Le système tel qu'il est actuellement s'avère très positif dans le traitement des routes droites et des véhicules mais ne suffit pas pour traiter tous les problèmes liés à l'analyse des schémas de constats d'accidents. En effet, comme nous l'avons déjà signalé dans la section 4.3, la classification des véhicules et des autres éléments fermés pose encore des problèmes (cf. table 4.1, page 60) et, à notre avis, peut être encore améliorée. D'autre part, il existe encore 20% des routes droites qui sont mal reconnues. Si dans certains cas ces erreurs sont dues à des routes dont les bords forment un angle trop important pour pouvoir être considérés comme parallèles ou sont très mal dessinés (cf. figure 5.5), dans d'autres cas (cf. figure 5.6), la présence de segments qui font normalement partie d'éléments comme des flèches ou des caractères et qui n'ont pas été reconnus par l'analyseur entraîne la création de routes (ou de lignes médianes sur des routes). Cette dernière erreur, liée à la non reconnaissance de certains éléments de l'image, est normale du fait des caractéristiques de l'analyseur développé et de ses limitations (il ne dispose d'aucun mécanisme de reconnaissance des flèches et/ou des caractères) mais devra être corrigée lors de la réalisation d'un analyseur grandeur nature du fait de l'importance de ces éléments dans le schéma. En effet, ces différents éléments apportent soit des informations pseudo-dynamiques importantes sur le déroulement de la scène (flèches de direction), soit des informations sur les antagonistes (indication du nom des véhicules), soit des informations sur la signalisation et doivent être pris en compte pour améliorer l'interprétation du schéma. Les objectifs de ce système étant de prouver la faisabilité d'un analyseur de schémas de constats d'accidents et la recherche de caractères ou de flèches dans une image ayant été abordée par d'autres équipes ([CNED, 1994] et [CNED, 1996]), nous n'avons pas jugé nécessaire de développer de tels mécanismes. Cependant, une version définitive de l'analyseur de schémas, devra, à notre avis, les intégrer.

Un des problèmes importants restant à résoudre pour améliorer cette maquette d'analyseur concerne la reconnaissance des courbes. En effet, seulement environ 50% des virages sont correctement reconnus. Ceci est dû à la méthode assez simpliste que nous avons employée pour effectuer la reconnaissance des courbes (approximation par un cercle). Une méthode un peu plus élaborée (utilisant, par exemple les courbes de Bézier) donnerait sûrement de meilleurs résultats mais, comme cette implémentation avait pour seul but de déterminer la faisabilité d'un tel analyseur, nous n'avons pas voulu pousser plus loin dans la reconnaissance des courbes. Un système réalisé dans un but d'utilisation réelle et non plus de maquette devra cependant tenir compte de ce problème.

Il existe aussi un problème au niveau des intersections et plus spécialement au niveau des ronds-points. En effet, les intersections ne sont bien reconnues que dans 50% des cas et ce pour plusieurs raisons : d'une part un écart trop important entre deux segments entraîne l'absence d'une relation de connexité et donc ne permet pas de détecter un point de jonction, d'autre part, une mauvaise reconnaissance d'une route entraîne une mauvaise reconnaissance de l'intersection.

En conclusion, nous pouvons dire que ce système, bien que limité dans ses objectifs et ses résultats, nous a permis de valider les idées générales que nous avions sur la reconnaissance de tels schémas et est de ce point de vue une réussite. Cependant, il est entièrement à reprendre sur certains points comme, entre autres, la reconnaissance des courbes et celle des véhicules. L'implémentation de cette maquette nous a aussi permis de nous apercevoir des problèmes liés au type de schémas que nous étudions comme l'importance d'intégrer des mécanismes de reconnaissance de caractères et de reconnaissances des flèches mais donne une bonne base pour la réalisation d'un analyseur moins limité et plus efficace.

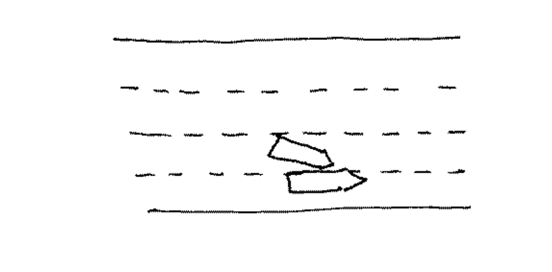
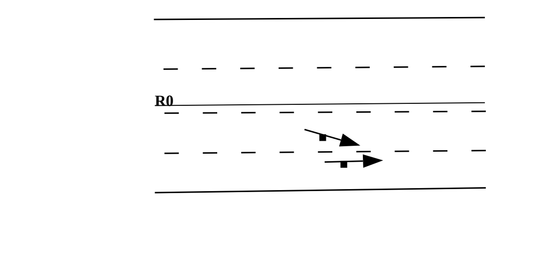
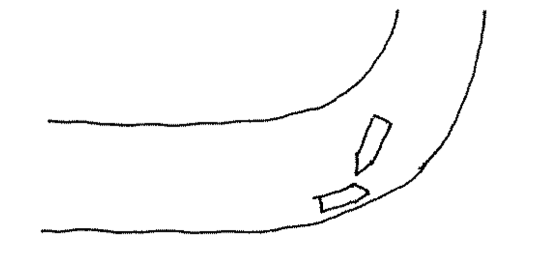
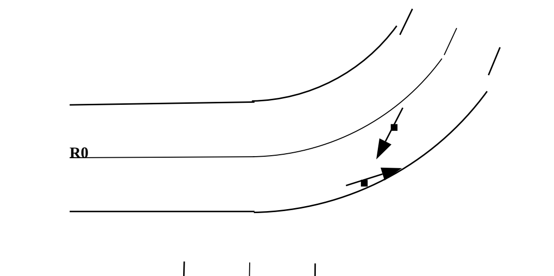
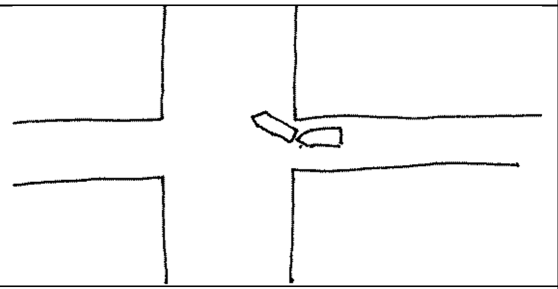
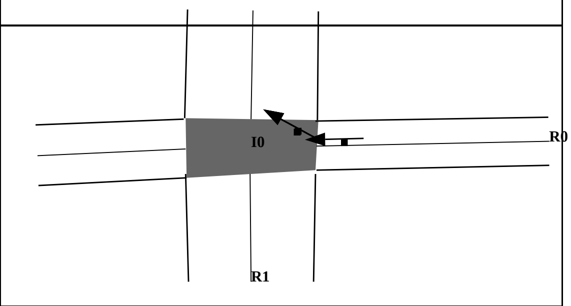
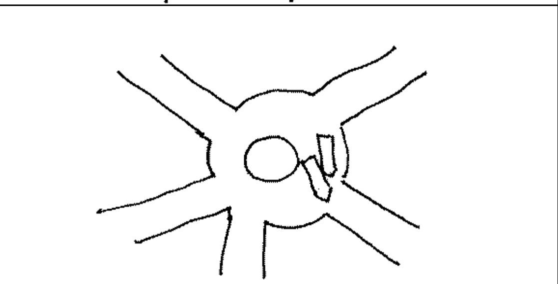
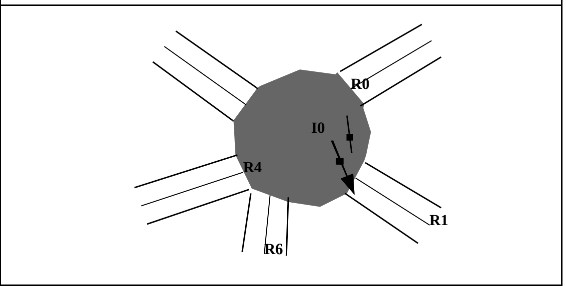
	Image résultant de la numérisation	Résultat de l'analyse
1		
2		
3		
4		

FIG. 5.1 - *Extrait des résultats sur le corpus 1*

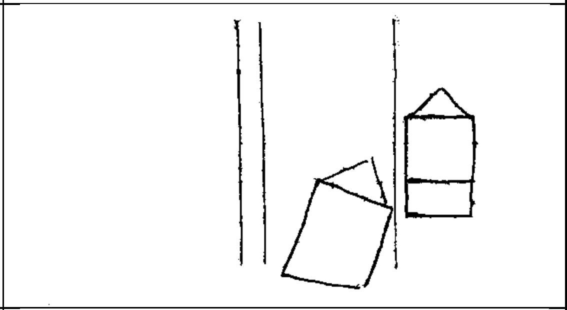
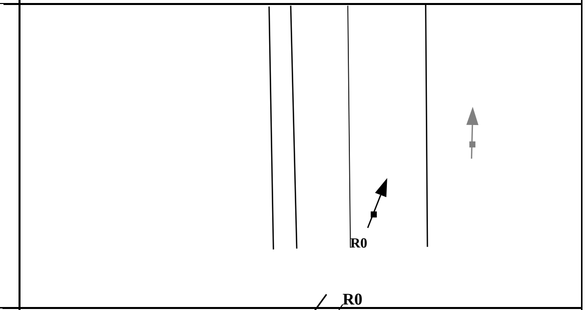
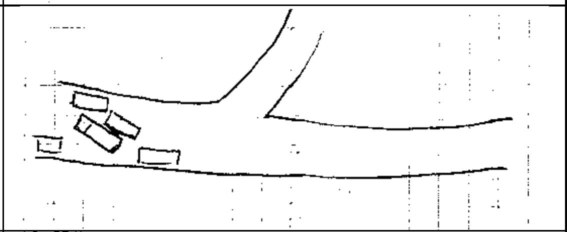
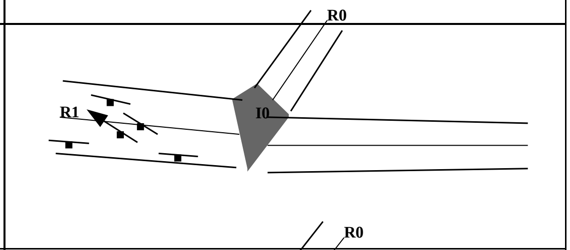
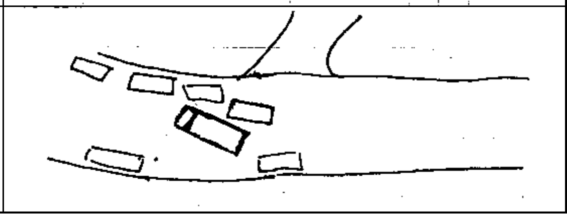
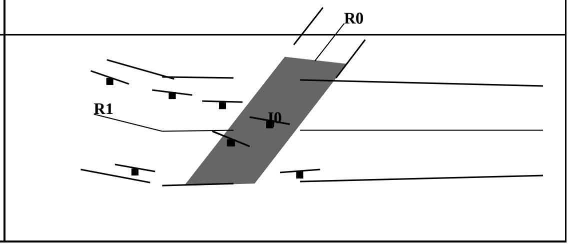
	Image résultant de la numérisation	Résultat de l'analyse
1		
2		
3		

FIG. 5.2 - *Extrait des résultats sur le corpus 2*

Dans le résultat de la ligne 2 de la figure 5.2 l'intersection déterminée est "pointue" en bas en raison du mode de calcul des intersections et des vis-à-vis et qui est visualisé sur la figure 5.3 (les flèches indiquent les relations de vis-à-vis).

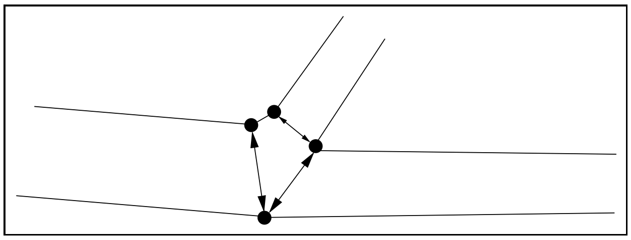


FIG. 5.3 - *Calcul d'une intersection (les flèches indiquent les relations de vis-à-vis)*

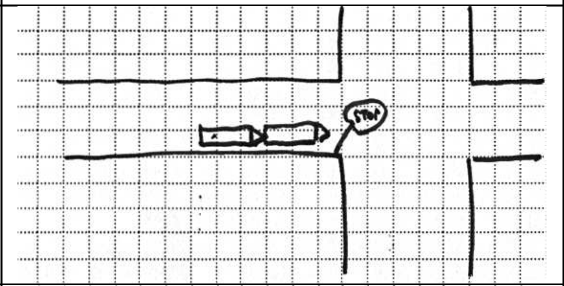
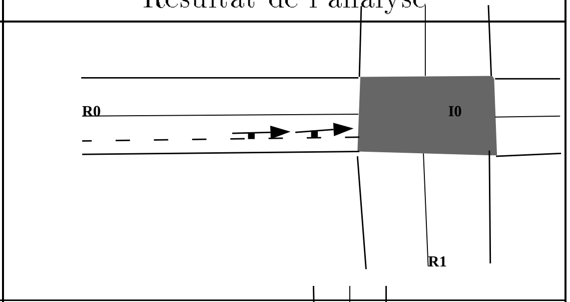
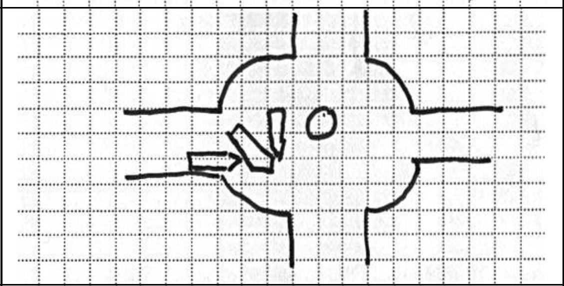
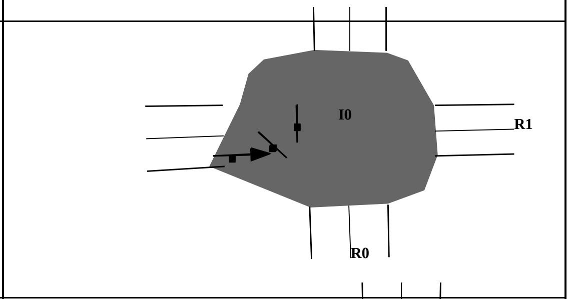
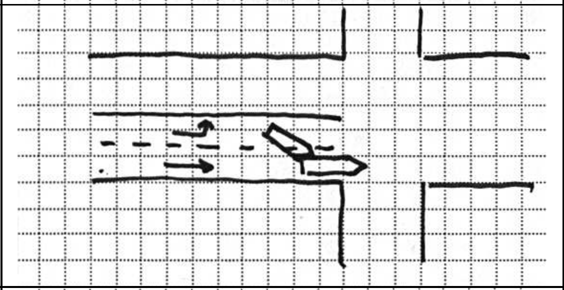
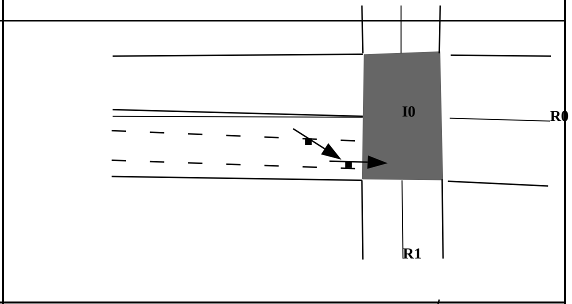
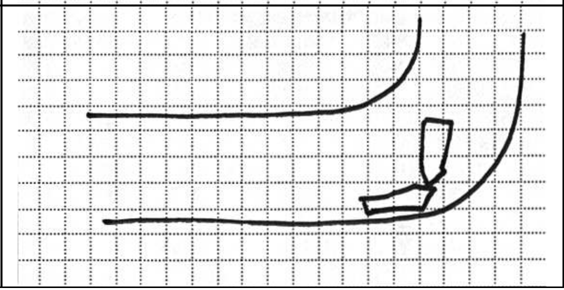
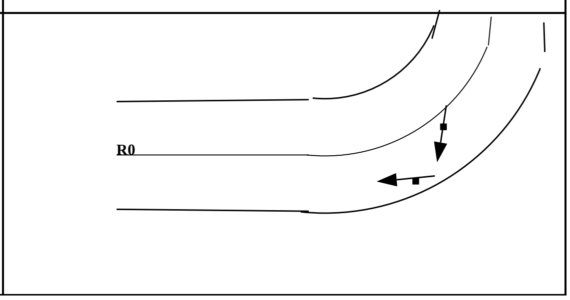
	Image résultant de la numérisation	Résultat de l'analyse
1		
2		
3		
4		

FIG. 5.4 - *Extrait des résultats sur le corpus 3*

Sur le résultat de le ligne 4 de la figure 5.4, un des véhicules est mal reconnu : l'orientation donnée est incorrecte (en sens inverse).

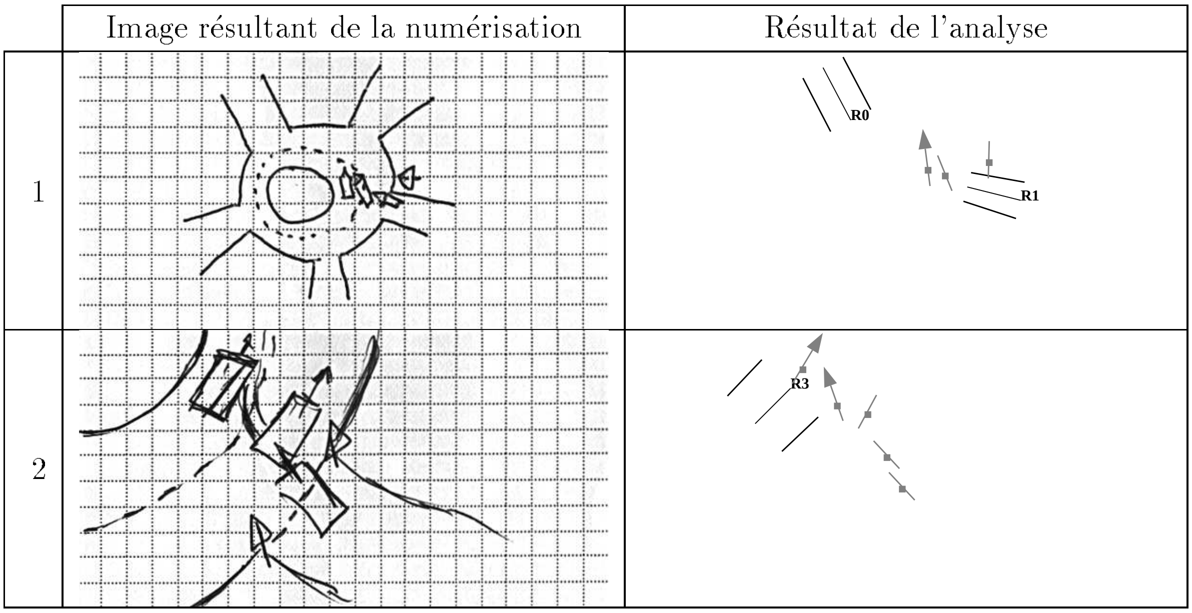


FIG. 5.5 - *Routes mal reconnues*

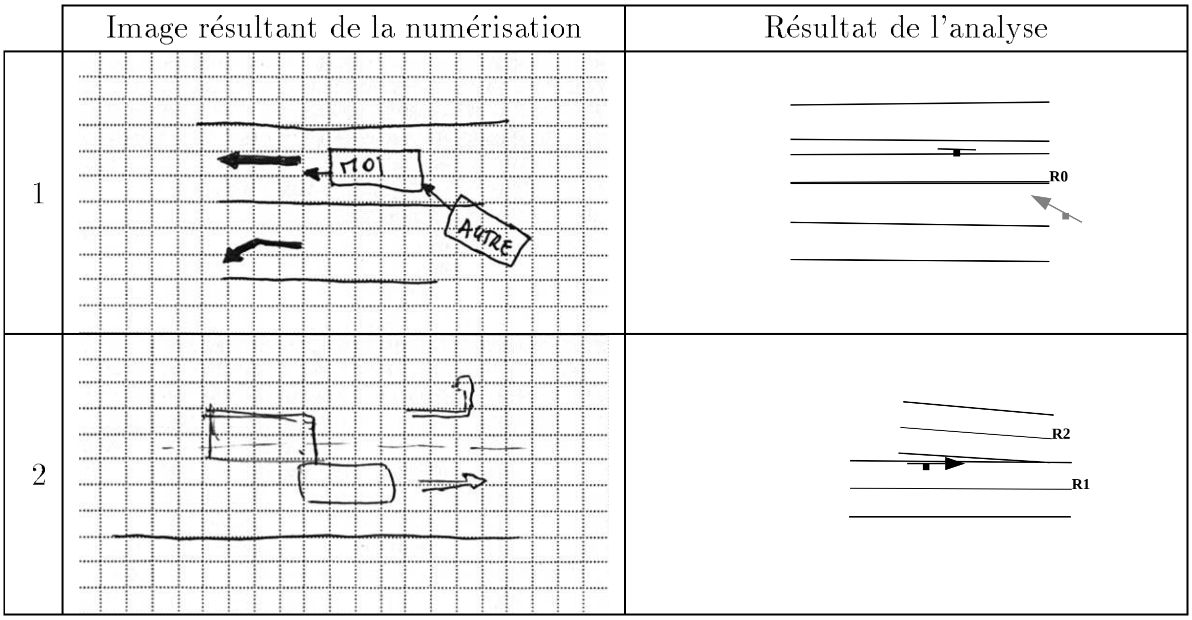


FIG. 5.6 - *Routes où des parties ont été inventées*

Troisième partie

L'analyseur linguistique

L'analyseur linguistique que nous avons mis en place a pour particularité essentielle de limiter son analyse à l'extraction des contraintes sémantiques imposées par le texte et ce, en ne disposant que de connaissances linguistiques.

Dans cette partie, nous donnerons tout d'abord une idée de ce que doit être la sortie de notre analyseur linguistique en indiquant, parmi les éléments de sens présents dans un texte, ceux que nous considérons comme relevant de connaissances linguistiques (et donc appartenant au résultat rendu par cet analyseur) et ceux relevant de connaissances extra-linguistiques (et qui seront gérés et utilisés par le sous-système de construction de la scène). Nous expliquerons ensuite les principes de la coopération mise en place entre l'analyseur linguistique et le sous-système de construction de la scène pour nous permettre d'effectuer tout de même une analyse du texte "en contexte".

La suite de cette partie sera consacrée plus spécialement à l'architecture de l'analyseur linguistique. Nous présenterons tout d'abord un certain nombre d'architectures modulaires couramment utilisées en traitement automatique du langage ainsi que les choix que nous avons effectués pour l'élaboration de l'architecture de cet analyseur et pour son implémentation. Dans la suite de cette partie nous présenterons les divers modules de l'analyseur linguistique. La réalisation de cet analyseur étant un travail d'équipe, nous ne présenterons en détail que les deux modules de l'analyseur que nous avons conçus et réalisés. Les autres modules ne seront qu'abordés et référencés. Lors de ces présentations, le fonctionnement et les résultats obtenus par ces deux modules seront précisés par l'intermédiaire de l'analyse détaillée d'une phrase.

Nous conclurons cette partie en indiquant l'état actuel de réalisation ainsi que les résultats obtenus et leur utilisation par le sous-système chargé d'effectuer la construction de la scène.

Chapitre 6

Particularités de l'analyseur linguistique

Notre analyseur linguistique a pour but de déterminer l'ensemble des contraintes sémantiques imposées par le texte et de les transmettre au sous-système de construction de la scène. Ces contraintes, regroupées autour des *procès*, peuvent porter non seulement sur les objets impliqués (véhicules, routes, conducteurs, etc.), les lieux parcourus (le bord droit de la route, le virage, etc.) mais aussi sur les mouvements effectués (rouler, se déporter, etc.). Le fait d'avoir dans notre corpus, de nombreux verbes de mouvement nous a amenés à attacher une attention plus particulière à la notion de temps et plus particulièrement à celle d'intervalle de temps.

Notre but principal est de réaliser un analyseur linguistique qui, contrairement à la plupart des systèmes, ne soit pas lié au problème qui nous préoccupe. Un des nos objectifs est donc de mettre en place un analyseur linguistique qui ne soit pas complètement ad hoc. Ce projet ambitieux est réalisable grâce à l'architecture globale que nous avons présentée à la section 2.2, page 24 : c'est le sous-système chargé de la construction de la scène qui détient les connaissances sur le monde de la route et qui lui est ad hoc.

Pour pouvoir réaliser un tel système, il nous faut tout d'abord distinguer parmi les éléments de sens présents dans un texte, ceux qui proviennent du texte lui-même et ceux qui relèvent de nos connaissances encyclopédiques. Les éléments de sens relevant de la linguistique (ou "traits linguistiques") seront intégrés au niveau du lexique et seront donc utilisés par l'analyseur linguistique contrairement aux autres éléments de sens. Nous allons maintenant, à partir de quelques exemples, préciser les caractéristiques que nous considérons comme relevant du système linguistique et celles que nous considérons comme dépendant de nos connaissances sur le domaine. Ces divers exemples sont extraits de [Enjalbert & Victorri, 1994], article présentant une réflexion d'ensemble de notre équipe de recherche sur la distinction entre caractéristiques linguistiques et caractéristiques extra-linguistiques.

6.1 Caractéristiques linguistiques et extra-linguistiques

Dans la phrase *Le camion roulait à toute allure*, les données linguistiques ne permettent pas d'estimer, à elles seules, la vitesse du camion. En effet, pour pouvoir estimer cette vitesse, il faut disposer de connaissances sur les camions et sur le code de la route (connaissances encyclopédiques) ainsi que de connaissances sur la situation (cela se déroule en agglomération, sur une autoroute, de jour, de nuit ?). Cependant, même si les données linguistiques liées à l'expression *à toute allure* ne permettent pas de préciser la vitesse du camion, elles apportent un certain nombre de renseignements sur celle-ci et le sens linguistique de cette expression peut être donné de la manière suivante : "à une vitesse qui, du point de vue de l'énonciateur, est beaucoup plus grande que celle à laquelle on pouvait s'attendre dans des conditions similaires, et proche du maximum des possibilités physiques de l'entité pour ces conditions de mouvement". C'est ce sens, valable quelle que soit l'entité mise en cause (camion, voiture, personne mais aussi ballon), que l'on peut obtenir par des procédés linguistiques de comparaison de type paradigmatique (avec *à vive allure*, *à grande vitesse*, *à faible allure*, etc.) et de type syntagmatique (avec *l'homme courait / marchait à toute allure*, *le vélo / le train roulait à toute allure*, etc.), qui doit constituer la sortie de notre analyseur linguistique.

Étudions quelques exemples mettant en valeur le problème des "entités manquantes". Pour comprendre la phrase *Une voiture est arrivée en face*, il faut représenter un conducteur et un véhicule alors qu'à première vue il n'y a de marque linguistique que pour un véhicule. Cependant, il n'est pas difficile d'imaginer l'emploi de la même phrase dans des contextes où le conducteur n'existerait pas (freins desserrés, voiture téléguidée, ...). Le conducteur du véhicule ne sera donc pas créé par l'analyseur linguistique mais inféré par le système de modélisation (à l'aide, par exemple, d'une règle par défaut : en l'absence d'indications contraires, les véhicules ont des conducteurs dans le cadre des témoignages utilisés, ou déduit des conditions d'énonciation ou des phrases lues précédemment).

Dans certains cas, pour déterminer l'existence d'une entité non mentionnée dans le texte, il faut lever une ambiguïté de nature linguistique; c'est le cas, entre autres, du verbe *rouler*. En effet, celui-ci admet une construction syntaxico-sémantique (sujet humain + *rouler* + complément de lieu) à laquelle sont rattachés deux sens différents : "se déplacer en tournant sur soi-même" ou "se déplacer à l'aide d'un véhicule à roues"; la dernière de ces significations sous-entendant la présence d'un véhicule. Pour pouvoir déterminer si le véhicule est une entité à représenter, il faut donc être capable de déterminer le sens de *rouler* à prendre en considération.

Cependant, pour pouvoir déterminer le sens probable du verbe *rouler*, le contexte linguistique ne suffit pas mais il faut disposer du contexte lié à la nature du document à analyser. En effet, dans un texte commençant par *J'ai roulé tout droit*, le verbe *rouler* n'aura pas le même sens s'il s'agit d'un texte portant sur la gymnastique ou sur un accident de la route et le choix entre les différents sens ne peut s'effectuer sur la base de données purement linguistiques. Cependant, on peut tirer, de la forme linguistique utilisée, l'existence de ces deux sens de *rouler* dont un implique l'existence d'un véhicule à roues. La sortie de l'analyseur linguistique sera donc composée des contraintes sémantiques correspondant aux deux sens possibles pour *rouler* et le choix entre ces deux possibilités devra être réalisé par le sous-système de construction de la scène qui dispose, pour pouvoir l'effectuer, du contexte d'énonciation[36].

Enfin, on peut aussi citer le cas des relations temporelles mais la complexité de ces relations nous a amenés à reporter les explications relatives à ce problème à la section 7.2 page 107.

A partir des exemples ci-dessus, nous avons mis au point une méthodologie permettant de distinguer les éléments de sens qui relèvent du système linguistique de ceux qui dépendent de nos connaissances sur le domaine. Celle-ci repose sur l'application de deux critères essentiels :

1. Existe-t-il d'autres formes linguistiques que l'on peut opposer à la forme étudiée sur la caractéristique en question ? Ainsi, le fait que *rouler* (troisième exemple),

36. Nous verrons, ultérieurement (section 6.2, page 91), que le "modèle des attentes du lecteur" nous permet d'éviter l'envoi des deux solutions.

dans la construction qui nous intéresse, s'oppose à *marcher/courir/voler* ... permet d'intégrer le trait "véhicule à roues" dans sa définition linguistique, ce qui n'est pas le cas pour les expressions comme *traverser*, *tourner*, *arriver* que l'on peut trouver dans des emplois similaires.

2. Existe-t-il d'autres situations où la même forme linguistique peut être employée alors que la caractéristique en question n'est plus présente? C'est ce critère qui nous force, par exemple, à renoncer à intégrer le trait "conducteur" à une voiture en déplacement.

Cette méthodologie nous assure une relative indépendance de la sémantique linguistique par rapport au domaine et aux objectifs du système de compréhension. De plus, cela permet une grande liberté dans la construction du système de modélisation, en fonction du problème précis que l'on veut résoudre. Ainsi, si l'on décide de négliger, dans un premier temps, certains aspects des textes de témoignage d'accidents de la route (les aspects argumentatifs, par exemple), on peut le faire sans aucune incidence sur la conception de l'analyseur : les données linguistiques portant sur ces aspects seront tout simplement ignorées par le système de modélisation. De même, toutes sortes de simplifications (sous forme de valeurs par défaut) pourront faciliter le traitement à ce niveau : en somme, c'est le système de modélisation qui est *ad hoc*, ce qui est tout à fait normal puisqu'il ne s'agit pas d'un modèle général du monde, mais d'un modèle bien particulier adapté à la réalisation d'une tâche de compréhension précise et limitée.

Bien sûr, ce découpage présente aussi des inconvénients. D'abord, il n'est pas toujours aussi simple de séparer définitions lexicales et connaissances encyclopédiques et ce, même si on s'appuie sur les deux critères exposés ci-dessus. En fait, il existe de nombreux cas intermédiaires pour lesquels la décision comporte une part d'arbitraire. Ensuite, la levée d'un certain nombre d'ambiguïtés purement linguistiques réclame la prise en compte de la situation. C'est ainsi que le choix entre les deux sens de *rouler* que nous avons relevés dans l'exemple présenté ci-dessus ("rouler sur soi-même" et "se déplacer dans un véhicule à roues") ne peut s'effectuer qu'en faisant appel à des informations contextuelles : "rouler sur soi-même" n'est pas complètement impossible dans le domaine des accidents de la route, mais réclame un contexte bien précis (passager éjecté d'une voiture, piéton renversé ...). C'est pour permettre à l'analyseur linguistique de lever ce genre d'ambiguïtés que nous avons mis en place une coopération entre analyseur linguistique et sous-système de construction de la scène.

6.2 Coopération avec le sous-système de construction de la scène

Comme nous venons de le voir, la levée d'un certain nombre d'ambiguïtés purement linguistiques nécessite des connaissances contextuelles (connaissances portant sur le sujet du verbe, par exemple). Dans le cadre de notre architecture, s'offraient à nous plusieurs possibilités pour résoudre ce problème : ne pas faire de choix au niveau de l'analyseur linguistique et traiter en parallèle les différentes possibilités ou fournir à l'analyseur linguistique les informations contextuelles lui permettant d'effectuer un choix. La première possibilité, même si elle présente, au premier abord, l'avantage de la simplicité de fonctionnement et de mise en œuvre, entraîne une lourdeur au niveau des traitements du sous-système de construction de la scène et augmente ainsi la complexité des traitements effectués par celui-ci.

Pour mettre en place la seconde possibilité, nous nous trouvions confrontés au problème de l'intégration des informations contextuelles au niveau de l'analyseur linguistique. En effet, du fait de l'architecture choisie au niveau de l'ensemble du projet, nous ne pouvions pas ajouter ces connaissances à celles détenues par l'analyseur linguistique. Nous avons donc mis en place un système de coopération entre l'analyseur linguistique et le sous-système de construction de la scène (sous-système disposant de cet ensemble de connaissances). Cette coopération prend plusieurs formes : d'une part une coopération "statique" par l'intermédiaire du lexique (coopération "statique" car l'ensemble des données fournies par le sous-système de construction de la scène ne varie pas au cours de l'analyse), d'autre part, une coopération "dynamique" par l'intermédiaire de ce que nous avons appelé le *modèle des attentes du lecteur* (coopération "dynamique" car, comme nous le verrons par le suite, ce modèle varie au cours de l'analyse - [Poirier & Pied, 1996a] et [Poirier & Pied, 1996b]). Ceci nous donne une architecture comme celle décrite dans la figure 2.1 page 27 et dont la partie ayant trait à l'interprétation du texte est rappelée dans la figure 6.1.

Nous verrons à la section 9.1 que le lexique contient non seulement des données purement linguistiques mais aussi des informations liées au modèle comme la catégorie sémantique des entrées lexicales ou le type des entités à créer.

Le modèle des attentes du lecteur, quant à lui, contient l'ensemble des entités et relations que l'on s'attend à trouver dans le texte en fonction, d'une part de la

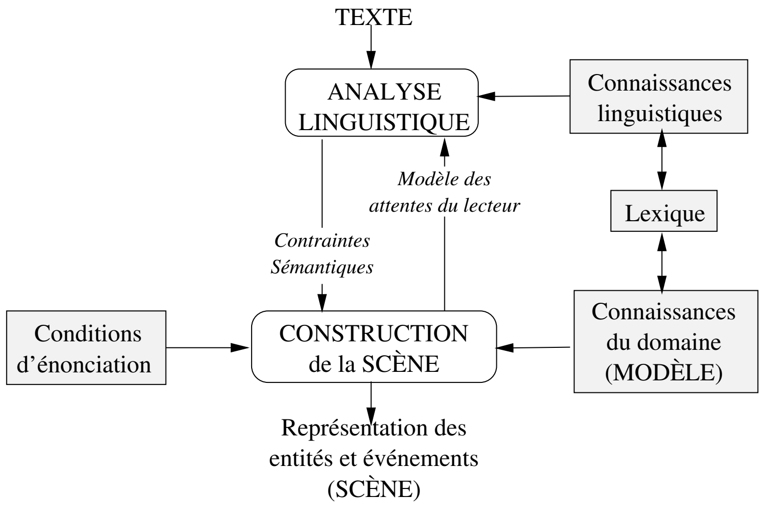


FIG. 6.1 - *Coopération entre analyseur linguistique et sous-système de construction de la scène*

nature du document en lui-même, d'autre part de la lecture des phrases précédentes [Dupont, 1996]. Il regroupe donc à la fois le contexte d'énonciation (les éléments issus des conditions d'énonciation) et le contexte linguistique (les éléments issus de la lecture du texte précédant la phrase en cours d'analyse). Ce modèle est initialisé par le sous-système de construction de la scène à partir des conditions d'énonciation et évolue par ajout des entités et relations rencontrées au cours de la lecture du texte ou déduites de la présence d'autres entités. L'évolution de ce modèle comprend donc l'ajout d'entités liées à celles rencontrées. Dans le cadre des documents qui nous préoccupent (les constats d'accidents), le modèle des attentes du lecteur est initialisé avec, entre autres, deux véhicules, deux personnes, un choc ainsi que les relations permettant de modéliser le fait que les personnes sont les conducteurs respectifs des véhicules, que l'énonciateur est un des conducteurs et que les événements se situent dans le passé.

C'est par l'intermédiaire de ces deux éléments (le lexique et le modèle des attentes du lecteur) que nous pouvons effectuer une analyse "en contexte" sans avoir à intégrer, au niveau de l'analyseur linguistique, de connaissances contextuelles.

De plus, le système de communication mis en place (cf. section 11.3.1) permet, aux deux sous-systèmes de pouvoir communiquer directement entre eux. Cette fonctionnalité est utilisée non seulement pour transmettre les résultats de l'analyse ainsi que le modèle des attentes du lecteur mais aussi, dans certains cas, pour résoudre des problèmes de référence.

La communication directe entre les deux sous-systèmes est utilisée aussi pour résoudre les incohérences rencontrées par le sous-système de modélisation lors de la phase d'interprétation (ceci sera étudié plus en profondeur dans la section 11.2) : elle permet au sous-système de construction de la scène de faire appel à l'analyseur pour revoir certains choix de schémas de sens qui pourraient être la cause de cette incohérence.

Comme nous pouvons le voir, les inconvénients créés par la séparation en plusieurs sous-systèmes sont contournables, sans remise en cause de l'architecture générale : ils forcent au contraire à prendre conscience de difficultés essentielles du traitement de la langue qui sont le plus souvent éludées dans les analyseurs *ad hoc*.

6.3 Les sorties de l'analyseur linguistique

La sortie de l'analyseur linguistique est constituée de l'ensemble des contraintes sémantiques imposées par le texte et regroupées par procès. Le résultat de cette analyse est fourni au sous-système de construction de la scène procès par procès à la fin de l'analyse de chaque phrase. Le fait d'effectuer une transmission des résultats à la fin de chaque phrase permet une meilleure évolution du modèle des attentes du lecteur. En effet, l'analyseur linguistique, ne disposant pas de connaissances sur le monde, ne peut ajouter dans ce modèle les entités liées à la présence d'autres entités. Seul le sous-système de construction de la scène est à même de faire de tels ajouts. La transmission des résultats en fin de phrase permet donc au sous-système de construction de la scène de faire évoluer le modèle des attentes du lecteur en fonction de l'analyse effectuée et de transmettre le nouveau modèle à l'analyseur linguistique qui l'utilisera pour l'analyse de la phrase suivante. De plus, comme nous le verrons dans la section 11.2, cette transmission permet aussi au sous-système de construction de la scène de détecter, le plus rapidement possible, des incohérences entre résultat de l'analyse linguistique et modèle du monde.

Le formalisme utilisé pour la description des contraintes est un formalisme logique lié à celui utilisé par le sous-système de construction de la scène. Pour la représentation des relations définies sur un intervalle de temps précis, nous utilisons une notation modale qui permettra au sous-système de construction de la scène d'effectuer un raisonnement logique sur les intervalles de temps (raisonnement basé sur les travaux de Gérard Becher [Becher, 1996] effectués à partir des intervalles et relations de [Allen, 1983]) de façon à retrouver la chronologie des événements.

Le résultat de l'analyseur linguistique est donc composé à la fois du texte correspondant au procès et des contraintes sémantiques imposées par le texte qui sont ordonnées en 3 grandes parties dont nous fournissons tout de suite la description de façon à donner une vue d'ensemble mais qui seront détaillées par la suite.

- la partie "objets" qui contient les entités mises en cause par les différentes relations utilisées ou créées lors de l'analyse ainsi que les intervalles temporels dont on verra le sens section 7.2, page 107.

- la partie "relations aspectuo-temporelles" qui contient les relations entre les différents intervalles déterminées par le module chargé d'établir les relations aspectuo-temporelles. Ces relations (que nous décrirons plus en détail section 7.2, page 107) sont des relations d'adjacence, d'antériorité ou d'inclusion entre intervalles (notées respectivement $|$, $<$ et $\subset$) et sont définissables par des relations d'ordre entre leurs bornes.

- la partie "relations événementielles" qui contient le reste des relations établies lors de l'analyse (relations spatiales, propriétés des entités, conditions sur les entités, ...).

Un exemple, accompagné de quelques commentaires, devrait permettre de préciser le format que nous avons adopté pour le codage des informations liées aux procès. Il s'agira du texte suivant :

Je roulais à droite sur la chaussée quand un véhicule arrivant en face dans le virage m'a percuté.

Ce texte est découpé en 2 propositions (proposition principale : *Je roulais à droite sur la chaussée*, proposition subordonnée : *quand un véhicule arrivant en face dans le virage m'a percuté*) et contient 3 procès marqués par les verbes *rouler*, *percuter* et *arriver*. Dans les descriptions correspondant à ces trois procès et qui sont données ci-dessous, les parties en italique correspondent à des commentaires et les notations "`[IP`$_i$`] relation`" (notations modales) nous permettent d'exprimer le fait qu'une relation est vraie sur un intervalle de temps donné.

Procès n° 1 :

TEXTE : ``je roulais à droite sur la chaussée (quand)''

OBJETS :

$\mathtt{S}_0$: personne	*Entité associée à "je"*
$\mathtt{V}_0$: véhicule	*Entité implicite liée au schéma de sens utilisé pour "rouler"*
$\mathtt{E}_0$: déplacement	*Entité correspondant à l'événement associé à "rouler"*
$\mathtt{R}_0$: route	*Entité associée à "chaussée"*
$\mathtt{Rep}_0$: repère	*Repère lié au sens de "à droite"*
$\mathtt{IE}_0$: intervalle	*Intervalle d'énonciation (moment où a été rédigé le texte)*[37]
$\mathtt{IP}_1$: intervalle	*Intervalle temporel lié aux bornes du procès (correspond à l'action de "rouler")*[37]
$\mathtt{IM}_1$: intervalle	*Intervalle de monstration lié à "roulais" (correspond à la partie de l'action qui est montrée)*[37]

RELATIONS ASPECTUO-TEMPORELLES :

$\mathtt{IM}_1 < \mathtt{IE}_0$ &	*Le procès est antérieur au moment de l'énonciation*[37]
$\mathtt{IM}_1 \subset \mathtt{IP}_1$	*Le procès n'est pas montré dans sa totalité*[37]

RELATIONS ÉVÉNEMENTIELLES :

[$\mathtt{IP}_1$] véhicule($\mathtt{S}_0$, $\mathtt{V}_0$) &	$\mathtt{S}_0$ *a comme véhicule* $\mathtt{V}_0$.
[$\mathtt{IP}_1$] conducteur($\mathtt{V}_0$, $\mathtt{S}_0$) &	$\mathtt{V}_0$ *a comme conducteur* $\mathtt{S}_0$.
à_roues($\mathtt{V}_0$) &	*Le véhicule* $\mathtt{V}_0$ *utilise normalement ses roues pour se déplacer.*
$\mathtt{E}_0$.acteur = $\mathtt{V}_0$ &	*L'acteur du déplacement est* $\mathtt{V}_0$.
[$\mathtt{IP}_1$] appartient_dim2($\mathtt{S}_0$, $\mathtt{R}_0$) &	$\mathtt{S}_0$ *se trouve sur la route* $\mathtt{R}_0$.
[$\mathtt{IP}_1$] inf(position_y($\mathtt{S}_0$, $\mathtt{Rep}_0$), "0")	$\mathtt{S}_0$ *est situé à droite du repère* $\mathtt{Rep}_0$.

Dans le procès n° 1, l'analyse linguistique stricte laisse inspécifié le référentiel $\mathtt{Rep}_0$ auquel réfère l'expression *à droite*, et qui doit donc être actualisé en fonction d'autres informations par le sous-système chargé d'effectuer la construction de la scène (rouler à droite tout en étant sur la chaussée correspond à rouler sur la file la plus à droite de celle-ci, c'est-à-dire, sur la file la plus à droite du repère lié à la route).

37. Ces divers intervalles et relations seront plus amplement détaillés dans la section 7.2, page 107

Procès n° 2 :

TEXTE : `''quand un véhicule m ' a percuté''`

OBJETS :

$\texttt{V}_1$: `véhicule`	*Entité associée à "véhicule"*
$\texttt{S}_0$: `personne`	*Entité associée à "m' "*
$\texttt{E}_1$: `choc`	*Entité correspondant à l'événement associé à "percuter"*
$\texttt{IE}_0$: `intervalle`	*Intervalle d'énonciation*[37]
$\texttt{IP}_2$: `intervalle`	*Intervalle temporel lié aux bornes du procès*[37]
$\texttt{IM}_2$: `intervalle`	*Intervalle de monstration lié à "a percuté"*[37]

RELATIONS ASPECTUO-TEMPORELLES :

$\texttt{IM}_1 \mid \texttt{IM}_2$ &	*Les procès "rouler" et "percuter" sont adjacents*[37]
$\texttt{IP}_2 = \texttt{IM}_2$ &	*Le procès est montré dans sa totalité*[37]
$\texttt{IM}_2 < \texttt{IE}_0$	*Le procès est antérieur au moment de l'énonciation*[37]

RELATIONS ÉVÉNEMENTIELLES :

$\texttt{E}_1$`.acteur =` $\texttt{V}_1$ &	*L'acteur du choc est* $\texttt{V}_1$.
$\texttt{E}_1$`.objet =` $\texttt{S}_0$	*L'objet du choc est* $\texttt{S}_0$.

Dans le procès n° 2, on remarque que, si lexicalement *percuter* nécessitait (dans cette acception) une entité de type "objet mobile", la valeur lexicale du sujet (*un véhicule*) a permis de spécifier davantage le type de cette entité (`véhicule`).

Procès n° 3 :

TEXTE : `''(un véhicule) arrivant en face dans le virage''`

OBJETS :

$\texttt{V}_1$: `véhicule`	*Entité associée à "véhicule"*
$\texttt{E}_2$: `déplacement`	*Entité correspondant à l'événement associé à "arriver"*
$\texttt{SR}_0$: `virage`	*Entité associée à "virage"*
$\texttt{R}_0$: `route,`	*Entité représentant la route sur laquelle se trouve le virage*
$\texttt{Rep}_1$: `repère`	*Repère lié au sens de "en face"*
$\texttt{IE}_0$: `intervalle`	*Intervalle d'énonciation*[37]
$\texttt{IP}_3$: `intervalle`	*Intervalle temporel lié aux bornes du procès*[37]
$\texttt{IM}_3$: `intervalle`	*Intervalle de monstration lié à "roulant"*[37]

RELATIONS ASPECTUO-TEMPORELLES :

$\texttt{IM}_3 \subset \texttt{IP}_3$ &	*Le procès n'est pas montré dans sa totalité*[37]
$\texttt{IM}_3 = \texttt{IM}_2$	*La partie montrée du procès "arriver" correspond à celle du procès "percuter"*[37]

suite page suivante ...

suite de la page précédente	
RELATIONS ÉVÉNEMENTIELLES :	
E_2.acteur = V_1 &	*l'acteur du déplacement est* V_1
E_2.borneinfip = "horschamp" &	*Au moment de la borne inférieure de l'intervalle* IP_3 *associé au déplacement, l'acteur de* E_2 *est hors d'une zone saillante de la scène*
E_2.bornesupip = "danschamp" &	*Au moment de la borne supérieure de l'intervalle* IP_3 *associé au déplacement, l'acteur de* E_2 *est dans une zone saillante de la scène*
[IP_3] sup(position_x(V_1, Rep_1), "0") &	*Le véhicule* V_1 *est "devant" par rapport au repère* Rep_1
[IP_3] position_y(V_1, Rep_1) = "petite" &	*Le véhicule* V_1 *est faiblement éloigné de l'axe des x du repère* Rep_1
inclus(SR_0, R_0)	*Le virage* SR_0 *est une partie de la route* R_0

Dans le procès n°3, Rep_1, comme Rep_0, est un repère laissé inspécifié par l'analyse linguistique. Ce repère est requis pour l'interprétation de "en face" et doit être déterminé lors de la construction de la scène en fonction de la saillance des différents repères.

Comme nous considérons que *arriver* consiste à entrer dans une zone saillante de la scène, V *arrive* est ici interprété comme : "V se déplace, le début du déplacement est hors de la zone saillante de la scène et la fin de celui-ci est dans cette zone". Cette interprétation est donnée, dans la description du procès, par les entités et relations suivantes : E_2 : déplacement, E_2.acteur = V_1, E_2.borneinfip = "horschamp" et E_2.bornesupip = "danschamp". De plus, nous considérons que pour *arriver en face* la zone saillante se trouve en face d'un certain repère et cette interprétation est spécifiée par les relations [IP_3] sup(position_x(V_1, Rep_1), "0") et [IP_3] position_y(V_1, Rep_1) = "petite". Nous n'irons pas plus loin dans la description des relations spatiales, ceci relevant, dans le cadre de notre projet, du travail de Y. Mathet [Mathet, 1995].

La dernière relation (inclus(SR_0, R_0)) suppose que l'analyseur linguistique ait pu résoudre l'anaphore associative reliant la route "support" de SR_0 et la route R_0 introduite par le premier procès.

Comme nous l'avons annoncé section 2.3, page 30, la description des contraintes sémantiques imposées par le texte que nous effectuons peut être rapprochée des DRS de la DRT. En effet, elle comprend, au même titre que les DRS, les objets (référents) extraits du texte ainsi que les relations (conditions) qu'ils entretiennent. La grande différence qui existe entre nos contraintes sémantiques et les DRS concerne la hiérarchie de la structure. En effet, alors que le résultat fourni dans le cadre de la DRT est une DRS contenant le plus souvent des sous-DRS reliées par des conditions, le résultat de notre analyseur est un ensemble non hiérarchisé de contraintes. Bien entendu nous avons un certain nombre de relations entre procès représentées, en particulier, par des relations entre intervalles temporels mais ces relations ne conduisent, en aucun cas, à intégrer un procès dans un autre contrairement aux relations entre DRS (une représentation de procès ne contient jamais une autre représentation de procès).

Une autre différence existe entre notre système et un système basé sur la DRT : il s'agit de la transmission du résultat. En effet, contrairement à la DRT, le résultat fourni par l'analyse linguistique que nous effectuons n'est pas constitué d'une seule structure englobante mais de plusieurs (une par procès détecté) qui sont transmises non pas à la fin de l'analyse du texte mais au fur et à mesure de l'analyse des phrases au sous-système chargé d'effectuer l'intégration dans le modèle du monde. Ce mode de transmission permet, comme nous l'avons vu précédemment, une meilleure évolution du modèle des attentes du lecteur et donc une gestion des anaphores en fonction de la saillance des entités ainsi qu'une meilleure gestion des incohérences (cf. partie IV).

Chapitre 7

Architecture et implémentation

Un certain nombre de systèmes développés en traitement automatique du langage l'ont été à partir d'architectures non-modulaires ([Rady, 1983] et [Seligman, 1985]). Cependant, ces systèmes, même s'ils ont prouvé la validité d'une telle approche, ont montré sa complexité de mise en œuvre : les connaissances nécessaires à la réalisation d'un système de traitement automatique du langage sont, comme l'ont montré [Pierrel & Sabah, 1991], [Sabah, 1990b] et [Sabah, 1989], non seulement importantes mais aussi de niveaux très différents (connaissances morphologiques, syntaxiques, sémantiques ...). De plus, du fait des nombreuses interactions existant entre les diverses parties qui le composent, l'évolution d'un tel système est difficile : les modifications apportées à une procédure A peuvent entraîner des modifications sur une procédure B etc.

Les systèmes à base d'architecture modulaire (systèmes où, à un type de connaissances, on fait correspondre un module particulier effectuant un traitement bien spécifique) semblent quant à eux plus faciles à comprendre et à manipuler [Simon, 1969]. En effet, l'avantage d'une telle architecture réside non seulement dans la répartition des connaissances nécessaires à un tel système, mais aussi dans la facilité de mise au point et de maintenance du système : une telle architecture permet de tester séparément les différentes parties qui la composent (les modules) et une répartition rigoureuse des différentes tâches à effectuer permet de réaliser plus facilement d'éventuelles modifications des modules. Le fait de choisir une architecture modulaire pour développer un système en traitement automatique du langage permet d'avoir un système dont l'évolution de chacune des parties est plus facile : dans de nombreux cas, les modifications étant internes à un module peuvent être effectuées sans avoir de conséquence sur les autres modules.

Nous nous sommes donc basés, pour définir l'architecture de notre système, sur les différentes architectures modulaires existantes. Avant de présenter l'architecture que nous avons choisie, nous présenterons diverses architectures modulaires utilisées en traitement automatique du langage.

7.1 Les architectures modulaires

Il existe un certain nombre d'architectures modulaires classiques ayant fait leurs preuves en traitement automatique du langage. Ces architectures se différencient non seulement par leur contrôle mais aussi par les possibilités qu'elles offrent au niveau de la circulation des données.

Dans une *architecture en série* (cf. figure 7.1) comme celle mise en place dans KEAL (système de dialogue homme/machine fortement hiérarchisé et développé au CNET[38] [Mercier *et al.*, 1977]), MYRTILLE I [Pierrel, 1975] ou ESOPE [Mariani, 1982] (systèmes hiérarchisés de compréhension automatique du discours continu et développés respectivement au CRIN[39] et au LIMSI[40]), le flot de données va toujours dans le même sens (du module N au module N+1) et le rôle d'une source de connaissances est limité à certains modules. Dans une telle architecture, le contrôle est un contrôle séquentiel : les modules sont exécutés les uns à la suite des autres.

Dans une *architecture de type hiérarchique* (cf. figure 7.1) comme celle mise en place dans HWIM [Wolf & Woods, 1980] et MYRTILLE II [Pierrel, 1981] (systèmes de compréhension de la parole continue, caractérisés par la présence d'un superviseur qui contrôle l'ordre d'activation des différentes composantes et développés respectivement par BBN[41] et le CRIN), bien que le contrôle soit un peu réparti (un module peut en contrôler plusieurs qui peuvent eux-mêmes en contrôler d'autres), le flot de données, comme dans une architecture en série, va toujours dans le même sens.

Dans une *architecture hétérarchique*, par contre, le contrôle peut être distribué dans

38. Centre National d'Études des Télécommunications.
39. Centre de Recherches en Informatique de Nancy.
40. Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur.
41. Bolt Beranek and Newman.

tous les modules : tout module peut communiquer avec un autre, et quand deux modules sont en liaison, n'importe lequel peut influer sur l'autre (cf. figure 7.1). L'avantage de cette dernière architecture semble évident puisque les conditions spécifiques au problème à traiter déterminent, à un moment donné, quel est le spécialiste le plus approprié qu'il faut appeler. La flexibilité du fonctionnement est alors très importante, comparée aux deux architectures précédentes, bien que des difficultés spécifiques, liées au contrôle du processus puissent apparaître. En théorie, l'architecture hétérarchique est capable de remédier à certaines insuffisances de l'architecture en série. Cependant, la mise en œuvre d'une telle architecture est délicate. En effet, si les modules comportent un trop grand nombre de points d'interaction, ce type de système devient inefficace et très coûteux du fait de la difficulté à mettre en place le contrôle. Ce type d'architecture, pour ne pas perdre de son efficacité, présuppose donc que les interactions entre les modules restent limitées et bien déterminées.

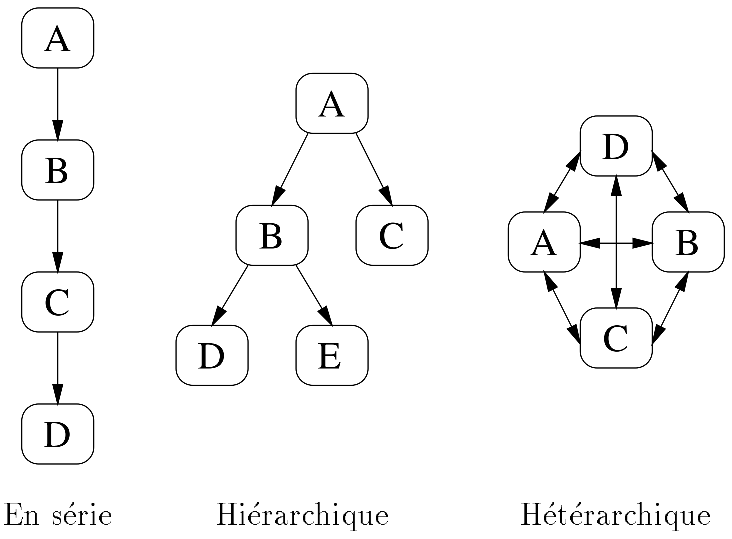


FIG. 7.1 - *Diverses architectures modulaires*

Dans une *architecture de type tableau noir*, les diverses bases de connaissances utilisées sont considérées comme indépendantes et collaborent pour résoudre un problème et ce, sans directement communiquer entre elles. Ce tableau est une structure de données globale où sont stockées les hypothèses émises par les diverses sources de connaissances à partir du problème à résoudre. Chaque source de connaissances cherche dans le tableau s'il existe des hypothèses qui la concernent, si oui, elle y écrit de nouvelles hypothèses à traiter par les autres sources. On peut considérer que cette structure est la généralisation de la simple base de faits que l'on trouve dans les systèmes experts. Ce type d'architecture est utilisé dans le système HEARSAY II [Erman *et al.*, 1980] (système construit dans le cadre du projet ARPA et destiné à comprendre des entrées vocales).

Ces quatre types d'architecture sont en fait les architectures classiques servant de base à la plupart des systèmes de compréhension automatique du langage. Cependant, leurs limites ont conduit au développement d'architectures complexes combinant divers de leurs aspects.

Les systèmes multi-agents ou multi-experts communicants [Haton, 1989] en sont un exemple. Ce type de système généralise la notion de tableau noir en permettant des modes de communication élaborés entre les différents modules et autorise l'utilisation de métaconnaissances pour leur gestion et la construction d'explications. En revanche, un problème important reste posé : celui de la communication entre des experts de domaines différents, que sont les différents modules. Par exemple, dans le système ATOME [Haton, 1989], les divers éléments de contrôle se contentent d'interpréter les résultats des processus spécialisés, mais n'interviennent pas dans leur fonctionnement. Le contrôle est donc intermittent. Une autre possibilité est la gestion des tâches par un agenda. Les tâches à réaliser sont inscrites dans un agenda et sont exécutées dans un ordre qui peut être fixé par l'ordre d'arrivée (c'est le cas du système CENTAUR, [Aikins, 1980]) ou calculé (GUS, [Bobrow *et al.*, 1977]). Ce dernier est issu de KRL [Bobrow & Winograd, 1977] où un régisseur peut réorganiser périodiquement les priorités des tâches de l'agenda. Ici aussi, le contrôle est intermittent puisque le régisseur ne peut intervenir qu'après que chaque processus a terminé le travail qui lui est confié.

Le système multi-experts CARAMEL[42] [Sabah, 1990a] est un système très général permettant de réaliser diverses tâches liées au traitement automatique du langage : réalisation d'interfaces conviviales en langue naturelle, l'aide à la préparation de documents, indexation automatique de documents, résumé automatique de textes .... Les concepteurs de ce système ont étendu la notion de système multi-experts en permettant une gestion des déclenchements et interruptions des processus ainsi qu'un contrôle permanent. Dans ce système, la synchronisation et le déclenchement des experts spécifiques sont gérés par un métaexpert (le superviseur). Ce superviseur utilise les concepts de tableau noir et d'agenda pour déterminer, en fonction de la tâche à effectuer, l'enchaînement des divers processus et gérer les problèmes rencontrés par ces processus. La planification effectuée par ce superviseur permet d'avoir un fonctionnement du système qui peut correspondre à diverses architectures : dans certains cas simples, le déroule-

42. Compréhension Automatique de Récits, Apprentissage et Modélisation des Échanges Langagiers.

ment du programme est équivalent à une architecture en série, en revanche, pour des cas plus complexes, il est équivalent à l'utilisation d'une architecture hiérarchique voire hétérarchique.

Dans le système DIAL (interface vocale homme/machine à l'intention du grand public et dans le cadre d'applications téléphoniques [Roussanaly & Pierrel, 1992]), la coopération entre les modules (qui sont au nombre de 5) est assurée par un contrôle réparti se traduisant par une communication bidirectionnelle entre les composantes.

L'architecture que nous avons mise en place est, comme celles de ces derniers systèmes, une combinaison des architectures classiques décrites précédemment.

L'analyse effectuée par notre analyseur peut être découpée en deux grandes parties : d'une part une analyse morpho-syntagmatique du texte, d'autre part une analyse syntaxico-sémantique effectuée à partir du résultat de l'analyse morpho-syntagmatique. La syntaxe est donc répartie sur les deux parties du système. L'analyse morpho-syntagmatique a pour but de déterminer uniquement les syntagmes de la phrase (sans aller jusqu'à la construction de l'arbre syntaxique). L'analyse syntaxico-sémantique détermine l'arbre syntaxique de la phrase en se basant, entre autres, sur les constructions syntaxiquement admissibles des verbes et en utilisant des données sémantiques. La réalisation de l'analyse syntaxico-sémantique est essentiellement basée sur l'utilisation d'études linguistiques effectuées dans des domaines comme le calcul de la référence [Ariel, 1990], l'étude des relations spatio-temporelles ([Sablayrolles, 1993] et [Vandeloise, 1986]) et celle des relations aspectuo-temporelles [Gosselin, 1995]. L'utilisation de ces différentes études sectorielles nous a conduit à mettre en place 3 modules experts distincts qui, tout en travaillant séparément, doivent cependant pouvoir communiquer directement entre eux. En effet, le traitement de certains problèmes demande une étroite collaboration entre plusieurs de ces experts : le traitement d'un verbe de mouvement nécessite une étroite collaboration entre l'expert chargé de traiter les problèmes liés aux relations aspectuo-temporelles et celui chargé des relations spatio-temporelles. Ces trois modules experts sont utilisés pour étoffer le squelette syntaxique de la phrase et sont donc déclenchés par le module chargé de la construction de ce dernier (le module d'analyse des relations actancielles), module qui est en fait le "contrôleur" de l'analyse syntaxico-sémantique. L'ensemble de ces modules syntaxico-sémantiques n'intervient que sur réception du résultat de l'analyseur morpho-syntagmatique.

L'architecture que nous avons mise en place est une architecture que nous considérons comme "faiblement hétérarchique" (ou architecture hétérarchique contrôlée). En effet, du fait des interactions existant au niveau des 3 experts, celle-ci peut être considérée comme hétérarchique. Cependant, ces interactions étant bien spécifiées et limitées à des problèmes précis ne demandant qu'un seul échange entre les modules, elle est contrôlée et dispose des avantages d'une telle architecture sans en avoir les inconvénients. Le contrôle effectué par le module d'analyse des relations actancielles intervient non seulement au niveau des appels de modules mais aussi au niveau de la gestion des données communes : tous les modules sont susceptibles d'accéder en lecture à ces données, mais il est le seul à avoir le droit de les modifier. Cette architecture, comme le montrent les figures 7.2 et 7.3, est constituée d'un flot principal série entre l'analyseur morpho-syntagmatique et le module d'analyse des relations actancielles, d'un contrôle hiérarchique du module d'analyse des relations actancielles sur les autres modules syntaxico-sémantiques ainsi que d'un flot hétérarchique secondaire au niveau des 3 experts. Cette architecture est à rapprocher de celle de KERNEL [Palmer *et al.*, 1993] (système de compréhension automatique de textes).

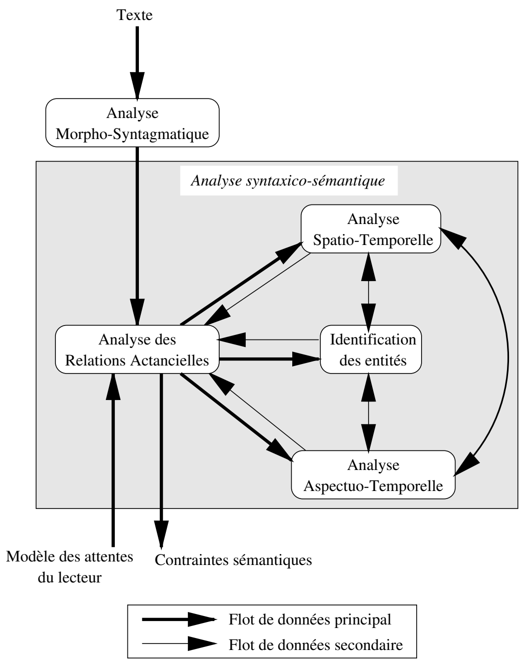


FIG. 7.2 - *Flots de données dans l'analyseur linguistique*

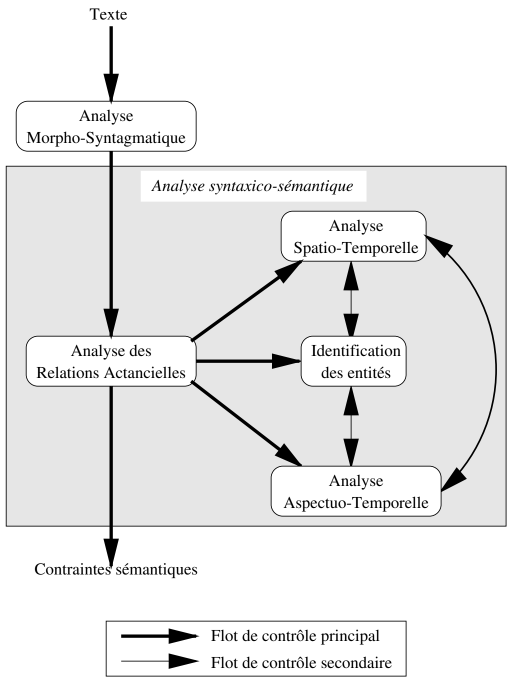


FIG. 7.3 - *Contrôle dans l'analyseur linguistique*

7.2 Architecture mise en place

Nous obtenons alors un système composé de 5 modules principaux où chacun des modules dispose de connaissances spécifiques au problème qu'il est chargé de résoudre. Les modules composant cet analyseur sont les suivants : "Analyse Morpho-Syntagmatique" (AMS) qui effectue un premier découpage du texte en segments, "Analyse des Relations Actancielles" (ARA) qui a pour but de construire la structure actancielle de la phrase et de faire appel aux trois modules experts pour la compléter. Ces modules experts sont : "Identification des Entités" (ID) qui gère les problèmes d'identification et de référence, "Analyse des Relations Aspectuo-Temporelles" (AT) et "Analyse des Relations Spatio-Temporelles" (ST).

L'analyse morpho-syntagmatique

Ce module, qui sera plus amplement détaillé dans le chapitre 8, occupe une place particulière dans notre analyseur : il est chargé d'effectuer un premier découpage du texte en réalisant une analyse utilisant uniquement des connaissances morphologiques et syntaxiques. L'idée de base utilisée est de regrouper les unités morphologiques en "segments maximaux sûrs", c'est-à-dire en groupes d'unités dont les relations syntaxiques sont déterminées de façon univoque par la catégorie et la contiguïté des unités sur la chaîne (idée largement inspirée des travaux de J. Vergne [Vergne, 1994]). Ce module permet aussi d'identifier les "relations potentielles" (c'est-à-dire les relations admissibles du point de vue purement syntaxique) pouvant exister entre ces segments et ce, sans choisir en cas d'alternative. Ce sont les autres modules qui, disposant eux de connaissances à la fois syntaxiques et sémantiques, effectueront le choix entre plusieurs rattachements possibles.

L'analyseur morpho-syntagmatique construit donc un arbre contenant les segments qu'il a déterminés ainsi que les liens entre ces segments. Cet arbre devra être élagué (suppression des branches/liens inutiles) et étoffé (ajout des données sémantiques liées aux différents segments) par les autres modules de l'analyseur linguistique.

L'analyse des relations actancielles

Ce module, qui fera l'objet du chapitre 9, est le point central de l'analyseur. En effet, il a pour fonction non seulement de déterminer le squelette syntaxique de la phrase mais aussi de faire appel aux autres modules syntaxico-sémantiques pour étoffer ce dernier. La première fonction de ce module consiste donc à élaguer les branches inutiles de l'arbre fourni par l'analyseur morpho-syntagmatique pour en extraire le squelette syntaxique de la phrase et construire ainsi la structure actancielle de chaque procès (détermination du sujet, des compléments essentiels et circonstanciels de chaque verbe, rôle sémantique de chaque actant, type sémantique de chaque circonstant, etc.). Comme on le voit, c'est d'abord et avant tout autour des verbes que cette construction est organisée. Il faut noter cependant que le même type de traitements s'applique aussi aux constructions à tête nominale ou adjectivale. De plus, il faut remarquer que ces traitements sont récursifs : la structure interne d'une subordonnée est ainsi rattachée au niveau supérieur à l'élément dont elle dépend. Pour effectuer cette construction, ce module utilise principalement les données fournies par le lexique qui contient pour chacune de ses entrées les constructions syntaxiques admissibles avec indication des classes sémantiques possibles des composants (animé, humain, lieu, etc. - cf. section 9.1, page 161).

Le squelette ainsi déterminé étant incomplet, il doit être étoffé par les données sémantiques obtenues par appel aux autres modules syntaxico-sémantiques. De plus, la résolution de certaines ambiguïtés peut nécessiter des appels au sous-système de construction de la scène. Ce n'est qu'à la fin de ce processus d'interactions que l'analyse sera terminée. La sortie de ce module est alors constituée par la description des procès ainsi que les relations entre entités et procès, c'est-à-dire la description des contraintes sémantiques liées aux procès.

Le module d'identification des entités

Ce module est chargé de la création d'entités nouvelles et du repérage d'entités déjà introduites précédemment dans le texte. Pour cela, il utilise un ensemble de connaissances linguistiques sur les phénomènes d'anaphore ainsi qu'une liste (historique) des entités qui ont déjà été créées (dont une partie est contenue dans le “modèle des attentes du lecteur”) et les expressions linguistiques qui leur correspondent dans le texte.

Le principe utilisé pour le repérage des anaphores est inspiré des travaux de M. Ariel [Ariel, 1990] et mis en place par M. Dupont [Dupont, 1995]. Après détection d'une anaphore, ce module doit renvoyer le symbole correspondant au module appelant qui peut ainsi continuer et compléter son analyse. Dans le cas contraire (quand il n'y a pas d'anaphore), une nouvelle entité, à laquelle est associé un nouveau symbole typé selon la classe sémantique de l'unité (humain, véhicule, lieu, ...), est renvoyée ainsi qu'une indication de création.

Ce module est principalement appelé par le module d'analyse des relations actancielles pour chaque groupe susceptible de représenter une unité (en général, un groupe nominal ou un pronom) mais peut aussi être appelé par le module aspectuo-temporel (création des entités correspondant aux intervalles de temps) ainsi que par le module spatio-temporel (création et repérage des entités spatiales).

Ce module qui est en cours de réalisation au sein de notre équipe ([Dupont, 1995] et [Dupont, 1996]), ne sera pas plus décrit dans ce mémoire.

Le module d'analyse des relations aspectuo-temporelles

Ce module peut être appelé par le module d'analyse des relations actancielles ou par le module d'analyse des relations spatio-temporelles. Il a été conçu par Y. Legendre [Legendre, 1995] à partir des travaux de Laurent Gosselin ([Gosselin, 1992], [Gosselin, 1993] et [Gosselin, 1995]) et est chargé d'établir les relations temporelles explicitées dans le texte. Pour cela, il utilise un ensemble de règles qui portent à la fois sur le type lexical du procès indiqué par le verbe, sur les marques de temps verbal, sur les déterminants des compléments essentiels, sur les adverbes et compléments circonstanciels temporels, y compris les propositions subordonnées. Son originalité essentielle consiste à introduire deux types principaux d'intervalle de temps :

- d'une part, des intervalles, notés IP_i (intervalle de procès), liés aux bornes des procès et qui correspondent à l'emploi du verbe à l'infinitif;
- d'autre part des intervalles, notés IM_i (intervalle de monstration), qui correspondent à ce qui est effectivement montré du procès, c'est-à-dire à la partie du procès dont la réalisation est certaine.

Afin de clarifier ces notions d'intervalles, nous allons montrer, sur un exemple, les relations et intervalles déterminés par le module d'analyse des relations aspectuo-temporelles et préciser leur signification.

Dans le cas de la phrase *je traversais le carrefour quand un camion percuta ma voiture*, il existe deux procès : un correspondant à l'action de *traverser le carrefour*, l'autre correspondant à celle de *percuter la voiture*. À chacun de ces procès sont associés deux intervalles (IP_1 et IM_1 pour le premier procès, IP_2 et IM_2 pour le second). L'action de *traverser un carrefour* signifiant "aller d'un bord à l'autre du carrefour", la borne inférieure de IP_1 correspond au début de la traversée, la borne supérieure de IP_1 correspond à la fin de la traversée. Cependant, rien, dans les données linguistiques, ne permet de dire si le procès *traverser le carrefour* a vraiment eu lieu : seules des connaissances encyclopédiques conduiront à conclure que l'énonciateur n'a sûrement pas pu terminer son déplacement. Cependant, les marques aspectuo-temporelles présentes dans cette phrase imposent des relations précises entre les intervalles liés aux procès et les intervalles sur lesquels portent les assertions, même si l'ensemble de ces relations ne suffit pas à déterminer complètement le cours des événements (cf. figure 7.4). Ainsi la marque de l'imparfait sur *traverser* indique que l'intervalle de temps IM_1 sur lequel porte l'assertion est strictement inclus dans l'intervalle IP_1 correspondant à l'accomplissement du procès (ce qui explique que l'on ne puisse rien asserter sur la fin du procès). De même, la marque du passé simple implique que les intervalles IP_2 et IM_2 sont égaux et la construction de la proposition circonstancielle introduite par *quand* indique que les intervalles IM_1 et IM_2 sont consécutifs et adjacents (la borne supérieure de IM_1 est égale à la borne inférieure de IM_2). De plus, les intervalles IM_1 et IM_2 sont forcément antérieurs à l'instant d'énonciation (représenté par l'intervalle IE_0). D'une manière générale, on peut, comme l'a montré Laurent Gosselin (ainsi que W. Klein [Klein, 1994] dans une approche similaire de l'anglais et de l'allemand), représenter l'ensemble du système aspectuo-temporel du français en termes de relations entre ces différents types d'intervalles : ce sont ces relations qui constituerons l'apport du module d'analyse des relations aspectuo-temporelles au résultat de l'analyseur linguistique.

En général, les relations aspectuelles sont des relations entre IP_i et IM_i, alors que les relations proprement temporelles (présent, passé, futur) sont des relations entre IM_i et le moment de l'énonciation, qui est aussi un intervalle noté IE_0 (d'autres intervalles d'énonciation IE_i peuvent être ajoutés dans le cas de discours rapportés).

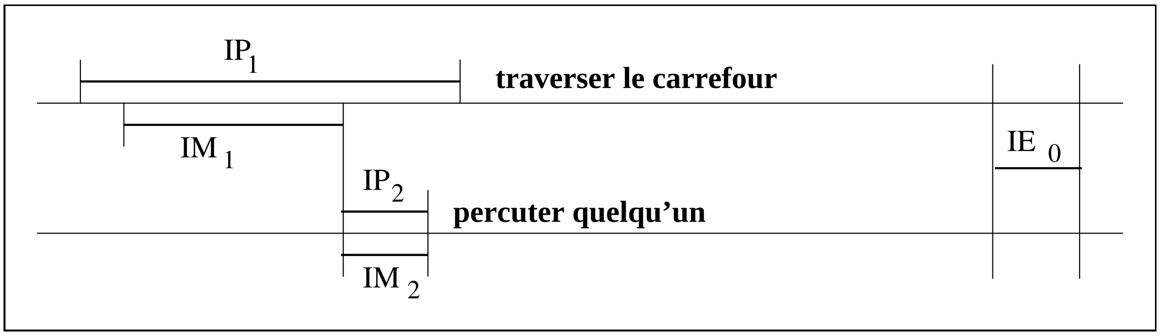


FIG. 7.4 - *Relations aspectuo-temporelles correspondant à : "je traversais le carrefour quand un camion percuta ma voiture"*

Le rôle de ce module est donc de créer ces intervalles et de déterminer les relations qu'ils entretiennent. Ces relations sont ajoutées à la sortie de l'analyseur, et l'intervalle de type IP_i est renvoyé au module appelant : en effet, dans le résultat de l'analyseur linguistique, toutes les relations actancielles seront indexées par cet intervalle car elles sont "potentiellement" vraies sur IP_i. C'est le sous-système de construction de la scène qui déterminera, en fonction des différents procès, des relations entre les différents intervalles liés à un même procès et des relations entre intervalles liés à des procès différents, l'intervalle de temps sur lequel est assertée chacune des relations actancielles.

Ce module s'appuie sur une première version autonome réalisée par Y. Legendre avec l'aide de L. Gosselin [Legendre, 1995].

L'analyse des relations spatio-temporelles

Ce module est chargé d'établir les relations spatiales (cf. [Mathet, 1995]) et en particulier celles liées aux mouvements, ce qui implique aussi de prendre en compte le temps. Il est appelé par l'analyseur des relations actancielles dès qu'un procès fait référence à l'espace, soit par le contenu lexical du procès lui-même (en particulier pour les verbes de mouvement), soit par l'intermédiaire d'un complément circonstanciel de lieu. Comme le module précédent, il doit alors utiliser un ensemble spécialisé de règles qui portent sur les différentes marques spatiales présentes dans la proposition étudiée pour en tirer les contraintes spatiales qui constitueront son apport à la sortie du système. En particulier, c'est ce module qui doit traiter les prépositions indiquant une localisation ou un mouvement (*à*, *dans*, *sur*, *vers*, etc.) pour les traduire par des relations spatiales précises.

L'une des difficultés essentielles provient du fait que les traitements de ce module sont centrés sur la notion de repère spatial, défini par une position donnée dans l'espace et une direction privilégiée : en effet, la plupart des indications spatiales (*à droite*, *en face*, *sur le côté*, etc.) sont relatives à un tel repère, qui peut être, suivant les cas, un repère lié à l'énonciateur, à l'un des actants en mouvement, à un point de vue extérieur d'où est décrite la scène, etc.. Or, les données linguistiques sont en général insuffisantes pour déterminer à elles seules de quel repère il s'agit. En effet, prenons les exemples suivants : *Je roulais sur la partie droite de la chaussée* et *Je roulais à droite*. Dans le premier exemple, *la partie droite de la chaussée* peut avoir deux significations : la partie rectiligne de la route ou la partie de la route située le plus à droite du milieu de celle-ci. Seule la seconde interprétation de cette partie de phrase nécessite un repère et dans ce cas, le fait que *de la chaussée* qualifie *partie droite* permet à ce module de préciser que l'indication spatiale *partie droite* est relative à un repère lié à la chaussée. Dans le second exemple, par contre, aucune indication ne permet de rattacher le repère sur lequel porte l'expression *à droite* à une route ou à un objet quelconque. Le module d'analyse des relations spatio-temporelles ne pourra donc que donner certaines contraintes explicites sur la nature de ce repère, et c'est bien sûr au sous-système de construction de la scène de le spécifier complètement, en fonction de connaissances sur le domaine, de la cohérence globale de la scène et des entités précédemment spécifiées.

Ce module doit pouvoir interagir étroitement avec le module aspectuo-temporel dans la mesure où, d'une part le contenu lexical des verbes de mouvement s'exprime par des relations qui font intervenir les bornes de l'intervalle IP lié au procès (et non pas l'intervalle de monstration IM : cf. l'exemple de *traverser*), et où d'autre part la notion même de mouvement implique une correspondance entre événements spatiaux et événements temporels (comparer *Je roulais en direction de Paris jusqu'à midi* et *Je roulais en direction de Paris jusqu'au premier village*).

Une première étude de ce module a été réalisée, au sein de notre équipe, par Y. Mathet [Mathet, 1995].

7.3 Choix effectués pour l'implémentation

Pour effectuer l'implémentation d'un système réparti comme le nôtre, il existe plusieurs possibilités : soit à un module on fait correspondre un processus ayant un seul fil de contrôle (processus mono-thread) et les communications entre modules s'effectuent par l'intermédiaire de boîtes aux lettres (représentées par des files de messages ou des pipes Unix), soit on associe à l'ensemble du système un seul processus ayant plusieurs fils de contrôle (processus multi-threads). Les avantages de la première solution résident essentiellement en la portabilité du logiciel sur tout système Unix ainsi qu'en la possibilité d'avoir des modules développés dans des langages différents. Un des principaux inconvénients de cette méthode concerne la gestion de la mémoire partagée par l'ensemble des modules : le programmeur doit alors non seulement gérer les accès en lecture et en écriture mais aussi les allocations et désallocations. Si la seconde solution n'est pas aussi facilement portable que la première (le portage sur un système ne disposant pas de multi-threading nécessite l'écriture d'un gestionnaire de la mémoire dynamique accessible dans un segment de mémoire partagé par tous les processus)[43] elle n'a pas l'inconvénient d'obliger le programmeur à gérer la mémoire partagée. En effet, un thread[44] étant un élément du système permettant de contrôler un ensemble d'instructions exécutées par un ou plusieurs programmes [Sun, 1993], un programme multi-threads est un programme auquel on associe plusieurs threads et où un thread permet de contrôler une partie des instructions du programme (on pourra par exemple associer à chacun des modules un thread différent). Dans ce style de programme, l'allocation en mémoire commune est donc entièrement gérée par le système.

Dans notre cas, les données communes à l'ensemble des modules étant importantes (l'ensemble du résultat de l'analyse effectuée par l'analyseur morpho-syntagmatique) et la gestion de la mémoire partagée étant délicate à mettre en place (essentiellement au niveau de l'allocation et de la désallocation), nous avons préféré utiliser un programme multi-threads plutôt qu'un ensemble de programmes mono-thread. Ce style de programme présente cependant un inconvénient au niveau des différents accès mémoire réalisés. En effet, dans un programme multi-threads il est nécessaire de contrôler les accès en écriture sur les différentes données communes de façon à éviter que deux modules n'écrivent en même temps sur la même donnée. Cependant, du fait des caractéristiques de notre architecture en ce qui concerne les données communes (seul le module d'analyse des relations actancielles a le droit de modifier ces données, les autres modules n'ayant que le droit de les consulter), nous ne risquons pas d'avoir deux

43. Il faut noter cependant que de plus en plus de systèmes (Solaris, OSF, ...) répondent à la norme POSIX 1003.e (norme pour l'implémentation des threads) et un programme multi-threads sera de plus en plus facilement portable sur ces systèmes.

44. Nous nous excusons de continuer en anglais, la traduction française “fil de contrôle” ne nous semblant pas très heureuse.

modifications de la même donnée en même temps.

La programmation multi-threads impose un certain nombre de contraintes sur le système d'exploitation et le langage utilisés : le système doit être capable de gérer le multi-threading et le langage doit comporter des fonctionnalités concernant ce type de programmation. L'environnement Unix sur SUN auquel on associe le langage `C` ou le `C++` permet de mettre en place une telle programmation. Au niveau du langage, notre choix s'est porté sur le `C++`, d'une part pour des raisons de génie logiciel [Stroustrup, 1989], d'autre part pour son caractère objet dont nous verrons l'utilité dans la suite de cette partie.

Cette première implémentation de l'analyseur linguistique est donc relativement indépendante du système utilisé (seule, comme nous venons de le voir, la gestion de la mémoire est dépendante du système mais du fait des possibilités de surcharge des opérateurs offertes par le `C++` et de l'évolution des systèmes vers une implémentation généralisée des threads, ceci ne devrait pas poser de problème) et permet à l'ensemble des intervenants dans ce projet de trouver un cadre de développement général.

7.3.1 Protocole de communication

Dans cet analyseur, l'échange de messages s'effectue par l'intermédiaire de "boîtes aux lettres" : quand un module (un thread) doit envoyer un message à un autre module, il le dépose dans la boîte aux lettres de ce dernier. Tous les modules ont le droit d'écrire dans une boîte aux lettres mais un seul (le module propriétaire de la boîte aux lettres) a l'autorisation et la possibilité d'enlever un message.

Le protocole de communication que nous avons mis en place comporte 4 types de message différents : les indications, les requêtes, les réponses et les réponses intermédiaires. Les indications, comme leur nom l'indique, permettent d'indiquer un événement (début de texte, début de phrase ...) à un module. Les requêtes permettent de poser une question à un module donné et demandent nécessairement une réponse. Les réponses et les réponses intermédiaires permettent d'envoyer, au module appelant, une réponse à une requête précise. Une réponse intermédiaire est utilisée quand un module a plusieurs réponses à envoyer pour une requête donnée : elle signifie que le message ne correspond qu'à une partie de la réponse. Une liste de réponses intermédiaires doit toujours être terminée par une réponse : l'émetteur de la requête peut alors considérer celle-ci comme terminée.

Les messages envoyés sont composés de différentes parties dont le nombre varie en fonction du type du message. Une indication ne comporte qu'une seule partie qui correspond au texte du message. Une requête est composée, en plus du texte du message, du nom de l'expéditeur (utilisé lors de l'envoi de la réponse) et d'un numéro de requête (numéro géré par l'expéditeur et qui lui permet de retrouver l'état dans lequel il doit se placer sur réception de la réponse). Les réponses (réponse ou réponse intermédiaire), quant à elles, contiennent, en plus du texte du message, le numéro de la requête à laquelle elles correspondent : cela permet au module expéditeur de savoir à quelle requête correspond cette réponse.

Afin de clarifier le protocole de communication, nous allons maintenant donner et expliquer une partie des messages échangés par les différents modules dans la cadre de l'analyse de la phrase *Je roulais sur la partie droite de la chaussée* (les explications relatives aux relations et entités seront données lors de l'analyse détaillée page 181).

- Indications portant sur l'état d'avancement de l'analyse.
  `RA → ID indication DEBUT TEXTE`
  `RA → ID indication DEBUT PROPOSITION`

- Demande de création d'une entité de type déplacement (entité liée à rouler)
  `RA → ID requête 24 nouvelleentite ‘‘déplacement’’`
- Demande d'exécution d'une instruction liée au schéma de *partie*
  `RA → ST requête 40 restriction_spatiale ‘‘droite’’ ‘‘de la chaussée’’`

- Réponse à la requête 24 : création de l'entité E_0
  `ID → RA réponse 24 nouvelleentite` E_0

- Ajout des entités et relations liées à l'exécution de l'instruction “restriction_spatiale”
  `ST → RA réponse int. 40 ajoutentiteassociee` SR_0
  `ST → RA réponse int. 40 ajoutentite` Rep_0
  `ST → RA réponse int. 40 ajoutrelation ‘‘inclus’’`
  `ST → RA réponse int. 40 ajoutrelation ‘‘lié’’`
  `ST → RA réponse int. 40 ajoutrelation ‘‘à droite’’`
  `ST → RA réponse int. 40 ajoutrelation ‘‘droit’’`
- fin de la liste des réponses intermédiaires
  `ST → RA réponse 40 fininstruction`

7.3.2 Fonctionnement et implémentation des modules

Au niveau des modules, nous devions, pour éviter un blocage complet du système, respecter les contraintes suivantes :

- un module doit pouvoir, à tout moment, envoyer une requête;
- un module doit pouvoir, même s'il est en attente d'une réponse à une requête, être capable de répondre à d'éventuelles requêtes venant d'autres modules : l'attente d'une réponse ne doit pas être bloquante.

Pour répondre à cet ensemble de contraintes, chaque module doit gérer une "table des requêtes en attente" qui lui permet de reprendre, sur réception d'une réponse, un traitement interrompu par l'envoi d'une requête dont le résultat est déterminant pour la suite de l'analyse. Cette table doit, de plus, lui permettre de prendre en compte le cas où, pour répondre à une requête qui lui était posée, il a dû envoyer une requête à un autre module. Cette table contient donc, pour chaque requête envoyée par le module, l'adresse de la fonction à utiliser pour poursuivre l'analyse en cours ainsi que les coordonnées (expéditeur et numéro) de la requête en attente (requête d'un autre module ayant nécessité cet envoi de requête).

Les différents modules composant l'analyseur linguistique étant développés par des personnes différentes, il nous semblait important de mettre en place un "prototype de module" qui aurait pour fonction de gérer cette table des requêtes et permettrait ainsi, d'une part, une certaine homogénéité des modules (principe de fonctionnement identique pour chacun des modules - cf. table 7.1), d'autre part, la transparence au niveau du support de communication (boîtes aux lettres). Ce prototype facilite non seulement le développement des modules mais aussi tout changement au niveau de l'implémentation des modules (passer d'une implémentation multi-threads à une implémentation sous forme de plusieurs programmes communiquant par boîtes aux lettres) et du système de communication.

Cette notion de prototype est très facile à mettre en place en `C++` du fait du caractère objet de ce langage est donc des notions d'héritage qui lui sont liées. Nous avons donc défini, pour les modules, une classe de base (`ControlModule`) ainsi qu'un certain nombre de méthodes de base (`EnvoieRequête`, `EnvoieIndication`, `EnvoieRéponse`, `EnvoieRéponseInt` et `Serveur`) qui permettent de gérer les communications entre

```
Gestion des messages
-  regarder le contenu de sa boîte aux lettres
   -  Si celle-ci est vide Alors
      -  attendre un message
      -  sur réception d'un message, traiter le message
   -  Sinon
      -  traiter le message
-  continuer la gestion des messages
```

```
Traitement des différents messages
-  suivant le type du message :
      si indication :
      -  effectuer le traitement correspondant
      si requête :
      -  effectuer le traitement correspondant
      -  envoyer une réponse
      si réponse intermédiaire :
      -  stocker la réponse
      si réponse :
      -  stocker la réponse
      -  continuer le traitement en cours
```

TAB. 7.1 - *Algorithme de fonctionnement d'un module*

modules ainsi que celles avec les autres sous-systèmes et de gérer le principe de fonctionnement. Chacun des modules possède un contrôleur qui est une classe qui hérite de la classe `ControlModule` et qui surcharge les méthodes permettant d'effectuer un traitement en fonction d'un message donné (`TraiterRequête`, `TraiterRéponse` et `TraiterAutre` définies dans le classe `ControlModule`).

Le lancement des différents threads (c'est-à-dire des différents modules) est effectué par le thread "principal" (`GestionAL` - lié au programme principal) qui aura ensuite comme tâche de gérer les messages arrivant de l'extérieur de l'analyseur linguistique (c'est-à-dire les déposer dans la boîte aux lettres correspondant au destinataire) et d'envoyer les messages destinés aux autres sous-systèmes du système global. Nous obtenons alors un programme dont le contrôle est décrit dans la figure 7.5 (les flèches grisées représentent le contrôle venant ou allant vers les autres sous-systèmes du système global).

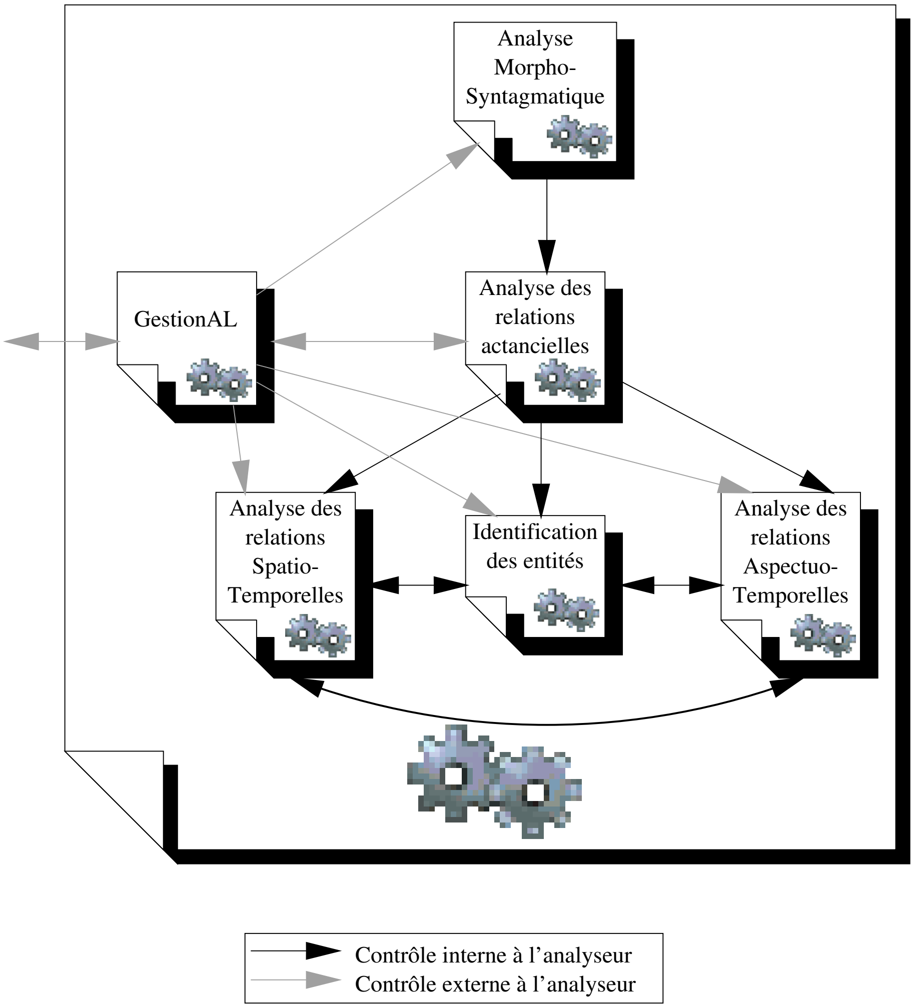


FIG. 7.5 - *Implémentation du contrôle dans l'analyseur linguistique*

Chapitre 8

L'analyseur morpho-syntagmatique

L'analyseur morpho-syntagmatique est chargé d'effectuer une première analyse du texte en n'utilisant que des connaissances morphologiques et syntagmatiques. Il a comme tâche d'effectuer un découpage du texte en syntagmes ainsi que de déterminer le découpage en propositions. Il doit aussi extraire des données syntaxiques les différentes relations pouvant exister entre les composants du texte (relations entre syntagmes ou entre propositions et relations reliant syntagmes et propositions).

L'idée de base utilisée pour déterminer les syntagmes est de regrouper les unités morphologiques en "segments maximaux sûrs" ; un segment maximal sûr étant la plus grande suite d'unités contiguës dont les relations syntaxiques sont déterminées de façon univoque par leur catégorie et leur ordre dans la chaîne.

Cet analyseur est un des modules dont la conception et la réalisation ont été effectuées au cours de cette thèse. Cependant, du fait de la nature de nos objectifs (mettre en place un premier système global complet dans le but de valider notre approche de la sémantique et non mettre en place un système capable de traiter tous les problèmes liés à l'étude de la langue), ce module n'est pas complet mais constitue simplement une base pour des travaux ultérieurs.

8.1 Fonctions de l'analyseur morpho-syntagmatique

Un principe qui figure parmi ceux qui, jusqu'ici, ont été les plus fréquemment employés en analyse morpho-syntagmatique pour déterminer la structure syntaxique d'une phrase consiste à utiliser les grammaires syntagmatiques (ou grammaires à règles de réécritures) issues des travaux de N. Chomsky [Chomsky, 1957] [45]. Cependant, la variété et la complexité de la syntaxe du français (avec, entre autres, la mobilité des compléments circonstanciels au sein de la phrase) nécessite un nombre de règles trop important pour que ce genre de grammaire soit efficace. Nous nous sommes donc tournés vers une analyse du style de celles effectuées par [Vergne, 1995] pour le français et [Georgiev, 1991] pour l'anglais, principes d'analyse qui actuellement sont en plein essor. Ce style d'approche consiste, non pas à mettre en place des règles permettant de définir complètement la phrase, mais à mettre en place des règles pour définir un certain nombre de groupes syntaxiques et à déterminer ensuite les relations existant entre ces différents groupes.

L'approche employée par J. Vergne consiste à décomposer la phrase en blocs, chacun de ces blocs étant composé de "séquences" (nominales ou verbales) et de "clips" (conjonctions, prépositions, signes de ponctuation, pronoms, etc.). Les séquences nominales sont composées d'un seul nom et des mots qui l'accompagnent (déterminant, adjectif, adverbe d'adjectif et partitif) ; les séquences verbales se construisent autour du verbe auquel peuvent être rattachés des voisins immédiats de type auxiliaire, adverbe, clitique, etc. La démarche d'analyse est basée sur une construction progressive de la syntaxe selon des critères propres à chaque niveau hiérarchique. Le système implémenté a pour but d'effectuer un découpage du texte en blocs en utilisant un dictionnaire restreint (uniquement les clips et les verbes). L'inconvénient majeur de cette implémentation est la non-déclarativité des règles. En effet, les règles utilisées pour construire les blocs sont implémentées de façon procédurale et sont donc difficilement compréhensibles et modifiables pour toute personne n'ayant pas participé à l'implémentation.

45. [Chomsky, 1969] pour la traduction française.

8.1.1 Les segments maximaux sûrs

Comme nous l'avons dit précédemment, un segment maximal sûr est composé de la plus grande suite d'unités contiguës dont les relations syntaxiques sont déterminées de façon univoque par leur catégorie et leur ordre dans la chaîne. Au total, nous avons recensé environ une dizaine de regroupements possibles [Picand, 1993] dont nous allons maintenant donner une description.

Les segments de la catégorie `SN` (segment nominal) correspondent aux groupes à tête nominale. En règle générale, ils sont composés du nom accompagné de son déterminant et de ses adjectifs antéposés (*une petite rue*). Ces groupes peuvent cependant contenir en plus une préposition (groupes nominaux prépositionnels - *sur la chaussée*) et/ou ne pas avoir de déterminant (le mur *de clôture*; *Fort trafic* sur la nationale). Par contre, nous ne pouvons pas intégrer au segment contenant un nom les adjectifs qui le suivent car ils ne sont pas forcément ses adjectifs postposés. En effet dans *la petite voiture blanche* blanche est bien un adjectif postposé de *voiture* alors que dans *une roue de voiture crevée*, *crevée* est un adjectif postposé de *roue* et non de *voiture*.

Les segments de la catégorie `SN_Pron` (segments pronominaux) représentent les groupes pronominaux composés d'un pronom personnel et d'une préposition. Ils représentent donc des suites comme *devant lui*.

Les segments de la catégorie `SN_propre` (segments nominaux propres) sont des segments nominaux dont la tête est un nom propre.

Les segments de catégorie `SV_conj` (segments verbaux contenant un verbe conjugué) représentent les groupes à tête verbale dont le verbe est conjugué. Ils sont composés du verbe conjugué, de ses pronoms clitiques[46] (sujets ou compléments - *je me trouve* dans le carrefour) et d'une négation (*je ne me trouve pas* dans le carrefour) mais ne contiennent pas le sujet ni les compléments du verbe quand ceux-ci sont des groupes nominaux comme dans le cas de *le véhicule roulait*.

Les segments de la catégorie `SV_p_pres` (segments verbaux contenant un verbe au participe présent) représentent les groupes à tête verbale dont le verbe est au participe présent. Ils sont composés du verbe, de ses clitiques, d'une négation et d'une préposition (*Voulant* dépasser ...- *en franchissant* la ligne blanche).

46. Un pronom clitique étant un pronom personnel sujet (je, tu, il, etc.) ou complément (le, en, y, etc.) se trouvant, sauf dans de rares exceptions (je, soussigné ...), à proximité immédiate du verbe.

Les segments de la catégorie `SV_p_passe` (segments verbaux contenant un verbe au participe passé) représentent les groupes à tête verbale dont le verbe est au participe passé. Ils sont composés uniquement du verbe.

Les segments de la catégorie `SV_inf` (segments verbaux contenant un verbe à l'infinitif) représentent les groupes à tête verbale dont le verbe est à l'infinitif. Ils sont composés du verbe, de ses clitiques, d'une négation et d'une préposition (*pour juger* l'étendue du bouchon - Voulant *dépasser* - *ne pas le juger*).

Les segments de la catégorie `SAdj` (segments adjectivaux) sont composés d'un adjectif et d'adverbes (*plus tellement vrai* - une roue de voiture *crevée*).

Les segments de la catégorie `SAdv` (segments adverbiaux) sont uniquement composés d'adverbes (*encore plus rapidement*).

Les segments de la catégorie `SQ_rel` (segments pronominaux relatifs) sont composés d'un pronom relatif et d'une préposition (*qui*, *que*, *sur lequel* ...).

Les segments de la catégorie `SQ_conj` (segments conjonctifs) sont composés d'une conjonction de subordination ou d'une locution conjonctive (*Alors que*).

Les segments de la catégorie `Scoord` (segments de coordination) sont composés d'une conjonction de coordination ou d'une locution de coordination.

Ces définitions des groupes sont bien entendu incomplètes et ne correspondent qu'à l'ensemble des définitions que nous avons dû mettre en place pour traiter une partie de notre corpus (corpus simplifié - cf. Annexe B.2, page XIV). Ces définitions, ainsi que la liste des regroupements possibles, seront donc amenées à être enrichies voire même transformées pour passer à une application plus importante.

8.1.2 Les propositions

Pour l'instant, cet analyseur cherche uniquement à repérer dans le texte les propositions principales ainsi que les propositions subordonnées relatives ou conjonctives (propositions subordonnées pour lesquelles la position du pronom permet de déterminer le début). Pour effectuer cette recherche des propositions nous ne disposons que de l'ordre des segments dans la phrase ainsi que des catégories des différents segments (on utilise, par exemple, le fait qu'un segment de catégorie `SQ_rel` marque le début d'une proposition relative). Malheureusement, à partir de ces seules indications, il n'est pas possible de déterminer avec exactitude l'unique proposition à laquelle appartient un segment donné. En effet, si on prend la phrase suivante : *J'ai su que vous étiez passés à 5 heures*, le segment *à 5 heures* peut être rattaché soit au verbe *savoir* (c'est à 5 heures que j'ai su que vous étiez passés), soit au verbe *passer* (j'ai su que c'est à 5 heures que vous êtes passés), ce qui donne deux découpages possibles pour cette phrase comme le montre la table 8.1.

	Découpage effectué
à 5 heures est rattaché à *savoir*	J'ai su [P1] · que vous étiez passés [P2] · à 5 heures [P1]
à 5 heures est rattaché à *passer*	J'ai su [P1] · que vous étiez passés à 5 heures [P2]

TAB. 8.1 - *Exemples de découpages en propositions*

Dans une phrase comme *J'ai su que vous étiez passés à 5 heures*, nous considérons qu'il existe deux noyaux propositionnels (*j'ai su* pour la principale et *que vous étiez passés* pour la subordonnée) ainsi qu'un segment dont le rattachement est imprécis, celui-ci pouvant être rattaché soit à la principale, soit à la subordonnée. Lors de la recherche des propositions, nous cherchons donc à déterminer, non pas l'ensemble des segments formant la proposition, mais le noyau propositionnel de chacune des propositions ainsi que les rattachements possibles pour les segments dont le rattachement est imprécis.

Pour pouvoir représenter ce type de situation, les propositions identifiées lors de la recherche des propositions sont composées de deux ensembles de segments : d'une part des "segments sûrs" (ceux pour lesquels il n'y a qu'une seule possibilité de rattachement - le noyau propositionnel), d'autre part des "segments possibles" (ceux pour lesquels le rattachement est imprécis). Un des travaux du module d'analyse des relations actancielles sera de déterminer, à partir des données lexicales qui sont en sa possession, à quelle proposition appartient chacun des segments possibles.

8.1.3 Les liens potentiels

Les liens que cet analyseur cherche à déterminer sont d'une part les liens pouvant exister entre différents segments d'une même proposition, d'autre part à quel segment d'une proposition de niveau supérieur peut se rattacher le connecteur d'une proposition subordonnée. Les liens entre propositions (liens proposition subordonnée → proposition principale, par exemple) sont calculés en même temps qu'est déterminée l'arborescence des propositions (c'est-à-dire lors de la construction des propositions). Pour les autres liens, nous parlons de liens potentiels car, malheureusement, il est impossible, en utilisant uniquement des données morpho-syntaxiques, de déterminer avec certitude le rattachement d'un segment dans sa proposition (nous verrons, section 9.3 page 172, les heuristiques nises en place au niveau de l'analyse sémantique pour effectuer les rattachements). En effet, dans la plupart des cas, des connaissances sémantiques seront nécessaires pour déterminer avec précision un rattachement.

Prenons les morceaux de phrases *le conducteur du camion bleu* et *le conducteur du camion ivre*. Ils ont tous les deux la même structure syntaxique (un segment nominal suivi d'un segment nominal prépositionnel et d'un adjectif) mais si dans le premier cas l'adjectif est à rattacher à *du camion*, dans le second cas l'adjectif est à rattacher à *le conducteur*. Le problème dans ce genre de cas est que seules des données sémantiques (indiquant, par exemple, que seule une personne peut être ivre[47]) permettent de faire le rattachement correct. Nous avons donc décidé de déterminer, au niveau de l'analyse morpho-syntagmatique, l'ensemble des liens réalisables au vu des données morpho-syntaxiques (dans le cadre de nos exemples, nous construirons donc à chaque fois deux liens : "adjectif" → *le conducteur* et "adjectif" → *du camion*) et de laisser, au module d'analyse des relations actancielles, le soin de déterminer lesquels sont à conserver. Ce travail est plus facilement réalisable par ce module du fait qu'il dispose de connaissances sémantiques.

47. Cependant, les données sémantiques ne suffisent pas toujours. En effet, comme le mentionne D. Kayser, si on ne permet pas au bateau d'être ivre, adieu Musset, Rimbaud, etc.

8.2 Structure de base utilisée

8.2.1 Choix et caractéristiques

Un des problèmes rencontrés pour mettre en place l'analyseur morpho-syntagmatique vient essentiellement de la polycatégorie d'un certain nombre de mots de la langue française. En effet, en français, les phénomènes de polycatégorie sont importants et peuvent être regroupés en deux catégories principales [Fuchs *et al.*, 1993] : les "polycatégories fortes" (ou homonymies catégorielles) et les "polycatégories naturelles" (ou dérivations impropres). Dans le cas des polycatégories fortes la même séquence correspond à différents homographes dont les significations, malgré une étymologie commune, sont fortement distinctes. Par exemple, une séquence comme *ferme* peut faire référence au substantif (l'exploitation agricole - *la ferme*), à l'adjectif (adjectif renvoyant à quelque chose de solide - *la viande ferme*) ou encore au verbe (verbe signifiant clore - *il ferme la porte*). Dans le cas des polycatégories naturelles (qui sont les cas les plus fréquents), la même séquence fait référence à des significations déductibles les unes des autres dans des emplois syntaxiques différents. Il existe par exemple, des cas de polycatégorie naturelle entre nom et adjectif (*informatique*, *petit*, *linguistique*, ...), entre nom et verbe à l'infinitif (*pouvoir*, *manger*, *rire*, ...), entre adjectif et adverbe (*clair*, *fort*, *juste*, ...).

Le fait d'associer à un "mot" une catégorie plutôt qu'une autre peut avoir des conséquences sur la construction des segments maximaux sûrs et donc sur l'analyse toute entière. Cela peut entraîner l'échec de l'analyse alors qu'un autre choix aurait conduit à un succès. En effet, le fait de considérer *ferme* comme un nom ou un adjectif dans la phrase *Il ferme la porte* rend le reste de l'analyse impossible. Pour résoudre ce problème il existe plusieurs solutions :

1. effectuer un backtrack sur le dernier choix effectué en cas d'échec de l'analyse ;

2. exécuter autant d'analyses qu'il y a de choix possibles ;

3. traiter en parallèle les différentes possibilités.

La première solution a l'avantage de permettre, si on a le bonheur d'effectuer directement les bons choix, de demander moins de travail que les autres solutions. Cette solution peut bien sûr être améliorée en donnant, aux diverses possibilités pour un même mot, une priorité fonction de leur emploi (on donnera, par exemple, une priorité plus forte au nom *pomme* qu'au verbe *pommer* - le nom *pomme* étant plus couramment employé que le verbe *pommer*) mais elle ne permet pas de pouvoir extraire toutes les analyses possibles dans le cas de textes ambigus (comme par exemple : *la belle ferme le voile*) contrairement aux deux autres solutions. L'avantage de la troisième solution par rapport à la seconde est d'effectuer l'analyse du texte en "une seule passe".

Pour mettre en place notre analyseur, et comme nous voulions éviter de traiter plusieurs fois les parties communes aux diverses analyses (éviter une trop importante explosion combinatoire), nous avons opté pour la troisième solution à laquelle nous avons ajouté un système de priorités. Afin de mettre en place cette dernière, il nous fallait tout d'abord déterminer une structure de données permettant de stocker, à chaque étape de l'analyse, les différentes possibilités. La structure de données qui nous a semblé la mieux adaptée à notre problème est le graphe.

Les possibilités de catégorie pour un même mot n'étant pas toutes de même niveau (*pommes* a plus de chances de faire référence au nom *pomme* qu'au verbe *pommer*), nous voulions maintenir cette différence de niveau à l'intérieur de nos graphes. Pour cela nous avons utilisé une structure de graphe spéciale que nous avons appelée "Graphe des Séquences Pondérées" (GSP). Dans un tel graphe, à chaque arc est associé un ordre de priorité qui correspond à un niveau de vraisemblance, par rapport à l'analyse, de la donnée portée par l'arc (plus cet ordre est faible, plus la donnée portée par l'arc est vraisemblable). Les arcs d'ordre de priorité 0 correspondant aux arcs les plus vraisemblables, les chemins d'ordres de priorité cumulés les plus faibles correspondent aux chemins les plus vraisemblables et sont donc ceux qui contiennent l'analyse qui, pour l'instant, semble la plus plausible. Les nœuds d'un GSP correspondent aux coupures effectuées dans le texte par l'analyse, tandis que les arcs portent la signification associée à la partie du texte se trouvant entre deux coupures. Cette sémantique attachée aux nœuds nous garantit d'avoir, pour chacun des graphes, non seulement un seul nœud de départ mais aussi un seul nœud d'arrivée : la dernière coupure effectuée dans le texte est la fin du texte et est commune à toutes les analyses.

Cette structure de données peut être considérée comme un CHART auquel nous aurions ajouté un système de priorité. Cependant, l'utilisation que nous en faisons est différente de celle des CHART effectuée dans le cadre d'une analyse par CHART ([Kay, 1973] [Kaplan, 1973] [Ramsay, 1985] [Wirén, 1992]). En effet, alors que l'analyse par CHART consiste à développer sur un seul et unique CHART pour l'ensemble de l'analyse, nous verrons dans la section 8.3 qu'un GSP donné correspond à un seul niveau d'analyse. Nous avons choisi cette solution de façon à ne pas mélanger, dans une même structure, des éléments de natures différentes.

Nous avons imposé, et ce afin d'éviter d'avoir des ordres de priorité minimaux trop élevés, d'avoir, pour chaque nœud des GSP, au moins un arc d'ordre 0. Ceci nous oblige à normaliser tout graphe qui vient d'être construit ou modifié. Le principe de normalisation est basé sur un parcours complet des nœuds du graphe et consiste à modifier les ordres de priorité des arcs issus d'un nœud sans ordre de priorité à 0 en reportant la valeur modifiée sur les arcs précédents (cf. algorithme, table 8.2). Le report de la valeur de l'ordre de priorité sur le nœud précédent impose de commencer la normalisation d'un graphe par le dernier nœud du graphe (ou nœud final). Dans la figure 8.1, les graphes de la colonne de gauche sont incorrects mais équivalents à ceux de la colonne droite qui, eux, sont normalisés.

- **Pour chaque** `noeud n'ayant aucun arc d'ordre de priorité égal à 0`
  Faire
  - `déterminer le plus petit des ordres de priorité existant`
  - `soustraire cet ordre à ceux des arcs issus de ce noeud`
  - **Si** `le noeud a un prédécesseur` **Alors**
    - `reporter cette valeur d'ordre sur l'arc situé entre le noeud courant et le noeud précédent`

TAB. 8.2 - *Algorithme de normalisation d'un GSP*

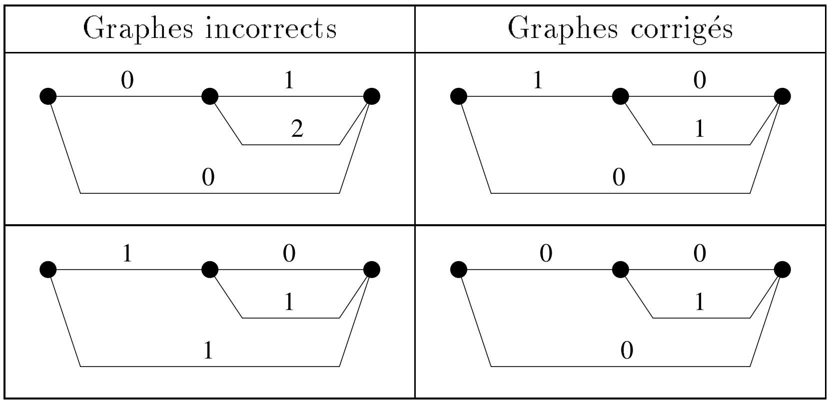


FIG. 8.1 - *Exemples de GSP avant et après normalisation*

Les ordres de priorité associés aux arcs dépendent de deux choses : d'une part des ordres de priorité initiaux se trouvant dans les bases de connaissances utilisées (le verbe *pommer* étant peu fréquent, il aura un ordre de priorité initial plus fort que le nom *pomme*), d'autre part de règles générales précisées pour chacune des tâches pour lesquelles c'est nécessaire. Par exemple, lors de l'analyse morphologique, on utilise la règle suivante : les unités morphologiques représentant des mots composés ou des expressions figées ont un ordre de priorité plus faible que les autres (et donc une priorité plus forte).

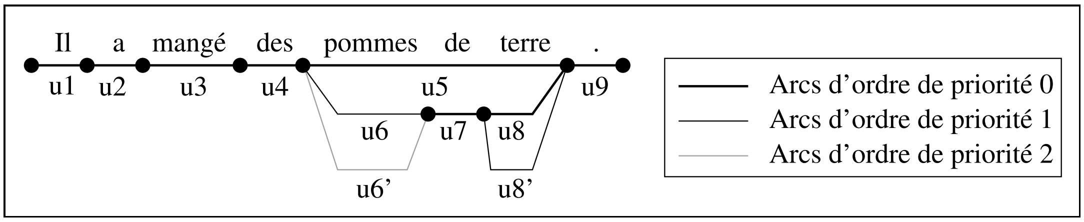


FIG. 8.2 - *GSP d'unités morphologiques de "Il a mangé des pommes de terre."*

Ce calcul des priorités, nous donne, dans le cadre de l'analyse de la phrase *Il a mangé des pommes de terre*, le GSP d'unités morphologiques de la figure 8.2. Dans cet exemple, l'ordre de priorité de u5 (*pomme de terre*) est plus faible que ceux de u6 et u6' parce qu'elle regroupe plusieurs "mots", pour placer u6 (*pomme*) avant u6' (*pommer*) et u8 (*terre*) avant u8' (*terrer*), on utilise les ordres de priorité initiaux donnés dans le dictionnaire.

8.2.2 Implémentation des graphes

Comme nous l'avons vu précédemment, la structure de données de base utilisée est le graphe. Cependant, les GSP utilisés ne se limitent pas à des GSP d'unités morphologiques mais, comme nous pouvons le voir sur la figure 8.6 page 133, nous avons aussi besoin de GSP de segments maximaux sûrs et même de listes de GSP. Ces divers graphes, tout en étant différents au niveau de leurs arcs (les unités portées par ces derniers étant de natures différentes), doivent cependant avoir un certain nombre de points communs comme, par exemple, la gestion de la normalisation des graphes. De ces constatations émerge la notion d'objet et aussi celle d'héritage entre diverses classes d'objets, notions qui font parties des caractéristiques du `C++`.

Pour représenter nos graphes, nous avons choisi de définir une classe de base (la classe `GrapheSimple`) ayant comme attributs, d'une part, un pointeur sur le premier nœud du graphe et un sur le dernier, d'autre part, la liste des nœuds du graphe ainsi que celle des arcs. Ces deux dernières listes nous permettent d'effectuer plus rapidement et plus simplement un certain nombre d'opérations ne nécessitant pas nécessairement un parcours du graphe suivant "l'ordre des nœuds et des arcs" (destruction du graphe, recherche dans le graphe d'un nœud ou d'un arc bien précis, etc.). Les nœuds sont représentés par des objets de la classe `Noeud` ayant comme attributs deux listes de pointeurs sur des arcs : d'une part celle des arcs issus de ce nœud, d'autre part celle des arcs arrivant sur ce nœud; ces deux listes étant ordonnées en fonction de la priorité des arcs. Ces deux listes nous permettent de parcourir le graphe dans les deux sens : du premier au dernier nœud (nous rappelons que tout GSP n'a qu'un seul nœud final) et du dernier au premier. Les arcs sont représentés par des objets de la classe `Arc` ayant comme attributs d'une part, un pointeur sur le nœud de départ de l'arc et un sur le nœud d'arrivée, d'autre part un pointeur sur la donnée portée par l'arc c'est-à-dire sur une unité morphologique, un segment .... La classe de base pour les données portées par les arcs est la classe `Unité`. Celle-ci a pour unique fonction de permettre le regroupement, dans un même arbre d'héritage, des différentes données à mettre sur les arcs. Cette classe qui n'a donc aucun attribut et ne contient que des méthodes virtuelles (méthodes qui seront surchargées lors de la création d'une classe "fille"), est une classe abstraite et ne peut donc avoir d'instance propre.

Pour implémenter les diverses listes demandées par cet ensemble de classes, nous avons utilisé les listes GNU. La figure 8.3 donne une représentation graphique de cet ensemble de classes. Un certain nombre de méthodes, utiles pour tous les graphes, ont été définies au niveau de la classe `GrapheSimple`. Ces méthodes permettent d'ajouter un arc ou un nœud, d'extraire un chemin de priorité p, de recopier un graphe et surtout de normaliser un graphe. De plus chaque type de GSP dispose de ses propres méthodes (par exemple, la classe représentant les GSP de segments dispose d'un constructeur à partir d'un GSP d'unités morphologiques et d'une liste de règles).

Pour pouvoir parcourir les divers graphes, nous avons mis en place des classes d'itérateurs. Les objets de ces classes ont pour fonction de permettre la simplification de divers parcours d'un graphe. Actuellement, nous avons défini deux itérateurs possibles : un itérateur en profondeur (classe `IterGrapheSimpleProf`) et un itérateur en largeur (classe `IterGrapheSimpleLarg`). Quelque soit la classe d'itérateur écrite, un certain nombre d'attributs et de méthodes sont communs, par conséquent nous avons écrit une classe `IterGrapheSimple` dont les deux précédentes (et toutes les classes d'itérateur que nous serons amenés à écrire) héritent.

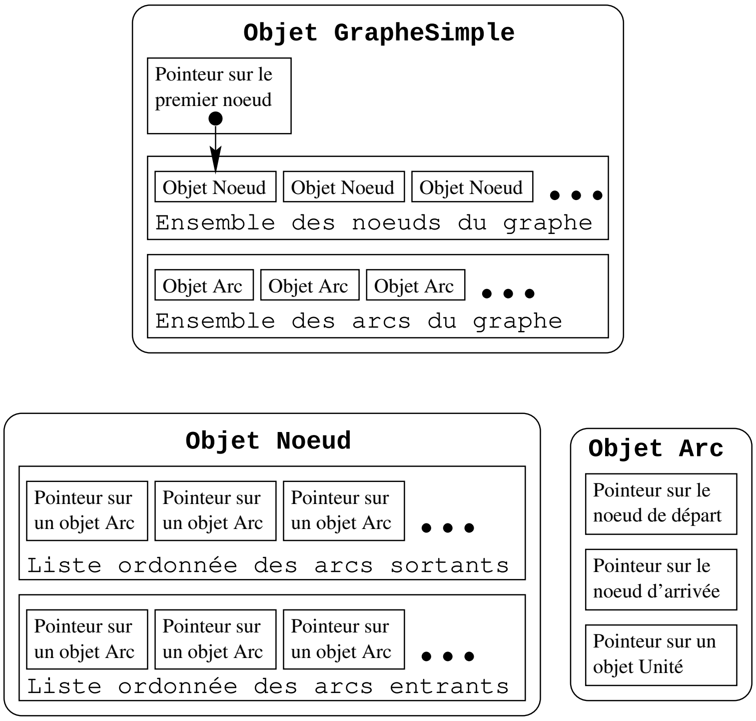


FIG. 8.3 - *Structure de base utilisée*

A partir de cette classe `GrapheSimple`, nous pouvons alors définir tous les GSP dont nous avons besoin. Pour cela, il faut définir une nouvelle classe (`GrapheMorph` par exemple) qui hérite de `GrapheSimple` (ce qui permet d'utiliser toutes les fonctionnalités définies dans `GrapheSimple`) ainsi qu'une nouvelle classe de données à mettre sur les arcs qui, elle, hérite de la classe `Unité` (une classe `UnitéMorph` par exemple).

De plus, nous avions besoin de listes de graphes (pour pouvoir effectuer la distinction entre les diverses phrases du texte, on associe un graphe à chaque phrase, le texte est donc représenté par une liste de graphes) ainsi que d'une structure permettant de regrouper à la fois des graphes et des listes de graphes (structure permettant de stocker les différentes étapes de la segmentation : certaines de ces étapes travaillent sur des graphes, d'autres sur des listes de graphes).

Pour définir les listes de graphes, il suffit de définir une classe `ListeDePtGraphes` qui, en fait, est une liste de pointeurs sur des objets de la classe `GrapheSimple` de façon à permettre de stocker, à l'intérieur, des graphes de n'importe quel type à partir du moment où ils héritent de cette classe de base. Cette classe `ListeDePtGraphes`, comme l'ensemble des listes définies pour cette structure de données, est construite à partir des listes GNU et a comme méthodes, en plus de celles définies dans la classe

GNU (ajout, suppression, parcours, etc.), des méthodes qui permettent d'effectuer une même opération sur chacun des graphes de la liste (c'est le cas, entre autres, de l'extraction d'un chemin de priorité p ou des méthodes permettant d'effectuer l'affichage de l'ensemble des graphes de la liste).

Pour définir la structure permettant de stocker les résultats des différentes étapes de l'analyse morpho-syntagmatique, il fallait une liste capable de stocker à la fois des graphes et des listes de graphes. Pour cela, il existe deux possibilités :

1. Définir une classe abstraite `Élément` dont héritent les classes `GrapheSimple` et `ListeDePtGraphes` et définir une classe `ListeDePtÉléments` qui correspond à une liste de pointeurs sur des objets de la classe `Élément` (ce qui donne la représentation graphique et l'arbre d'héritage de la figure 8.4).

2. Définir une classe abstraite `Graphe` dont héritent les classes `GrapheSimple` et `ListeDePtGraphes` et modifier la définition de la classe `ListeDePtGraphes` comme étant, non plus une liste d'objets de la classe `GrapheSimple`, mais une liste d'objets de la classe `Graphe`. Dans ce cas, la classe `ListeDePtGraphes` sera capable de stocker à la fois des graphes et des listes de graphes (cf. figure 8.5).

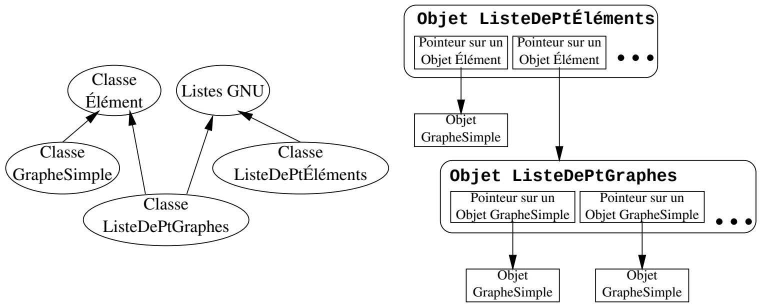


FIG. 8.4 - *Arbre d'héritage et représentation de listes de graphes simples*

L'avantage de la seconde solution est que toute méthode écrite pour une liste de graphes simples est aussi écrite pour une liste contenant à la fois des graphes simples et des listes de graphes. Cette particularité est essentiellement intéressante lorsque beaucoup de méthodes consistent à effectuer une même opération sur chacun des éléments de la liste et doivent être effectuées non seulement au niveau de la liste de graphes simples, mais aussi au niveau d'une liste de liste. Comme nous nous trouvons dans ce cas (méthode de dessin, de recherche, etc.), nous avons donc choisi la seconde méthode. Celle-ci a cependant un désavantage par rapport à la première. En effet, contrairement

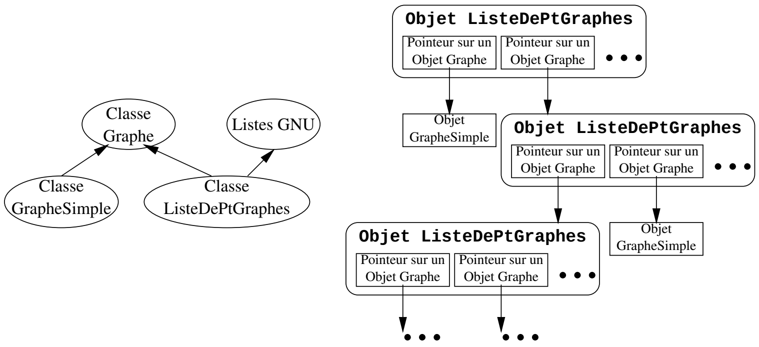


FIG. 8.5 - *Arbre d'héritage et représentation de listes de graphes récursives*

à la première solution, elle permet de faire un peu ce que l'on veut au niveau de l'imbrication des listes (imbrication virtuellement indéfinie) et au niveau de la constitution des listes : dans une liste dont les éléments ne devraient être que des graphes simples, rien n'empêche de mettre des listes de graphes. Cette seconde solution demande donc une plus grande rigueur de programmation.

8.3 Architecture

Du fait de la structure de données utilisée, aucun retour arrière n'est nécessaire au cours de l'analyse; l'ensemble des traitements effectués par l'analyseur morpho-syntagmatique peut donc se faire de manière séquentielle (architecture en série) et est organisé en 4 grandes tâches qui sont les suivantes :

```
-  Reconnaissance des "mots" et phrases
-  Regroupement en "segments maximaux sûrs"
-  Identification des propositions existantes
-  Identification des relations potentielles entre segments
```

L'identification des propositions nous permet surtout de réduire la portée des différents segments lors de l'identification des relations potentielles.

Chacune des étapes de l'analyse morpho-syntagmatique peut être vue comme un petit "module expert" créant un GSP en exécutant une tâche précise sur un GSP d'entrée et ce, à l'aide d'un certain nombre de connaissances spécifiques. À ces tâches principales, nous avons dû rajouter un certain nombre de tâches secondaires qui ont pour but de préparer le travail des différentes tâches principales.

L'analyseur morpho-syntagmatique recevant en entrée un texte représenté par une chaîne de caractères, il a été nécessaire d'ajouter une étape préliminaire à l'analyse pour mettre en place le GSP initial. Cette étape est réalisée par un module expert supplémentaire qui prend en entrée non pas un GSP mais la chaîne de caractères représentant le texte et la découpe en sous-chaînes en fonction de divers critères. Le GSP créé par cette étape préliminaire est un GSP de chaînes de caractères et sert d'entrée à l'étape suivante (recherche des unités morphologiques correspondant aux sous-chaînes ou à des groupes de sous-chaînes).

Le regroupement des unités morphologiques en segments maximaux sûrs ne devant s'effectuer que sur les unités morphologiques[48] d'une même phrase, celle-ci est précédée d'une étape consistant à découper le GSP d'unités morphologiques courant en sous-graphes représentant chacun une phrase et qui seront, à l'étape suivante, considérés comme des graphes distincts. Les étapes suivantes de l'analyse s'effectuent donc successivement sur un ensemble de GSP.

La reconnaissance des propositions existantes impliquant de travailler globalement sur chacun des chemins existant dans le GSP courant (un chemin représentant une analyse possible de la phrase), il nous fallait auparavant séparer tous les chemins et donc transformer le graphe existant en un arbre. Le nombre de chemins, toutes priorités cumulées confondues, pouvant être élevé, et le but de notre analyse étant d'obtenir les analyses les plus plausibles, nous avons choisi d'effectuer, avant la reconnaissance des propositions, l'extraction des chemins priorité cumulée par priorité cumulée : nous fournissons donc à l'analyseur des relations actancielles l'ensemble des analyses de priorité maximale (0) puis, sur sa demande, l'ensemble des analyses de priorités inférieures et ainsi de suite. En définitif, l'analyseur morpho-syntagmatique est donc composé de 7 modules (cf. figure 8.6 et table 8.3) qui s'enchaînent séquentiellement avec possibilité d'entrer dans le système soit au niveau de l'étape 1 (le plus courant) soit au niveau de l'étape 5 sur demande de l'analyseur des relations actancielles :

48. Nous entendons par unité morphologique, non seulement la chaîne de caractères mais aussi les informations de nature morpho-syntaxique (forme lemmatisée, catégorie syntaxique, attributs liés à la catégorie syntaxique, ...) qui lui sont liées.

```
1)  Découpage du Texte en ``mots''
2)  Analyse Morphologique du GSP de chaînes de caractères fourni
    avec appel éventuel au module de Traitement des Mots Inconnus
3)  Segmentation en Phrases
4)  Segmentation en Segments Maximaux Sûrs
5)  Extraction des meilleurs Chemins
6)  Segmentation de la Phrase en Propositions
7)  Établissement des Liens Potentiels
```

TAB. 8.3 - *Liste des différentes étapes de l'analyseur morpho-syntagmatique*

Les classes définies pour mettre en place cette architecture sont les suivantes (les graphes d'héritage sont donnés dans la figure 8.7) :

(1) classe `GrapheString` (représentant le graphe de chaînes de caractères correspondant à un texte et héritant de `GrapheSimple`) pour représenter le GSP et classe `UniteString` (représentant une sous-chaîne de caractères et héritant de `Unité`) pour les unités portées par les arcs.

(2) classe `GrapheMorph` (représentant le graphe des unités morphologiques correspondant à un texte et héritant de `GrapheSimple`) pour représenter le GSP et classe `UnitéMorph` (représentant les unités morphologiques détectées dans le texte et héritant de `Unité`) pour les unités portées par les arcs.

(3) classe `GrapheGroupes` (représentant la liste des phrases correspondant à un texte et héritant de `ListeDePtGraphes`) pour la liste de GSP, classe `GrapheGroupe` (représentant la liste des segments associés à une phrase et héritant de `GrapheSimple`) pour les GSP et `UnitéGroupe` (représentant un segment maximal sûr et héritant de `Unité`) pour les unités portées par les arcs.

(4) classe `GrapheChemins` (représentant la liste des chemins associés aux phrases du texte et héritant de `ListeDePtGraphes`) pour la liste de GSP, classe `GrapheChemin` (représentant la liste des différents chemins associés à une phrase et héritant de `GrapheSimple`) pour les GSP et `UnitéChemin` (représentant un segment maximal sûr d'un chemin et héritant de `Unité`) pour les unités portées par les arcs.

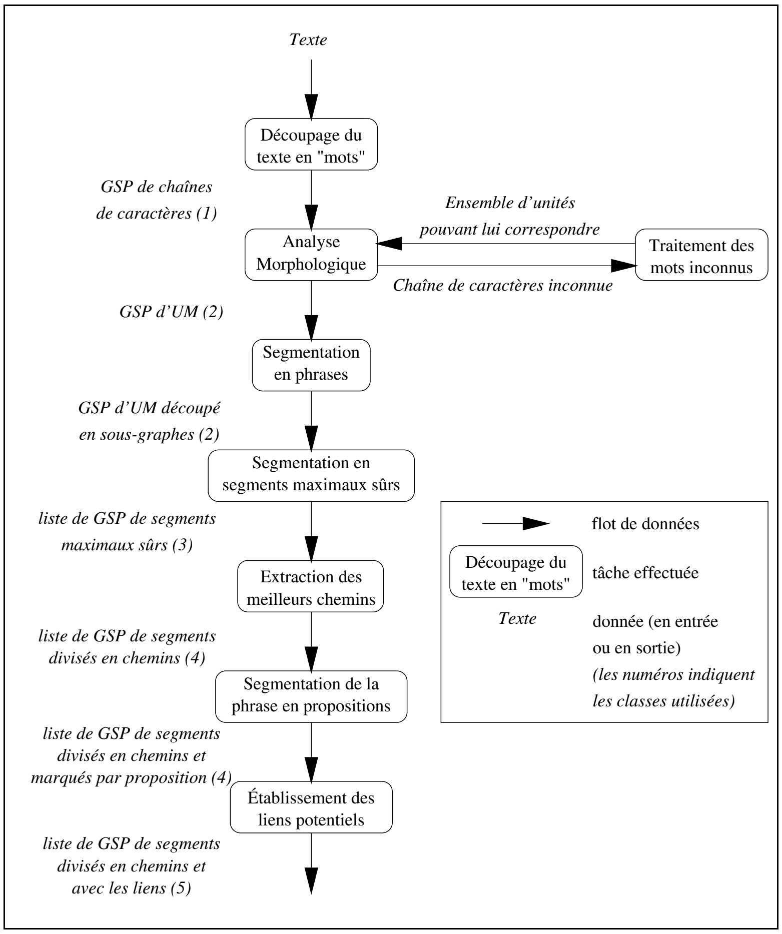


FIG. 8.6 - *Architecture de l'analyseur morpho-syntagmatique (les numéros font référence aux classes utilisées)*

(5) classe `GrapheCheminsLiens` (représentant la liste des chemins associés aux différentes phrases du texte et héritant de `ListeDePtGraphes`) pour la liste de GSP, classe `GrapheCheminLiens` (représentant la liste des chemins associés à une phrase et héritant de `GrapheSimple`) pour les GSP et `UnitéCheminLiens` (représentant un segment maximal sûr d'un chemin ainsi que les liens qu'il entretient et héritant de `Unité`) pour les unités portées par les arcs.

Avant de donner le détail des divers sous-modules de l'analyseur morphologique, nous allons, tout d'abord, donner quelques précisions sur les choix effectués pour mettre en place les diverses bases de connaissances utilisées.

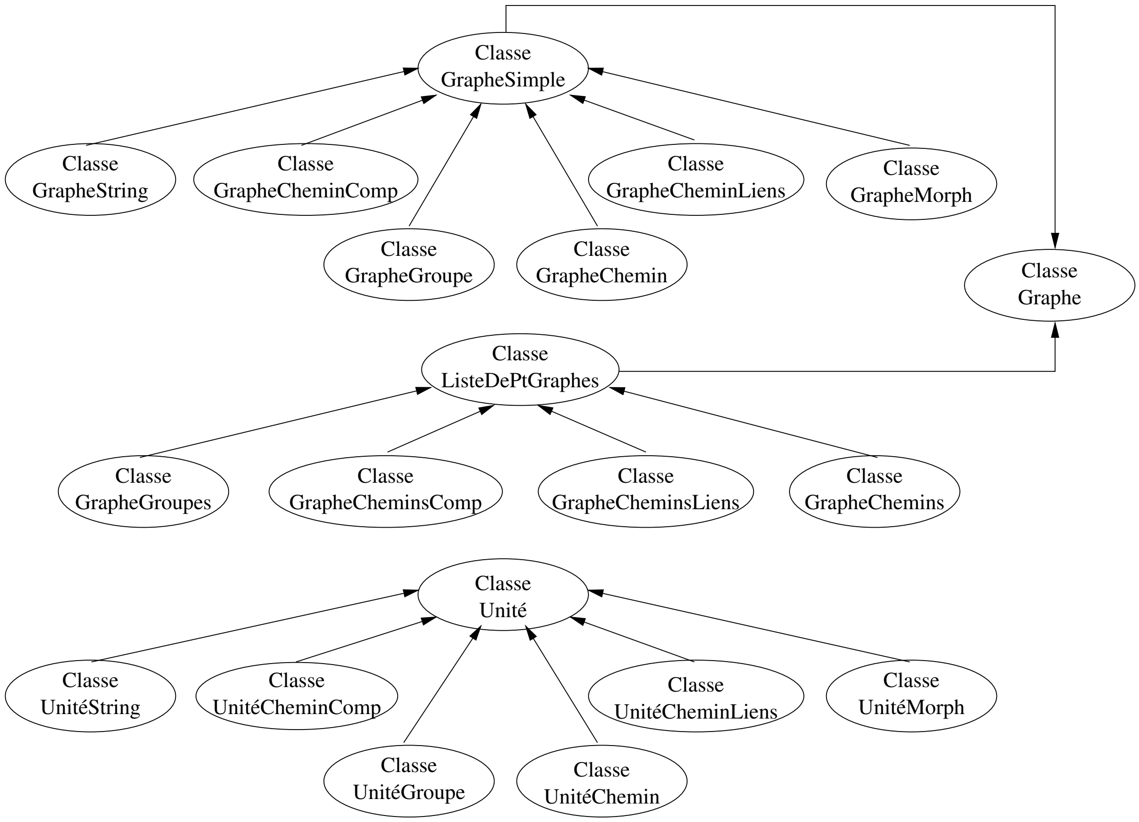


FIG. 8.7 - *Graphes d'héritage*

8.4 Bases de connaissances

Les divers sous-modules de l'analyseur morpho-syntagmatique ont chacun besoin d'un ensemble de connaissances spécifiques pour effectuer la tâche qui leur est assignée. Ces connaissances peuvent être soit sous forme de données (pour effectuer le découpage en unités morphologiques, par exemple, on utilise un dictionnaire), soit sous forme de règles. Ces diverses connaissances peuvent être formalisées de deux manières différentes : soit de façon déclarative, soit de façon procédurale.

L'avantage d'une méthode procédurale par rapport à une méthode déclarative se situe en fait dans la rapidité d'exécution du programme (due au fait qu'il n'y a aucune "interprétation" des données à effectuer pour obtenir les connaissances) et dans l'apparente facilité de mise en place. Cependant, les modifications et les ajouts dans une telle base de connaissances nécessitent des connaissances informatiques générales et liées à la conception même du programme rendant ceux-ci difficiles pour toute personne qui n'a pas participé à la conception (la complexité aidant, il peut arriver que même les concepteurs soient incapables d'effectuer toute modification de la base). Les bases de connaissances que nous utilisons étant incomplètes et étant amenées à être complétées par des linguistes (pas forcément informaticiens) il nous a donc semblé préférable d'implémenter les connaissances nécessaires aux diverses étapes de façon déclarative.

Les connaissances utilisées lors de la mise en place des diverses étapes de l'analyse sont les suivantes : le Dictionnaire des Formes Fléchies (utilisé pour déterminer les unités morphologiques présentes), la Grammaire de Segmentation en Segments Sûrs (utilisée pour effectuer le regroupement des unités morphologiques en segments maximaux sûrs), la Grammaire de Découpage en Propositions (utilisée pour permettre l'identification des propositions existantes) et enfin la Grammaire des Liens Potentiels (utilisée pour permettre l'identification des relations potentielles entre segments). Les grammaires mentionnées ci-dessus sont composées d'ensembles de règles adaptées au problème spécifique à traiter et qui seront détaillées lors des explications relatives à chacune des étapes de l'analyseur morpho-syntagmatique.

Le dictionnaire des formes fléchies contient les diverses formes fléchies des lemmes contenus dans l'un des deux dictionnaires de lemmes que nous utilisons : le Dictionnaire des Unités Grammaticales et le Dictionnaire des Unités Lexicales. Le premier de ces dictionnaires contient l'ensemble des lemmes appartenant à des catégories syntaxiques figées (préposition, clitique, auxiliaire, etc.), c'est-à-dire l'ensemble des mots dont le rôle syntaxique est particulièrement important. Le second, quant à lui, contient l'ensemble des lemmes appartenant à des catégories syntaxiques ouvertes (nom, adjectif, verbe, etc.). Ces éléments sont donc des bases de connaissances volumineuses et évoluant de façon permanente. Pour cette raison, il nous a semblé judicieux de les spécifier hors du programme et donc de les fournir à l'analyseur de manière déclarative.

Dans cette première version de l'analyseur, les dictionnaires ainsi que la plupart de ces grammaires sont très peu développés par rapport à ce qu'ils devraient être mais l'ensemble du processus de traitement est mis en place. Rappelons, enfin, que l'enrichissement des connaissances pourra être effectué hors programme car nous avons opté pour une implémentation déclarative des connaissances utilisées.

Aux connaissances décrites ci-dessus, il nous a fallu ajouter, du fait de l'implémentation choisie, un certain nombre de connaissances supplémentaires (données sous forme déclarative par l'intermédiaire de listes de valeurs possibles; cf. Annexe B.3) pour permettre la vérification de la cohérence des divers fichiers.

8.5 Détail des différents sous-modules

Nous allons maintenant détailler les divers sous-modules de l'analyseur morpho-syntagmatique à l'exception des modules de segmentation en phrases et d'extraction des chemins qui sont extrêmement simples (étapes 3 et 5 de l'analyseur morpho-syntagmatique, cf. table 8.3, page 132).

La segmentation en phrases s'effectue en recherchant les fins de phrases et ce, en se basant d'une part sur l'ensemble des caractères indiquant une fin de phrase, d'autre part sur un ensemble de règles sur les fins de phrase. Dans un GSP, pour un nœud donné, les arcs étant ordonnés par ordre de priorité croissant, l'extraction des chemins consiste uniquement en un parcours en profondeur du graphe qui s'arrête dès que le cumul des ordres de priorité du chemin courant est supérieur à celui du premier chemin extrait.

8.5.1 Découpage préliminaire du texte

Le découpage préliminaire du texte en sous-chaînes de caractères a pour but de permettre ensuite une recherche des unités morphologiques correspondantes dans le dictionnaire des formes fléchies. Ce découpage s'effectue donc en fonction de la définition choisie pour l'unité morphologique. Or, comme l'ont expliqué par exemple [Fuchs *et al.*, 1993], il n'existe aucune définition idéale de l'unité morphologique mais la plus réaliste, dans le cadre d'une utilisation en traitement automatique du langage, est de rester proche de la notion intuitive de “mot”. Cependant ce n'est pas si facile à définir formellement. En particulier, la définition à laquelle on pourrait penser (toute chaîne de caractères se trouvant entre deux blancs ou entre un blanc et une ponctuation) n'est pas bonne. Par exemple, *l'automobile* doit faire deux unités tandis que *aujourd'hui* n'en fait qu'une seule. De même *semble-t-il* est composé de deux unités et non de trois, alors que *semi-remorque* n'est composé que d'une seule unité tout comme *passage à niveau*. Dans notre analyseur, une unité morphologique peut donc être représentée par une ou plusieurs sous-chaînes de caractères (*aujourd'hui* sera représenté par trois sous-chaînes : *aujourd*, *'* et *hui*). Cette conséquence du découpage doit être prise en compte au niveau du dictionnaire : l'entrée correspondant à l'unité morphologique représentant *aujourd'hui* doit être composée des trois sous-chaînes extraites.

8.5.2 Analyse morphologique

L'analyse morphologique prend en entrée un GSP de chaînes de caractères et rend en sortie un GSP d'unités morphologiques. L'extraction des unités morphologiques est basée sur l'utilisation du dictionnaire des formes fléchies.

Le Dictionnaire des Formes Fléchies

Comme nous nous trouvons actuellement dans une phase de développement et de test, le dictionnaire des formes fléchies ne contient que très peu de mots (de l'ordre de la centaine). Cependant, nous avons déjà défini une partie du format de ce dictionnaire et c'est ce dernier que nous allons maintenant présenter.

Comme nous venons de le voir, chacune des entrées de ce dictionnaire correspond à une ou plusieurs chaînes de caractères que l'on peut trouver dans le GSP résultant du découpage préliminaire. Pour chacune de ses entrées, il contient un certain nombre de données : le lemme auquel correspond l'entrée (si celui-ci est différent de l'entrée), une éventuelle priorité (qui permet de marquer les utilisations peu fréquentes d'un mot), le nom du dictionnaire auquel appartient son lemme (DUG - dictionnaire des unités grammaticales - ou DUL - dictionnaire des unités lexicales), sa catégorie morphologique ainsi qu'un certain nombre d'attributs propres à la catégorie en question. L'ordre de priorité donné dans le dictionnaire des formes fléchies correspond à l'ordre de priorité initial associé à l'unité créée, il n'est obligatoire que dans le cas où il est différent de 0 (par défaut, les ordres de priorité initiaux sont à 0). Le dictionnaire des formes fléchies, pour pouvoir traiter l'exemple donné précédemment (*l'automobile*), et ce, en considérant que le découpage effectué a extrait trois chaînes de caractères (*l*, *'* et *automobile*), aura besoin d'avoir deux entrées *l '* (entrée composée de deux sous-chaînes de caractères : *l* et *'*) et *automobile* dont les descriptions sont les suivantes[49] :

`entrée : l '`	`entrée : automobile`
`lemme = le`	`lemme = automobile`
`catégorie = déterminant`	`catégorie = nom`
`nombre = singulier`	`nombre = singulier`
`genre = féminin ou masculin`	`genre = féminin`
`source = DUG`	`source = DUL`

49. Les entrées données dans le tableau correspondent uniquement à celles utilisées pour analyser *l'automobile* en tant que segment nominal. Il existe aussi dans le dictionnaire des formes fléchies une entrée correspondant à *l'* de catégorie pronom mais qui n'est pas indiquée ici.

Comme pour la deuxième entrée le lemme a la même valeur que l'entrée, il n'est pas nécessaire de le préciser. De même, pour la première entrée, le genre associé au déterminant étant indéterminé (il peut prendre les deux valeurs) il n'est pas nécessaire de le préciser. En effet, afin de simplifier les données associées à de telles entrées, nous avons choisi de ne pas donner les valeurs des attributs dans le cas où ces derniers pourraient prendre toutes les valeurs possibles. Ces valeurs devant tout de même être connues du programme, nous avons dû ajouter des connaissances à l'analyseur morphologique : la liste des catégories (liste qui donne, pour chaque catégorie l'ensemble des attributs qui lui correspondent - cf. Annexe B.3) ainsi que la liste des attributs (liste qui donne, pour chaque attribut, l'ensemble des valeurs qu'il peut prendre - cf. Annexe B.3). Ces deux listes permettent de définir pour chaque catégorie "un schéma de la catégorie" et de compléter les valeurs manquantes. Ces deux listes sont aussi utilisées pour effectuer la vérification de la cohérence du dictionnaire des formes fléchies lors de sa lecture ainsi que pour gérer le problème des mots inconnus comme nous le verrons plus bas. La description dans le dictionnaire des formes fléchies des deux entrées données ci-dessus se fait donc de la manière suivante :

```
{ l ' : {lemme = le, cat = det, nbr = sing, source = DUG }}
{ automobile : {cat = nom, genre = fem, nbr = sing, source = DUL }}
```

TAB. 8.4 - *Entrées dans le Dictionnaire des Formes Fléchies*

Un dictionnaire étant forcément incomplet, il est tout à fait possible d'avoir des chaînes de caractères issues du texte et inexistantes dans le dictionnaire des formes fléchies. Pour traiter ce type de problème, nous avons mis en place un module de traitement des mots inconnus.

Traitement des mots inconnus

Le module de traitement des mots inconnus prend en entrée une chaîne de caractères et renvoie au module d'analyse morphologique une ou plusieurs unités morphologiques pouvant correspondre à cette chaîne. Pour effectuer ce traitement, nous avons choisi de générer autant d'unités morphologiques qu'il existe de catégories morphologiques et de signaler à l'utilisateur qu'il y a un mot inconnu. Cette solution n'est bien sûr pas définitive car elle est trop simpliste pour donner de bons résultats et elle génère trop d'unités morphologiques.

Nous pensons cependant que la solution de générer autant d'unités morphologiques qu'il y a de possibilités est une solution qui, du fait de la structure de données que nous avons choisie, est viable. Il reste cependant à limiter les possibilités. Celles-ci pourraient être limitées, d'une part aux catégories ouvertes (nom, adjectif, verbe, etc.), d'autre part en utilisant les travaux de J. Vergne [Vergne, 1990] afin de déterminer les catégories possibles pour un mot en fonction de sa terminaison (par exemple, un mot terminant en *tion* est probablement un nom de genre féminin).

Comme nous pouvons le voir, le fonctionnement de ce module est actuellement très restreint mais il nous a semblé nécessaire d'inclure une telle étape dans le fonctionnement du module d'analyse morpho-syntagmatique de façon à pouvoir le rendre robuste.

8.5.3 Segmentation en segments maximaux sûrs

Le texte ayant été découpé en phrases, la segmentation en segments maximaux sûrs s'effectue successivement sur chacune des phrases du texte. Celle-ci consiste à construire un GSP de segments maximaux sûrs à partir d'un GSP d'unités morphologiques. Il faut donc regrouper un certain nombre d'unités morphologiques contiguës dans un même segment. Pour effectuer cette opération nous utilisons une liste de règles de construction des segments maximaux (règles regroupées dans la grammaire de segmentation en segments sûrs). Dans cette grammaire, quand une catégorie de segments (par exemple, les segments nominaux) admet plusieurs constructions, nous lui associons autant de règles qu'il existe de constructions possibles.

La grammaire utilisée

Les règles de construction utilisées sont composées d'au plus cinq parties qui sont toutes facultatives exceptées les parties *identification* et *construction* et dont la signification est la suivante :

1. Une partie *identification* qui contient le nom du segment à créer éventuellement précédé du nom de la règle utilisée (s'il n'y a pas de nom de règle alors celui-ci est identique au nom du segment). Nous avons utilisé deux noms pour permettre, en fin d'analyse, de connaître les règles utilisées afin de faciliter une vérification de l'analyse effectuée. Cela ne pouvait se faire uniquement par le nom des segments étant donné qu'il peut exister plusieurs règles pour générer une même catégorie de segment.

2. Une partie *priorité* qui, comme dans le cas du dictionnaire des formes fléchies, permet de spécifier, pour la règle en question, un ordre de priorité différent de 0. Cette priorité permet de mettre en place un ordre de priorité entre les différentes constructions possibles pour une même catégorie de segment.

3. Une partie *construction* qui contient l'ensemble des composants du segment. Dans cette partie, on précise pour chacun des composants, son caractère obligatoire ou facultatif, sa catégorie (une des catégories morphologiques) ainsi que les contraintes qu'il doit respecter. Les contraintes imposées aux composants permettent de limiter les différentes possibilités tout en restreignant les valeurs possibles d'un attribut.
   Une construction de la forme de celle du tableau 8.5 permet de définir un segment verbal composé d'un verbe conjugué et de ses clitiques. Les numéros permettent de nommer les composants de la construction (on verra leur utilité lors des explications relatives aux autres parties de la construction), le caractère `?` placé derrière un numéro de composant permet d'indiquer qu'à ce composant peut être associée au plus une unité morphologique (dans notre exemple, les clitiques sont donc facultatifs contrairement au verbe); le caractère `*`, à la même place, permet de spécifier qu'à ce composant peuvent être associées autant d'unités morphologiques que l'on veut. Ensuite, est indiquée la catégorie morphologique (clitique ou v_conj - pour "verbe conjugué") que doivent respecter les unités morphologiques pour pouvoir être un composant possible. La dernière partie de chaque définition de composant correspond aux contraintes portant sur les unités morphologiques : `(cas = accusatif)` indique que l'on recherche une unité morphologique dont l'attribut "cas" a pour valeur possible "accusatif". Cette contrainte permet aussi de restreindre la valeur du cas de l'unité morphologique à "accusatif". Une construction comme celle du tableau 8.5 accepte des segments verbaux comme *je le lui donne*, *je le donne*, *je donne* ainsi que ceux composés uniquement d'un verbe (la règle complète est donnée en Annexe B.3, page XXIII).

```
0 ?  clitique  (cas = nominatif)
1 ?  clitique  (cas = accusatif)
2 ?  clitique  (cas = datif)
3    v_conj
```

TAB. 8.5 - *Partie construction d'une règle de construction de segment*

4. Une partie *contraintes globales* qui permet de préciser les contraintes que doivent respecter l'ensemble des éléments composant le segment. Ces contraintes précisent, en général, des relations entre les différents attributs des éléments du segment. Ces contraintes, comme celles portant sur un élément, permettent d'imposer un certain nombre de limitations sur les valeurs d'attributs des différents éléments.
   Les contraintes du tableau 8.6 permet de définir les accords en personne et en nombre de la construction du tableau 8.5. On remarque, ici, l'utilité d'avoir associé, au niveau de la construction des numéros aux composants.

{ `0.pers = 3.pers` } { `0.nbr = 3.nbr` }

TAB. 8.6 - *Contraintes globales d'une règle de construction de segment*

5. Une partie *affectations* qui permet de préciser la valeur des attributs du segment construit en fonction de celles des attributs des éléments.
   Les affectations du tableau 8.7 permettent de définir la valeur des attributs "genre" et "nombre" d'un segment nominal composé d'un déterminant (composant `0`), d'adjectifs (composant `1`) et d'un nom (composant `2`).

{ `nbr = 2.pers` } { `genre = 2.genre` }

TAB. 8.7 - *Affectations dans une règle de construction de segment*

Nous allons maintenant, sur un exemple complet (un ensemble de règles permettant de construire un segment nominal), donner un aperçu du contenu des différentes parties.

Des travaux que nous avons réalisés [Picand, 1993] il ressort un certain nombre de constructions possibles pour un segment nominal dont les suivantes[50] :

(1) la partie d'un syntagme nominal classique comprise entre le déterminant et le nom (comme par exemple, *la petite voiture* ou *la voiture* mais pas *la voiture verte*).
(2) un syntagme prépositionnel à complément nominal (ex : *de la voiture*).
(3) un syntagme nominal sans déterminant comme : *voiture* ou *verte campagne* (construction moins fréquente que les précédentes).

50. Ces règles ne sont pas les seules qui permettent de construire un segment nominale mais sont, pour l'instant, les seules qui sont sont fournies à l'analyseur morpho-syntagmatique.

Ces trois constructions possibles nous donnent les règles suivantes :

1. *identification* : nom règle = SN, nom segment = SN.
   construction :

   le premier élément est obligatoire et de catégorie "déterminant";

   le second élément est facultatif et peut regrouper plusieurs unités morphologiques de catégorie "adjectif";

   le troisième élément est obligatoire et de catégorie "nom".

   contraintes globales : les trois éléments doivent être égaux en genre et en nombre.
   affectations : le genre et le nombre du segment sont égaux à ceux du nom.

2. *identification* : nom règle = SN1, nom segment = SN.
   construction :

   le premier élément est obligatoire et de catégorie "préposition", celle-ci devant permettre d'introduire un nom;

   le second élément est facultatif et de catégorie "déterminant";

   le troisième élément est facultatif et peut regrouper plusieurs unités morphologiques de catégorie "adjectif";

   le quatrième élément est obligatoire et de catégorie "nom".

   contraintes globales : les trois derniers éléments doivent être égaux en genre et en nombre.
   affectations : le genre et le nombre du segment sont égaux à ceux du nom.

3. *identification* : nom règle = SN2, nom segment = SN.
   ordre de priorité : 1.
   construction :

   le premier élément est facultatif et peut regrouper plusieurs unités morphologiques de catégorie "adjectif";

   le second élément est obligatoire et de catégorie "nom".

   contraintes globales : les deux éléments doivent être égaux en genre et en nombre.
   affectations : le genre et le nombre du segment sont égaux à ceux du nom.

Ces règles sont données dans la grammaire de construction des segments maximaux sûrs sous la forme suivante (* indique qu'il faut 0 ou plusieurs unités morphologiques de ce type et **?** signifie qu'il faut au plus une unité morphologique de ce type) :

{SN : construction = [0 {det}; 1∗ {adj}; 2 {nom}], contraintes = [{0.genre = 1.genre = 2.genre}; {0.nbr = 1.nbr = 2.nbr}], affectations = [{nbr = 2.nbr}; {genre = 2.genre}]}
{SN1 SN : construction = [0 {prep (introduit = nom)}; 1? {det}; 2∗ {adj}; 3 {nom}], contraintes = [{1.genre = 2.genre = 3.genre}; {1.nbr = 2.nbr = 3.nbr}], affectations = [{nbr = 3.nbr}; {genre = 3.genre}]}
{SN2 SN : priorité = 1, construction = [0∗ {adj}; 1 {nom}], contraintes = [{0.genre = 1.genre}; {0.nbr = 1.nbr}], affectations = [{ nbr = 1.nbr}; {genre = 1.genre}]}

TAB. 8.8 - *Règles de construction d'un Segment Nominal*

Reprenons l'exemple de *l'automobile.* L'unité morphologique associée à la chaîne de caractères *l '* et celle associée à *automobile* seront regroupées dans un même segment grâce à la règle 1. Le fait de les regrouper ainsi, et ce à cause des contraintes globales de la règle, entraîne que l'attribut "genre" de l'unité morphologique associée à *l '* sera modifié : il avait, au départ, deux valeurs possibles (masculin et féminin) mais comme *automobile* est de genre féminin, l'attribut aura, après application de la règle, la même valeur que dans l'unité associée à *automobile*, c'est-à-dire féminin. Les attributs associés au segment correspondant à *l'automobile* seront le genre (féminin) et le nombre (singulier).

Ce module, malgré l'ensemble du travail effectué, n'est cependant pas complet. En effet, nous n'avons pas encore défini tous les segments nécessaires à l'étude de l'ensemble des textes du corpus et surtout, nous n'avons pas encore déterminé comment nous traiterons le problème de la coordination. Le travail, cependant, est relativement bien avancé et suffit pour traiter un grand nombre des phrases du corpus et valider notre approche de l'analyse morphologique.

La grammaire utilisée par ce module étant une grammaire d'unification, le fonctionnement de ce module est basé sur une unification effectuée entre les règles de la grammaire et le graphe des unités morphologiques. Avant de donner l'algorithme utilisé par ce module, nous donnons quelques indications sur l'unification et les grammaires d'unification.

L'unification

L'unification est une opération qui consiste à fusionner deux ensembles structurés d'informations. Le résultat de cette opération est, soit un ensemble structuré d'informations correspondant à la fusion des deux ensembles unifiés, soit un échec si ces deux ensembles contiennent des informations contradictoires ou incompatibles.

L'intégration de l'unification au sein des formalismes syntaxiques a permis d'enrichir la notion de catégorie syntaxique grâce à l'introduction de *structures de traits*. Un trait s'apparente à un attribut pouvant stocker une ou plusieurs informations linguistiques et permettant de stocker une caractéristique de l'élément linguistique. Les structures de traits permettent de stocker, selon une organisation hiérarchique, les différentes caractéristiques associées à un élément linguistique. Un trait peut contenir une valeur atomique (symbole, nombre, ...) ou une structure de traits (cf. figure 8.8 au niveau du trait `0` de la règle). Les grammaires d'unification utilisent à la fois la structure de traits et le mécanisme d'unification.

Les grammaires d'unification

Dans une grammaire d'unification, un constituant est caractérisé par un ensemble de traits et peut être représenté par une structure de traits. Une structure de traits peut représenter soit un "mot", soit une règle de construction d'un segment (cf. figure 8.8).

L'opération d'unification permet de fusionner plusieurs traits distincts de façon à ce que leur valeur soit commune. Le succès de l'unification de deux traits repose sur la compatibilité de leurs valeurs respectives : deux valeurs atomiques ne peuvent s'unifier que si elles ont le même type et la même valeur, un trait sans valeur est compatible avec tous les types de valeur et, enfin, si deux valeurs à unifier sont des structures de traits, le résultat de l'unification est une structure de traits correspondant à l'union des deux structures unifiées. Soient les objets $\mathtt{O}_1$ et $\mathtt{O}_2$ de la figure 8.8 (représentations partielles des éléments lexicaux *l'* et *automobile*). L'unification de ces deux éléments avec la règle représentée par l'objet $\mathtt{O}_3$ revient à appliquer cette règle en construisant un objet $\mathtt{O}_4$ comme celui donné sur cette même figure. L'unification a été possible, dans ce cas, parce qu'un certain nombre de conditions exprimées dans $\mathtt{O}_3$ (sur les catégories de 0 et 1, sur les accords de genre et de nombre) étaient réalisées. De plus, l'unification a permis de compléter des informations manquantes (le genre de *l'*, qui était indéterminé, a pris la valeur féminin dans $\mathtt{O}_4$).

$$\mathtt{O}_1 = \begin{bmatrix} lex = l' \\ cat = det \\ nombre = sing \end{bmatrix}$$

$$\mathtt{O}_2 = \begin{bmatrix} lex = automobile \\ cat = nom \\ nombre = sing \\ genre = fem \end{bmatrix}$$

$$\mathtt{O}_3 = \begin{bmatrix} cat = SN \\ forme = (0\ 1) \\ 0 = [cat = det] \\ 1 = [cat = nom] \\ genre = (0, genre) = (1, genre) \\ nbr = (0, nbr) = (1, nbr) \end{bmatrix}$$

$$\mathtt{O}_4 = \begin{bmatrix} lex = l'automobile \\ cat = SN \\ forme = (0\ 1) \\ 0 = \begin{bmatrix} lex = l' \\ cat = det \\ nombre = sing \\ genre = fem \end{bmatrix} \\ 1 = \begin{bmatrix} lex = automobile \\ cat = nom \\ nombre = sing \\ genre = fem \end{bmatrix} \\ genre = fem \\ nbr = sing \end{bmatrix}$$

FIG. 8.8 - *Représentation de l'unification sous forme de matrices de traits*

Implémentation de l'unification

Pour pouvoir utiliser les grammaires d'unification, il faut donc disposer d'un mécanisme d'unification. Il existe des langages tel PROLOG qui disposent, en standard, d'un tel mécanisme. Cependant, comme nous avons choisi d'utiliser le `C++` comme langage d'implémentation, et étant donné que les "PROLOG objets" ne sont pas aussi performants que le `C++` au niveau de la programmation objet, nous avons choisi, non pas de changer de langage, mais d'implémenter un mécanisme d'unification en `C++`.

Comme nous avons pu le voir précédemment, l'unification est basée sur la comparaison entre valeurs de traits. Quand les traits sont atomiques, il suffit d'effectuer un test d'égalité mais quand ils sont complexes (quand ils contiennent d'autres traits), il faut effectuer une comparaison de structures. La comparaison de structures pouvant se ramener à un ensemble de comparaison de traits atomiques, il suffit, pour implémenter un mécanisme d'unification, de disposer d'un mécanisme de comparaison de traits atomiques. Pour que l'unification soit efficace, cette comparaison de traits atomiques doit être le plus rapide possible.

Nous avons donc choisi d'implémenter les traits sous la forme d'entiers correspondant à des ensemble de bits et utilisons, pour effectuer la comparaison, les opérateurs bit à bit du `C++`. Effectuer l'unification de deux traits atomiques consiste alors à comparer leur nom (il est incohérent d'unifier un trait `genre` avec un trait `nombre`) et à calculer $\mathtt{V}_1$ `&` $\mathtt{V}_2$: si le résultat de cette opération est 0 alors l'unification est impossible sinon le résultat de cette unification est celui de l'opération effectuée. Pour pouvoir effectuer ainsi l'unification, un trait "vide" doit être représenté par l'ensemble de ses valeurs possibles. Effectuer l'unification de deux listes de traits consiste à effectuer l'unification de chacun des traits de la liste $\mathtt{L}_1$ avec chacun des traits de la liste $\mathtt{L}_2$.

Pour mettre en place cette unification, nous avons donc défini une classe `Trait` composée de deux attributs (une chaîne de caractères représentant son nom et un pointeur sur un objet de la classe `Valeur` qui contient la valeur du trait) ainsi qu'une classe `ListeTraits` qui représente les listes de traits. La classe `Valeur` est une classe abstraite ayant deux filles : la classe `ValeurInt` qui représente les traits à valeur entière et la classe `ValeurString` qui représente les traits dont la valeur est une chaîne de caractères (nous verrons l'utilité de cette dernière dans le paragraphe suivant). La principale méthode définie sur cet ensemble de classes est la méthode `Unification` qui, dans le cas de la classe `Trait`, prend un paramètre de type `Trait` et rend un pointeur sur un nouvel objet de la classe `Trait` (celui-ci est égal à 0 en cas d'échec). Cette méthode fait appel à la méthode `Unification` de la classe `Valeur` (méthode surchargée par chacune des classes filles) quand les deux noms de traits sont égaux. Dans le cas de la classe `ListeTraits`, nous avons défini deux méthodes `Unification` : une qui prend en paramètre un objet de la classe `ListeTraits` et qui rend un pointeur sur un objet de la classe `ListeTraits`, une autre qui prend en paramètre un objet de la classe `Trait` et qui rend un pointeur sur un objet de la classe `Trait`; la seconde méthode étant utilisée par la première.

Dans le cadre de ces définitions, une unité morphologique est donc composée d'une chaîne de caractères (la chaîne correspondant à l'entrée du dictionnaire) et d'un objet de la classe `ListeTraits` représentant sa structure de traits. Pour les unités morphologiques, un des traits a un statut un peu spécial : il s'agit du trait `lemme` qui donne le nom de l'entrée lexicale à laquelle correspond la chaîne de caractères (la chaîne *la* de catégorie "clitique" correspond à l'entrée lexicale "le"). Du fait de la variété de valeurs que peut prendre un tel trait, il n'est pas possible de le représenter par une valeur entière. C'est pour représenter ce style de trait que nous avons donc défini la classe `ValeurString` (la structure de données utilisée pour définir les différents traits est représentée sur la figure 8.9).

Dans une règle de la grammaire de construction des segments, une construction

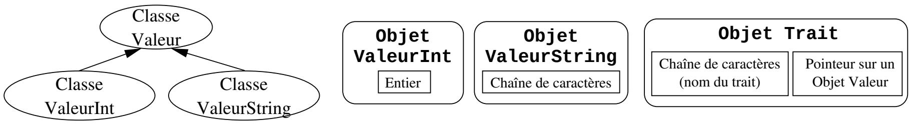


FIG. 8.9 - *Représentation des traits*

correspond à une liste de composants dont la catégorie et les traits mentionnés dans la partie "contraintes" portant sur les unités morphologiques doivent être unifiés avec chacune des unités morphologiques détectées dans la phrase pour, d'une part construire une nouvelle unité morphologique (unité contenant les modifications éventuellement apportées par l'unification), d'autre part regrouper les nouvelles unités construites en un segment. Les contraintes globales et les affectations correspondent à un ensemble d'unifications à effectuer sur des traits atomiques. Une règle de la grammaire de construction des segments est donc composée d'une catégorie, d'une liste de composants représentés par des objets de la même classe que les unités morphologiques dans lesquels la valeur des traits tient compte des contraintes portant sur ces unités, de deux listes de listes de traits correspondant aux contraintes globales à vérifier (chaque contrainte est représenter par une liste de traits indiquant les différents traits à instancier) et aux affectations à effectuer. Afin de clarifier cette structure de données, nous donnons dans la figure 8.10 (la valeur des traits sera indiquée, pour faciliter la compréhension, sous forme de chaîne de caractères et non d'entier), la représentation de la première règle de construction d'un segment nominal de la table 8.8, page 143.

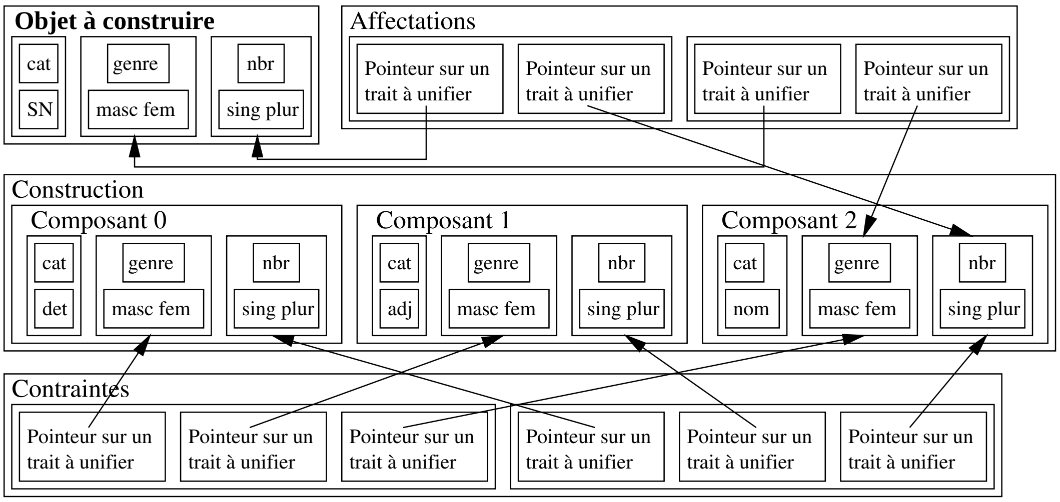


FIG. 8.10 - *Représentation d'une règle de construction d'un segment nominal*

L'algorithme de découpage en segments maximaux sûrs

Le fonctionnement de ce sous-module consiste à unifier l'ensemble des règles de la grammaire de segmentation en segments maximaux sûrs avec un graphe d'unités morphologiques. Dans le cas de ce sous-module il existe cependant une contrainte supplémentaire : pour pouvoir prendre en compte l'application d'une règle il faut être capable de trouver un ensemble de regroupements permettant d'atteindre le dernier nœud du graphe car toutes les unités morphologiques doivent être regroupées en segments pour que le découpage soit correct. Le principe d'unification des règles consiste donc à unifier l'ensemble des règles avec le graphe à partir du premier nœud, puis à partir du nœud d'arrivée de toute règle ayant abouti, de continuer ainsi de suite et de construire le graphe résultat uniquement en dépilant les appels. Pour cela, il suffit donc de savoir unifier un ensemble de règles avec un graphe à partir d'un nœud donné, ce qui peut être fait par l'algorithme suivant :

```
- Si le noeud courant est le dernier noeud ou Si une recherche a
  déjà été effectuée à partir de ce noeud Alors
  | - retourner 1
- ok ← 0
- Pour chacune des règles de la liste Faire
  | - Unifier la règle courante avec les unités issues du noeud N
  | - Si l'unification a réussi Alors
  |   | - déterminer N1 le noeud final de l'unification
  |   | - Si l'ensemble des règles peut s'unifier avec le reste
  |   |   du graphe (ce même algorithme exécuté en partant de N1
  |   |   retourne la valeur 1) Alors
  |   |   | - insérer le groupe créé dans le nouveau GSP
  |   |   | - ok = 1
- Retourner la valeur de ok
```

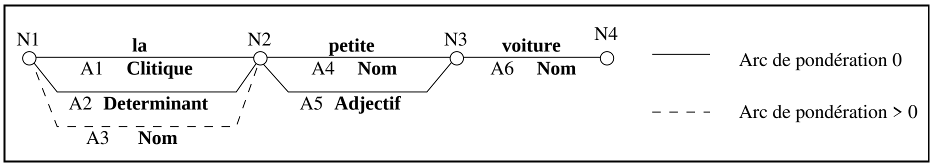


FIG. 8.11 - *Graphe d'unités morphologiques*

L'application de cet algorithme sur le graphe de la figure 8.11 s'effectue de la manière suivante (le résultat est donné sur la figure 8.12) :

Étape 1 : recherche des règles applicables à partir de N1
- → Règle unifiable : SN ; Arcs sélectionnés : (A2 A4) et (A2 A5 A6)
- Nœuds d'arrivée : N3 et N4
- Étape 2 : recherche des règles applicables à partir de N3
  - → Règle unifiable : SN1 ; Arc sélectionné : (A6)
  - Nœud d'arrivée : N4
  - Étape 3 : recherche des règles applicables à partir de N4
    - Dernier nœud du graphe ⇒ le groupe (A6) est validé
  - Insertion du groupe (A6) et validation du groupe (A2 A4)
  - Pas d'autre règle unifiable à partir de N3
- Insertion du groupe (A2 A4) et validation d'un chemin
- Étape 4 : recherche des règles applicables à partir de N4
  - Dernier nœud du graphe ⇒ validation de (A2 A5 A6)
- Création du groupe (A2 A5 A6) et validation d'un chemin
- → Règle unifiable : SN1 ; Arcs sélectionnés : (A2)
- Nœud d'arrivée : N2
- Étape 5 : recherche des règles applicables à partir de N2
  - → Règle unifiable : SN1 ; Arc sélectionné : (A4)
  - Nœud d'arrivée : N3
  - Étape 6 : recherche des règles applicables à partir de N3
    - Nœud déjà parcouru ⇒ validation de (A4)
  - Insertion du groupe (A4) et validation du groupe (A2)
- Insertion du groupe (A6) et validation d'un chemin
- Pas d'autre règle unifiable à partir de N1

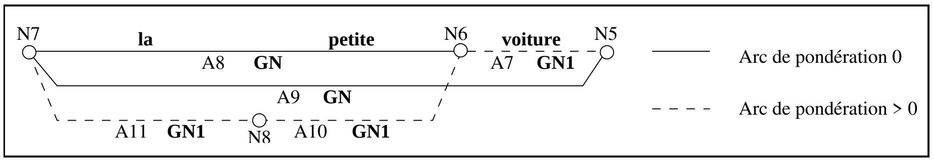


FIG. 8.12 - *Résultat du découpage en segments*

8.5.4 Segmentation de la phrase en propositions

Nous rappelons que le fonctionnement de ce module est basé sur un système de règles et que ce module a pour but de déterminer, pour chaque proposition, son noyau propositionnel ainsi que la liste des segments dont le rattachement à la proposition est possible mais incertain. Les règles utilisées permettent de préciser les bornes entre lesquelles nous sommes sûrs de ne pas avoir d'ambiguïté sur le rattachement des segments. Chacune de ces règles contient donc la catégorie des segments indiquant un début de proposition ainsi que celle des segments indiquant une fin de proposition. Par exemple, si on indique, sous forme de règle, que le début d'une proposition est marqué par la présence d'un segment de catégorie SQ (segment commençant par que, qui, etc.) et que la fin d'une proposition est marquée par la présence d'un segment de catégorie SV_conj (groupe verbal contenant un verbe conjugué) alors tous les segments compris entre un SQ et un SV_conj ainsi que ces deux segments feront partie des "segments sûrs" d'une proposition (et donc du noyau propositionnel de cette dernière), ceux qui suivent le verbe conjugué pourront faire partie des "segments possibles" de la proposition. A partir de la phrase *J'ai su que vous étiez passés à 5 heures* dont les deux découpages possibles ont été donné au début de ce chapitre (cf. table 8.1, page 121), on obtient le découpage donné dans la figure 8.13. Dans cette figure, les parties de phrase sont marquées en fonction de leur appartenance aux différentes propositions, le segment *à 5 heures* appartient aux propositions P_1 et P_2 (il fait partie des rattachements possibles des propositions P_1 et P_2) car son rattachement est imprécis.

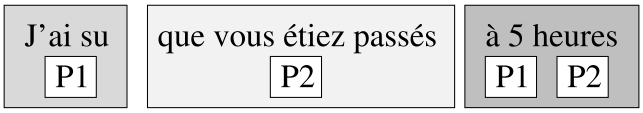


FIG. 8.13 - *Découpage effectué par le module de segmentation en propositions*

En plus des règles de segmentation données explicitement, il en existe une autre, implicite, qui permet d'exprimer le fait que le début d'une phrase, si celle-ci ne commence pas par un marqueur de début de proposition, est dans une proposition qui est la proposition principale. C'est cette règle qui permet de construire une proposition pour *j'ai su* dans l'exemple précédent. Ce module de l'analyseur morpho-syntagmatique effectue, en même temps que la segmentation en propositions, le repérage et le rattachement des propositions subordonnées. Pour cela, il se base sur les débuts de proposition : tout début de proposition marqué par une conjonction de subordination ou un pronom relatif indique une profondeur supplémentaire dans l'arborescence des propositions (dans l'exemple précédent, P_2 est à une profondeur supérieure à P_1, c'est donc une proposition subordonnée à P_1). Pour l'instant, ce module essaie de faire au mieux pour déterminer l'imbrication des propositions subordonnées. Cependant, il ne traite que le cas des relatives et conjonctives pour lesquelles la position du pronom permet de connaître la place dans l'arborescence des propositions. Le traitement de l'arborescence des propositions est cependant incomplet du fait que nous n'avons pas encore abordé le problème de la coordination de propositions.

8.5.5 Établissement des liens potentiels

Ce module, dont le fonctionnement est basé sur un système de règles contenu dans une grammaire, a pour but d'établir les liens pouvant exister entre les différents segments d'une même proposition (la portée d'un segment est donc limitée aux propositions auxquelles il appartient) ainsi que les rattachements possibles d'une proposition (c'est-à-dire, déterminer l'élément auquel on peut rattacher le connecteur de la proposition). Il ne s'occupe pas des liens entre les propositions (liens entre subordonnées et principale, par exemple) car ceux-ci ont déjà été établis lors de l'étape de détection des propositions (construction de l'arborescence des propositions en fonction des pronoms relatifs).

La grammaire utilisée

Les liens sont établis à partir d'un système de règles qui sont des règles de dépendance entre segments d'une même proposition (pour déterminer les liens entre segments) ou des règles de dépendance entre deux segments appartenant à des propositions de niveaux différents (pour déterminer le rattachement du connecteur de la proposition subordonnée à un des segments de la proposition subordonnante). Toutes ces règles sont basées sur deux notions : la notion de "régi" et celle de "régissant". Un segment régi est un segment qui est rattaché sous certaines conditions à un autre, celui-ci étant appelé le régissant. Les règles servent donc à définir les conditions de rattachement entre un régi et un régissant, ceux-ci étant repérés par leur catégorie (SN, SV_conj, etc.). Les conditions de rattachement que nous avons extraites portent sur la position relative de ces deux éléments, sur la présence de segments d'une certaine catégorie entre ces deux éléments ainsi que sur les éléments constituant les segments. Une règle est donc composée de trois parties : une partie définition du régi, une partie définition du régissant et une partie contraintes. Afin de préciser ces notions, nous allons maintenant donner quelques exemples de règles (les deux premières sont des règles internes à une proposition, la dernière étant une règle permettant d'effectuer un rattachement hors de la proposition) :

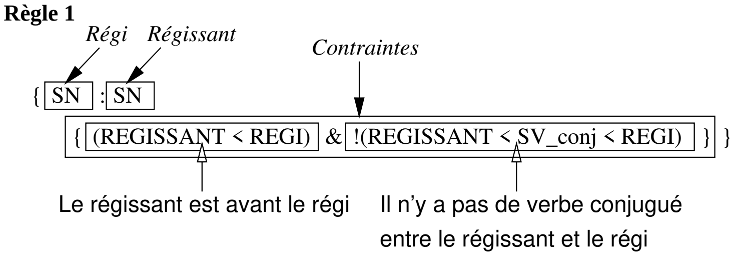
(1) Pour pouvoir rattacher un SN (SN1) à un autre SN (SN2) il faut, d'une part que le segment régissant (SN2) soit avant le régi (SN1) et, d'autre part qu'entre ces deux segments il n'y ait pas de verbe conjugué (SV_conj).

(2) Pour pouvoir rattacher un SAdj (groupe adjectival) à un SN, il faut que, si le régissant (le SN) est avant le régi (le SAdj) alors il ne doit pas y avoir de SV_conj entre les deux, et si le régi est avant le régissant alors le régissant ne doit pas contenir de préposition et il ne doit pas y avoir de SV_conj entre le régi et le régissant.

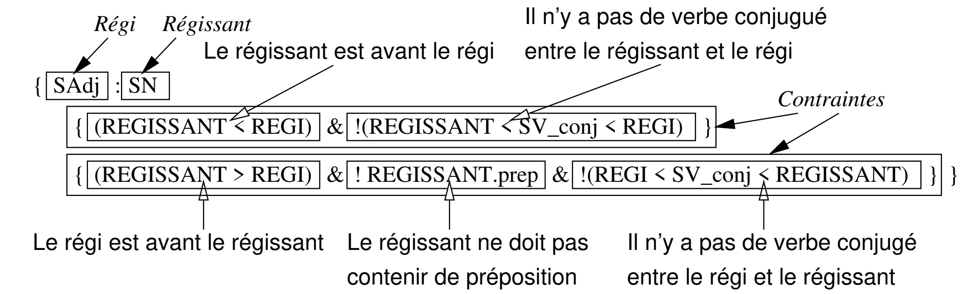
(3) Pour pouvoir rattacher un SQ_rel (connecteur du type qui, que, etc.) à un SN, il faut que le régissant (le SN) soit avant le régi (le pronom) et qu'il n'y ait pas de verbe conjugué entre ces deux segments.

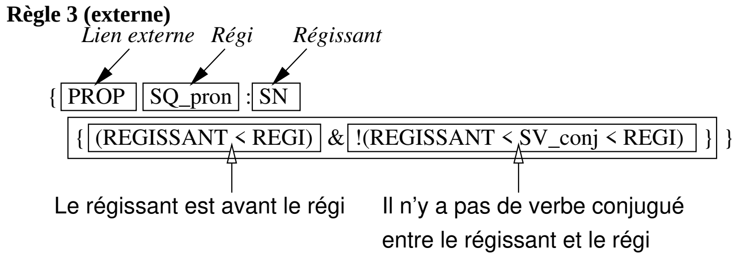
Ces trois règles dont données dans la grammaire de construction des liens potentiels, de la façon suivante :

Règle 1

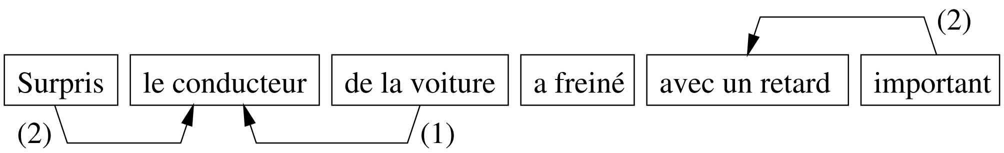


Règle 2

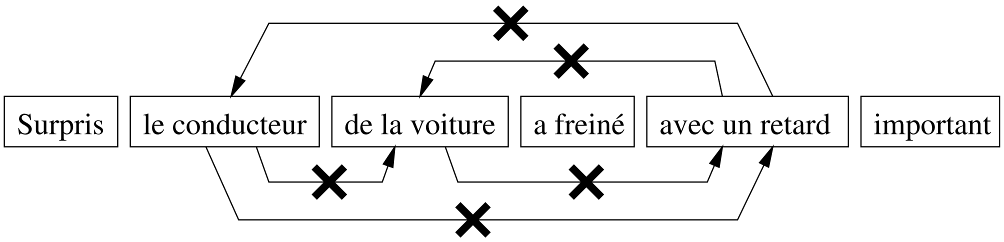




Les deux premières règles nous permettent d'effectuer les rattachements suivants :

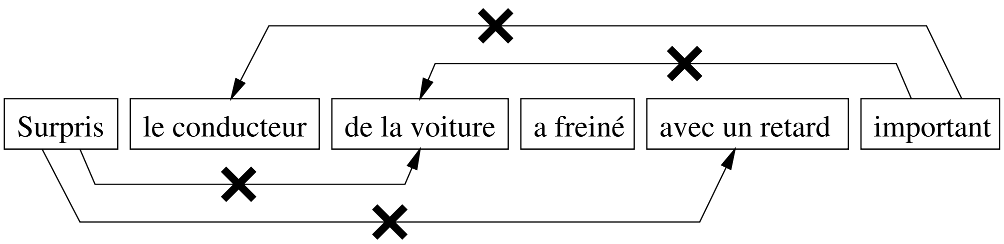


Par contre, la règle (1) permet de ne pas construire les liens suivants :



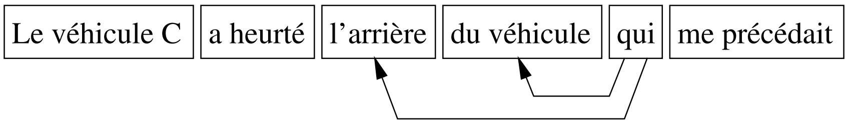
En effet, pour la règle (1), les liens *le conducteur* → *de la voiture*, *le conducteur* → *avec un retard* et *de la voiture* → *avec un retard* sont impossibles car le régissant est après le régi. De même, les liens *avec un retard* → *de la voiture* et *avec un retard* → *le conducteur* sont impossibles car le régi et le régissant sont séparés par un SV_conj (*a freiné*).

La règle (2) permet de ne pas construire les liens suivants :

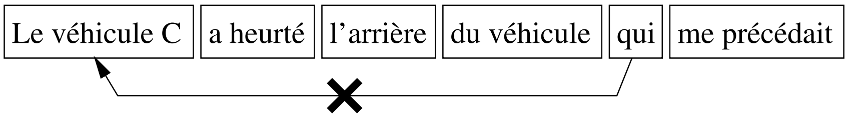


Le lien *surpris* → *de la voiture* est impossible car *de la voiture* contient une préposition (*de*). Les liens *surpris* → *avec un retard*, *important* → *de la voiture* et *important* → *le conducteur* sont impossibles car le régi et le régissant sont séparés par un SV_conj (*a freiné*).

La troisième règle permet d'effectuer les rattachements suivants :



La règle (3) permet de ne pas construire les liens suivants :



Le lien *qui* → *le véhicule C* est impossible car le régi et le régissant sont séparés par un SV_conj (*a heurté*).

Cette implémentation de la construction des liens possibles est seulement une première étape pour un tel système, il reste encore beaucoup de problèmes à traiter tant au niveau des liens entre segments internes à une même proposition qu'au niveau des liens entre segments de propositions différentes. Cependant, cette approche nous semble satisfaisante du fait qu'elle permet de limiter l'explosion combinatoire tout en ne limitant pas excessivement le nombre de liens : le choix des liens valides sera effectué par le module d'analyse des relations actancielles en fonction de données sémantiques.

L'algorithme de création des liens

Le fonctionnement de ce sous-module consiste à appliquer l'ensemble des règles de la grammaire des liens potentiels à l'ensemble des arcs du graphe de segments en tenant compte de leur appartenance à une même proposition. Pour effectuer cette opération, nous n'avons pas jugé nécessaire de suivre l'ordre du graphe et ce pour deux raisons : d'une part, suivre l'ordre du graphe en évitant de traiter deux fois le même arc demande un marquage des arcs et donc un "nettoyage" de ces derniers, d'autre part déterminer si un arc est avant un autre dans le graphe est de complexité identique

que l'on parcourre, en premier lieu, le graphe en suivant les chemins ou non. Cela nous donne l'algorithme suivant :

```
- Pour chacun des arcs du graphe Faire
    - Pour chacune des règles de la liste Faire
        - Si le groupe correspond au régi de la règle Alors
            - rechercher les régissants qui conviennent
            - créer un lien par régissant possible
```

8.6 Analyse détaillée

Afin de préciser le rôle, le fonctionnement et les résultats rendus par l'analyseur morpho-syntagmatique, nous allons maintenant détailler l'analyse de la phrase suivante :

je roulais à vive allure sur la partie droite de la chaussée.

Tout au long de cette section nous présenterons les graphes résultant des différentes étapes de notre analyseur. Dans ces graphes, les nœuds sont représentés par des carrés et les arcs par des traits. Les arcs en grisé sont les arcs d'ordre de priorité supérieur à 0 (plus le grisé est clair, plus la priorité est faible). La première étape consiste à effectuer un découpage préliminaire du texte. Le résultat de ce découpage est donné dans la figure 8.14.

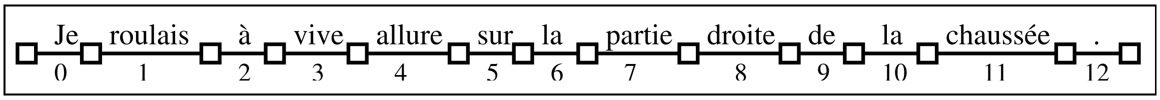


FIG. 8.14 - *Exemple : Graphe résultant du découpage préliminaire du texte*

La recherche des unités morphologiques est effectuée à partir du graphe de chaînes de caractères résultant de ce découpage et en utilisant le dictionnaire des formes fléchies.

Recherche des unités morphologiques

Lors de cette étape, il est possible d'avoir, pour une même chaîne de caractères, plusieurs unités morphologiques différentes (polycatégorie). C'est le cas, dans notre exemple, des "mots" *vive*, *la*, *partie*, *droite* et *chaussée*. Ces différentes possibilités sont indiquées sur la figure 8.15 par différents arcs correspondant à la même chaîne de caractères.

Au mot *vive*, correspond non seulement le nom (arc 16, de priorité faible) mais aussi l'adjectif (arc 17) et le verbe (vivre - arc 18). Au mot *la*, on associe plusieurs unités morphologiques : le déterminant (arc 21, arc 29), le clitique (arc 22, arc 30) et le nom (arc 23, arc 31), ce dernier ayant une priorité moindre car il est moins courant que les deux autres. Pour *partie* les deux unités possibles sont, d'une part le nom (arc 25), d'autre part le participe passé (arc 24). Quant à *droite* les deux possibilités sont, d'une part le nom (arc 26), d'autre part l'adjectif (arc 27). Les deux arcs correspondants à *chaussée* sont d'une par le nom (arc 32), d'autre part le participe passé (verbe chausser - arc 33). Dans ce graphe, l'ordre des numéros des arcs représentant un même mot dépend uniquement de l'ordre des "mots" dans le dictionnaire des formes fléchies.

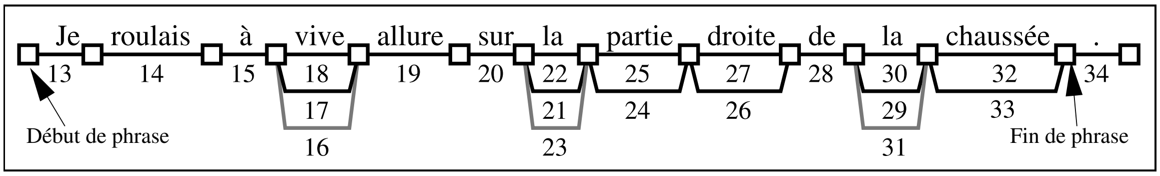


FIG. 8.15 - *Exemple : Graphe des unités morphologiques du texte*

La reconnaissance des phrases dans le texte permet de repérer une seule phase qui se termine par un point. L'arc correspondant à ce point ne sera plus pris en considération lors des traitements ultérieurs.

Construction des segments maximaux sûrs

Le regroupement des unités en segments maximaux sûrs (dont le résultat est donné sur la figure 8.16) tient compte de la polycatégorie. Pour la chaîne de caractères *à vive allure*, il existe différents regroupements possibles. Soit on forme un SN (arc 45) à partir des arcs 15, 17 et 19 (*vive* est alors considéré comme un adjectif), soit on forme un SN (arc 47) à partir des arcs 15 et 16 (*vive* est alors considéré comme un nom) puis un autre SN avec l'arc 19. La priorité de l'arc correspondant au segment créé à partir de l'arc 19 est normalement de 1 (priorité due à la règle de constitution d'un SN à partir uniquement d'un nom) mais la normalisation du graphe entraîne le report de cette priorité sur l'arc formé à partir des arcs 15 et 16 qui est donc d'ordre de priorité 2 du fait de la priorité associée à l'arc 16. Il faut remarquer que l'arc 18 (*vive* considéré comme verbe conjugué) n'est pas utilisé.

Pour la chaîne de caractères *sur la partie* on a en fait plusieurs possibilités : regrouper les trois mots en un seul segment de type SN (arc 41) en prenant en compte les arcs 20, 21 et 25, regrouper *sur* et *la* en un seul segment de type SN (arc 44) en prenant en compte les arcs 20 et 23 et former un autre segment avec *partie*, ce dernier étant soit de type SN (arc 42) en considérant l'arc 25 soit de type SV_p_passe (arc 43) en considérant l'arc 24. Les arcs 44 et 42 ont des priorités moindres du fait, soit de la priorité des unités morphologiques qu'ils englobent (c'est le cas du 44 qui englobe le 23), soit du fait de la priorité associée à la règle permettant de créer le segment (ce qui est le cas pour le 42 : un SN composé uniquement d'un nom est peu fréquent).

A partir de *droite* on a deux possibilités : soit on forme un SN de priorité 1 (arc 39) en utilisant l'arc 26, soit on forme un SAdj (arc 40) en utilisant l'arc 27.

Dans le cas de *de la chaussée* on se retrouve dans un cas similaire à celui de *sur la partie* on a donc plusieurs regroupements possibles qui découlent des unités possibles pour *la* et pour *chaussée*. Ces différents regroupements sont les suivants : arc 35 (segment de type SN) qui regroupe les arcs 28, 29 et 32, l'arc 38 (segment de type SN) qui regroupe les arcs 28 et 30 suivi de l'arc 36 (segment de type SN) qui contient uniquement l'arc 32 ou de l'arc 37 (segment de type SV_p_passé) qui contient uniquement l'arc 33. Les arcs 36 et 38 ont des priorités moindres du fait, soit de la priorité des unités morphologiques qu'ils englobent (c'est le cas du 38 qui englobe le 30), soit du fait de la priorité associée à la règle permettant de créer le segment (ce qui est le cas pour le 36 : un SN composé uniquement d'un nom est peu fréquent).

On notera que les arcs 22 et 29 (*la* considéré comme clitique) ne sont pas utilisés.

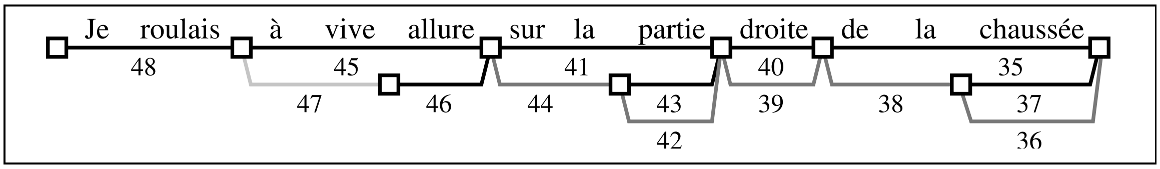


FIG. 8.16 - *Exemple : Graphe des segments maximaux sûrs*

Extraction des meilleurs chemins et construction des liens potentiels

Dans notre exemple, l'extraction des meilleurs chemins ne pose aucun problème car il n'existe qu'un seul chemin d'ordre de priorité total égal à zéro : c'est donc le meilleur. Le graphe résultant de ce découpage est donc composé d'un seul chemin représenté sur la figure 8.17. Le découpage en propositions n'est pas visible sur ces graphes mais lui non plus ne pose pas de problème : comme il n'y a qu'un seul verbe, il n'y a qu'une seule proposition. De même, les liens potentiels ne sont pas directement visibles sur le graphe mais font partie des attributs contenus dans les unités portées par les graphes et sont visualisés sur la figure 8.18.

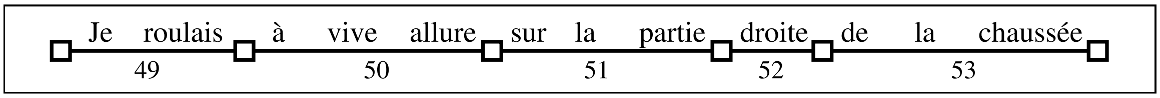


FIG. 8.17 - *Exemple : Graphe résultant de l'extraction des meilleurs chemins*

Le résultat rendu par l'analyse morpho-syntagmatique et utilisé par les autres modules de l'analyseur linguistique est composé de l'ensemble des graphes construits au cours de ces différentes étapes ainsi que de la description des propositions existantes.

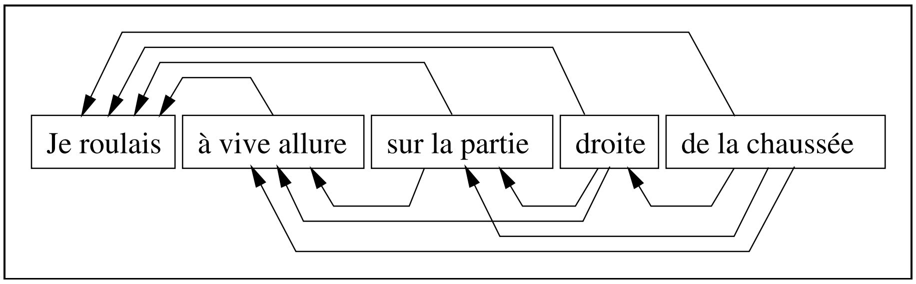


FIG. 8.18 - *Exemple : liens déterminés par l'analyse morpho-syntagmatique*

8.7 Bilan du travail sur l'analyseur morpho-syntagmatique

Comme nous venons de le voir, le résultat fourni par l'analyseur morpho-syntagmatique est composé de l'ensemble des graphes (et listes de graphes) correspondants à ces différentes étapes reliés les uns aux autres par des liens "père-fils" (un segment maximal sûr est le père de chacune des unités morphologiques qui le composent) ainsi que de la description des propositions. Pour chaque proposition sont indiqués son verbe, son connecteur (qui peut être vide), ses groupes sûrs ainsi que les groupes susceptibles d'appartenir à cette proposition (les groupes possibles). Il y a donc une certaine redondance des informations (une proposition connaît l'ensemble de ses composants et chacun des composants connaît la proposition à laquelle il appartient) mais celle-ci permet ensuite une plus grande facilité d'accès aux informations et facilite donc les traitements suivants (effectués par les modules syntaxico-sémantiques). Cet ensemble de graphes et de listes de graphes et donné, à l'analyseur des relations actancielles, sous forme d'un objet de la classe `ListeDeGraphes`, objet qui, comme nous l'avons indiqué précédemment, peut contenir à la fois des objets de type `GrapheSimple` et des objets de type `ListeDeGraphes`. Afin de permettre une vérification des résultats de l'analyseur morpho-syntagmatique, nous avons mis en place, comme nous le verrons dans la section 11.3.2, page 204 une interface graphique qui permet non seulement de visualiser le résultat de ce module mais aussi de paramétrer l'analyse. Un aperçu de cette interface sera aussi donné en Annexe C.2.1.

Cet analyseur étant avant tout une maquette ayant pour but de déterminer la validité d'une approche, il nous est, pour l'instant, difficile d'en évaluer les performances. Cependant, cet analyseur, tout en étant incomplet (il ne traite pas tous les cas et, entre autres, ne traite pas le problème important qu'est la coordination), nous a permis de bien débroussailler le problème qu'est l'analyse morpho-syntagmatique d'un texte. Ce système a prouvé que l'approche pour laquelle nous avons opté (découpage de la phrase en segments maximaux sûrs en fonction de données réduites - nous n'utilisons pas les constructions syntaxiquement admissibles des verbes) convenait à notre de style de problème et qu'elle permettait d'effectuer une grosse partie du travail d'analyse. De plus, l'implémentation que nous avons réalisée (analyse par l'intermédiaire des GSP) est efficace contre l'explosion combinatoire (dans l'analyse détaillée, malgré le nombre de possibilités au niveau des unités morphologiques, cet analyseur arrive à ne fournir qu'une seule solution au module d'analyse des relations actancielles).

Comme nous l'avons déjà indiqué en début de chapitre, ce style d'approche est utilisé non seulement pour l'analyse du français [Vergne, 1995] mais aussi pour celle de l'anglais [Georgiev, 1991]. Les différences principales entre le système que nous avons développé et celui de J. Vergne se situent, d'une part dans les donnés utilisées pour effectuer le découpage en segments, d'autre part dans la représentation de ces données. En effet, alors que J. Vergne n'utilise volontairement qu'un dictionnaire restreint, notre de but est d'avoir, à terme, un dictionnaire des formes fléchies le plus complet possible. De plus, les règles utilisées par J. Vergne sont données sous forme procédurale alors que celles que nous utilisons sont données sous forme déclarative afin de permettre une meilleure évolution du système. Une autre différence existe entre notre système et celui de J. Vergne et concerne le résultat fourni. En effet, alors que J. Vergne effectue, au cours de son analyse, systématiquement un choix quand deux possibilités se présentent, nous avons décidé, afin de ne pas trop limiter les capacités du système, de ne pas faire de choix au niveau de l'analyse morpho-syntagmatique (ceux-ci seront effectués plus tard en fonction de données sémantiques). Les résultats fournis par les deux systèmes sont non seulement différents en nombre mais aussi en forme : le résultat que nous fournissons contient les segments extraits du texte mais aussi les liens potentiels entre les segments ainsi que les différentes propositions détectées dans le texte ce qui n'est pas le cas du résultat fourni par l'analyseur de J. Vergne.

La maquette d'analyseur morpho-syntagmatique ayant montré qu'une telle approche convenait à notre problème, nous comptons continuer dans cette direction pour la suite du développement. Cependant, étant donné qu'il existe d'autres systèmes basés sur les mêmes principes et ayant fait leurs preuves, nous ne comptons pas tout refaire mais utiliser les résultats de ces derniers.

Chapitre 9

L'analyseur des relations actancielles

Le module d'analyse des relations actancielles a pour but de déterminer les procès présents dans le texte et de construire l'ensemble des entités et des contraintes imposées par chacun de ces procès. Pour effectuer son analyse, il dispose du résultat fourni par l'analyseur morpho-syntagmatique (segments, hiérarchie des propositions et liens potentiels entre les segments), d'un ensemble de connaissances dont la principale est le lexique et utilise les autres modules syntaxico-sémantiques pour résoudre des problèmes bien spécifiques. Il a donc un double rôle : tout d'abord celui d'analyseur (il analyse la structure actancielle de chaque proposition afin d'établir les procès), d'autre part celui de contrôleur des autres modules syntaxico-sémantiques.

Avant de détailler le fonctionnement de ce module, nous allons tout d'abord expliciter les choix effectués au niveau du lexique et donner le formalisme mis en place pour représenter les procès ainsi que les entités et relations qui leur sont liées.

9.1 Le lexique

Le lexique est composé à la fois des entrées se trouvant dans le dictionnaire des unités grammaticales et de celles se trouvant dans le dictionnaire des unités lexicales. La partie que nous utilisons au niveau de l'analyseur des relations actancielles (et que nous allons étudier dans ce chapitre) concerne les informations syntaxico-sémantiques associées à chacune des entrées.

Comme dans tout système de compréhension automatique de texte, le lexique joue ici un rôle capital : il permet de construire les entités et événements dont il est question dans le texte et de déterminer les relations sémantiques entre unités et événements correspondants. Il est utilisé par le module d'analyse des relations actancielles mais construit en fonction du modèle du sous-système chargé de la construction de la scène. C'est lui qui fournit à l'analyseur linguistique une partie des données sémantiques dont il a besoin.

Un des problèmes les plus importants concernant les entrées lexicales est lié au traitement de la polysémie (un verbe comme *rouler* peut signifier "déplacer (un corps arrondi) en le faisant tourner sur lui-même", "avancer au moyen de roues, de roulettes", "avancer, voyager dans un véhicule à roues" ... ; un mot comme *bureau* peut faire référence soit au meuble, soit à la pièce).

Pour rendre compte cette polysémie il existe une solution qui consiste à déterminer, pour chacun des mots polysémiques, un sens "premier" ou "plus abstrait" - un "archétype cognitif" selon J-P Desclés [Desclés, 1991], ou un "noyau de sens" selon d'autres auteurs (pour une revue, voir [Victorri & Fuchs, 1996]) - à partir duquel les autres sens pourraient se dériver en fonction des spécificités contextuelles. Cette position présente plusieurs difficultés : l'existence d'un sens "premier" n'est à l'heure actuelle qu'une hypothèse parmi d'autres. D'autre part, en admettant cette hypothèse, il reste à trouver des mécanismes généraux de dérivation et à les formaliser. Ce travail, par ailleurs très intéressant, n'est pas encore assez avancé pour pouvoir être utilisé (cf. [Victorri & Fuchs, 1996]) aussi nous lui avons préféré une solution qui, tout en étant moins satisfaisante, a cependant l'avantage d'être plus facilement utilisable en traitement automatique du langage.

Cette solution consiste à décrire, pour chaque entrée polysémique, plusieurs *schémas de sens* qui devront être sélectionnés en fonction du contexte et/ou de constructions grammaticales. La distinction entre les différents sens associés à un mot donné se fera donc en fonction, d'une part des constructions syntaxiquement admissibles pour ce mot, d'autre part des catégories sémantiques des éléments environnants. Cette position peut relever d'options théoriques sans doute tout aussi sujettes à débat mais elle permet de disposer de représentations partielles liées au problème analysé, extensibles à volonté et plus directement exploitables en traitement automatique - sans préjuger du fonctionnement cognitif ou linguistique plus profond et sans anticiper sur une compréhension plus sûre du phénomène. Les représentations partielles étant liées au problème étudié, elles sont, dans notre cas, établies en relation avec le sous-système chargé d'effectuer la construction de la scène, ceci explique que nous considérions le lexique comme une des composantes de la coopération entre ce sous-système et l'analyseur linguistique.

On peut retrouver cette idée d'association de schémas de sens à partir de constructions syntaxiques dans la structure des lexies de [Mel'čuk *et al.*, 1995] ou de la structure des entrées du TLF [51].

Ce choix de représentation de la polysémie nous permet d'avoir un lexique évolutif sans pour cela être obligés d'effectuer des modifications au niveau du module d'analyse des relations actancielles. Ceci nous permet par exemple de ne pas tenir compte actuellement de l'aspect argumentatif des textes (les schémas de sens ne contiennent aucune indication de ce type) sans nous en enlever la possibilité.

La création et la confection du lexique étant un travail à long terme dans lequel nous ne pouvions nous engager avant d'avoir déterminé un format pour représenter les entrées, nous n'avons travaillé que sur un très petit nombre de mots (une soixantaine d'entrées) dont essentiellement des verbes étant donné leur importance dans la langue. Nous allons donc maintenant donner une description du lexique en détaillant tout d'abord les verbes puis nous aborderons les autres catégories.

9.1.1 Les verbes

Les schémas de sens sont associés aux entrées verbales par l'intermédiaire des constructions grammaticalement possibles. Dans le lexique nous précisons donc, pour chaque entrée verbale, un ensemble de constructions syntaxiquement admissibles ainsi que l'ensemble des schémas de sens correspondant à chacune de ces constructions. L'ensemble des constructions et schémas de sens que nous avons mis en place pour un verbe donné n'est pas exhaustif : il correspond aux constructions et schémas nécessaires pour l'interprétation des constats d'accidents automobiles (par exemple, nous n'avons pas mis dans notre lexique les schémas correspondant à "rouler une cigarette", "rouler quelqu'un" ...). Cependant, un simple ajout dans le lexique des constructions et schémas permettrait de prendre en compte ces différents sens lors d'une analyse.

Dans ce lexique, une construction liée à un verbe est composée de trois parties.

51. Trésor de la Langue Française [TLF, 1971],[Gorcy, 1996]

Tout d'abord, une partie précisant les caractéristiques de la construction (voix - active, passive, emploi, ...). Cette partie est suivie de la construction proprement dite, à savoir la liste des constituants syntaxico-sémantiques. Pour chaque constituant, est précisé son caractère obligatoire ou facultatif, sa catégorie syntaxique, sa catégorie sémantique ainsi que les contraintes portant sur la structure de l'unité syntaxique. Ces constituants correspondent aux compléments intégrés au verbe et non à l'ensemble des compléments susceptibles d'être rattachés au verbe. Les compléments intégrés au verbe correspondent aux compléments essentiels du verbe par opposition aux compléments circonstanciels. Ces compléments ne sont pas toujours obligatoires mais sont considérés comme faisant partie intégrante du sens du verbe. Le fait de considérer tel ou tel complément comme intégré relève d'un choix totalement arbitraire dépendant du sens associé au verbe. Ainsi, le fait de considérer que le verbe *rouler* (comme tous les autres verbes de déplacement) a un complément intégré de type sémantique "lieu" vient du fait que nous considérons, qu'en général, un déplacement s'effectue dans un lieu. Les compléments présents dans une construction ne forment donc qu'une partie des compléments que l'on peut rattacher au verbe : un verbe peut avoir plus de compléments que n'en contient sa construction. Dans la dernière partie se trouve l'ensemble des schémas de sens associés à cette construction. Au niveau des constructions, on peut remarquer une certaine redondance dans le fait que nous indiquons les constructions non seulement à la voix active mais aussi à la voix passive alors que cette dernière peut être déduite de la voix active. Cette redondance est tout à fait volontaire et a pour but de simplifier les traitements ultérieurs.

Nous allons, maintenant, montrer sur l'exemple du verbe *rouler* le format adopté pour les constructions des verbes (cf. table 9.1).

Tout d'abord une première remarque concernant le format adopté : les catégories syntaxiques des éléments acceptables comme sujet ou comme complément d'objet direct sont toujours explicitées dans la construction alors que celles-ci ne varient pas d'une construction à une autre (quand on recherche un sujet de type nominal il est toujours de la même forme). Nous aurions très bien pu avoir une définition générale d'un sujet de type nominal mais, comme dans le cas des voix active et passive dans une construction, nous avons préféré, au risque d'une certaine redondance, indiquer à chaque fois les différentes possibilités de façon à simplifier les traitements ultérieurs. Cette solution est viable dans le cadre de la maquette que nous avons développée mais il faudra opter pour une technique plus automatique permettant d'éviter la redondance pour effectuer une implémentation "grandeur nature". Cette automatisation de la définition de certains composants pourra s'effectuer soit par l'intermédiaire d'un "précompilateur" qui remplacera toutes les apparitions des composants par leur définition, soit grâce à un ensemble de définitions de ces composants fourni au module d'analyse des relations actancielles.

Dans la table 9.1, nous pouvons voir que nous avons associé trois constructions grammaticales différentes au verbe *rouler* (sujet de type "humain" + complément d'objet direct de type "objet physique", sujet de type "humain" + complément de lieu, sujet de type "véhicule" + complément de lieu - les deux dernières constructions étant une conséquence directe des critères que nous avons choisis pour distinguer les éléments de sens qui relèvent du système linguistique de ceux qui dépendent de nos connaissances sur le domaine, cf. page 89) ainsi que quatre schémas de sens différents ("déplacer quelque chose en le faisant tourner sur lui-même", "se déplacer à l'aide d'un véhicule à roues", "se déplacer en tournant sur soi-même" et "se déplacer, en parlant d'un véhicule à roues") dont deux sont rattachés à la seconde construction.

rouler :
construction 1 : voix active; emploi transitif;
sujet (SN1) {SN!(prep), clitique(cas nominatif), q_pron(cas nominatif)} : humain
complément direct (SN2) {SN!(prep), clitique(cas accusatif), q_pron(cas accusatif)} : objet physique
→ schéma 1 : ''déplacer quelque chose en le faisant tourner sur lui-même''
construction 2 : voix active; emploi intransitif;
sujet (SN1) {SN!(prep), clitique(cas nominatif), q_pron(cas nominatif)} : humain
complément prépositionnel (SP1) facultatif {SN(prep), SAdv} : lieu
→ schéma 2 : ''se déplacer à l'aide d'un véhicule à roues''
→ schéma 3 : ''se déplacer en tournant sur soi-même''
construction 3 : voix active; emploi intransitif;
sujet (SN1) {SN!(prep), clitique(cas nominatif), q_pron(cas nominatif)} : véhicule
complément prépositionnel (SP1) facultatif {SN(prep), SAdv} : lieu
→ schéma 4 : ''se déplacer, en parlant d'un véhicule à roues''

TAB. 9.1 - *Format adopté pour les constructions syntaxico-sémantiques*

Nous allons, maintenant donner quelques explications relatives à différentes expressions se trouvant dans la table 9.1. `sujet (SN1) {SN!(prep), clitique(cas nominatif), q_pron(cas nominatif)} : humain` signifie que le sujet du verbe est de catégorie sémantique "humain" et peut être représenté soit par un segment nominal sans préposition, soit par un clitique ou un pronom relatif de cas nominatif. De plus, `(SN1)` permet de nommer les segments associés à ce composant de la construction de façon à pouvoir les référencer dans le reste de la construction (c'est-à-dire dans les schémas de sens associés à la construction). `complément prépositionnel (SP1) facultatif {SN(prep), SAdv} : lieu` signifie que le verbe admet comme complément essentiel un complément prépositionnel de lieu qui peut être représenté soit par un segment nominal contenant une préposition, soit par un groupe adverbial.

Une fois le format des constructions déterminé, il faut fixer celui des schémas de sens (cf. table 9.2 pour un exemple). Ceux-ci sont constitués d'une liste d'indications utiles à différents modules de l'analyseur linguistique et divisées, pour l'instant, en trois parties (le développement des différents modules experts pouvant entraîner l'ajout d'une ou plusieurs parties dans cette structure) :

1. une partie *entités*, qui donne la liste des entités dont la construction (création ou identification d'une entité déjà créée) est imposée par l'unité lexicale elle-même. Ce sont en général des entités implicitement liées au sens considéré (quand *rouler* signifie "avancer, voyager dans un véhicule à roues", il existe une entité implicite représentant le véhicule utilisé pour le déplacement).

2. une partie *relations* qui décrit les propriétés et relations entre les entités qui sont impliquées par le verbe (entités qui peuvent être liées soit au verbe, soit aux autres composants de la construction). Dans cette partie nous indiquons, entre autres, l'acteur de l'événement lié au verbe (`E.acteur = SN1` signifie que l'acteur du déplacement est l'entité référencée par le segment associé à SN1).

3. une partie *instructions* où se trouvent des demandes de calculs complexes qui devront être effectués par un des modules de l'analyseur linguistique ainsi que des demandes de vérification sur des relations que doivent respecter les entités implicites (`identifier(IP, conducteur(V, SN1))` indique qu'il faut identifier l'entité référencée par le segment associé à SN1 comme étant le conducteur de V pendant l'intervalle IP).

Dans le cas de schéma 2 du verbe *rouler*, il faut préciser, par l'intermédiaire de ce dernier, un certain nombre d'implicites (existence d'un véhicule à roues avec prise en compte du fait que le véhicule est en mouvement sur tout l'intervalle du procès - IP[52]) ainsi qu'un certain nombre de relations (identification entre l'entité référencée par le sujet du verbe rouler et le conducteur du véhicule, indication concernant l'acteur du déplacement - le véhicule). C'est uniquement à partir de ces indications que l'analyseur des relations actancielles pourra construire les différentes contraintes sémantiques qui dépendent directement de ce sens de *rouler*. Voici à titre d'illustration le "schéma 2" pour *rouler* (les parties en *italique* correspondent à des commentaires) :

rouler :
 schéma 2:
  <u>entités</u>:
   v : véhicule *déclaration du véhicule implicite*
   E : déplacement *déclaration de l'événement associé à l'action de rouler*
  <u>relations</u>:
   à_roues(v) *le véhicule se déplace grâce à ses roues*
   E.acteur = v *le véhicule est l'acteur du déplacement*
  <u>instructions</u>:
   identifier(IP, conducteur(v, SN1)) *identifier l'entité référencée par le SN1 (le sujet) comme étant le conducteur de* V *pendant l'intervalle* IP [53]

TAB. 9.2 - *Un des schémas de sens associés à* rouler

L'instruction `identifier(...)` est destinée au module d'identification des entités (cf. section 7.2, page 105) et impose une contrainte forte en même temps qu'une vérification sur V en fonction de l'analyse du sujet SN1 : le conducteur du véhicule V est le référent calculé pour SN1. Cette contrainte est relativisée à l'intervalle IP. Nous verrons l'utilisation de ces constructions et schémas dans la section 9.2.

52. voir section 7.2, page 107 pour la définition de cet intervalle.

53. Cette dernière relation nous permet de simplifier, au niveau de l'analyseur linguistique, la recherche du conducteur du véhicule implicite mais devra, dans des développements ultérieurs être remplacée par un traitement plus compliqué.

9.1.2 Les autres entrées lexicales

Pour les autres entrées lexicales, le travail est moins avancé mais le principe est le même : à une entrée lexicale nous associons un ou plusieurs schémas de sens avec ou sans construction grammaticale.

Le cas de la polysémie nominale est un peu différent de celui de la polysémie verbale. En effet, comme nous venons de le voir, dans le cas des verbes les indices permettant de faire la différence portent plus sur les constructions grammaticalement possibles que dans le cas des noms. Pour les noms, les indices se situent plutôt dans l'environnement : nature sémantique des verbes par exemple. Dans le cas d'un mot comme *bureau* qui fait référence à un meuble ou à une pièce, les schémas associés ne mentionneront que la catégorie sémantique ("meuble" ou "pièce"). Le choix entre les deux sens sera effectuer en fonction du verbe : dans *Il entre dans le bureau* le sens à considérer est "pièce", dans *Le bureau a deux tiroirs* le sens à considérer est "meuble" mais dans *Il a rangé son bureau* les deux sens sont possibles.

Il existe un certain nombre de noms ayant des constructions grammaticales permettant de déterminer un sens différent. C'est le cas, par exemple, pour un nom comme *partie*. En effet, une construction comme "*partie* + adjectif de catégorie sémantique forme ou localisation + segment nominal commençant par *de* et contenant un nom de catégorie sémantique lieu" permet de définir une portion du lieu. À ce type de nom, comme dans le cas des verbes, le schéma de sens sera associé à l'entrée lexicale par l'intermédiaire d'une construction grammaticale (cf. définition de *partie* ci-dessous; les parties en *italique* correspondent à des commentaires).

```
partie
 {  construction (1)
    0?  {  ident SA1 groupe {SAdj} catsem localisation forme }
    1   {  ident SN1 groupe {SN(prep) & prep.lemme = de &
           nom.catsem = lieu} catsem lieu }
    schéma {
        instructions { restriction_spatiale(SN1, SA1) } } }
        le lieu est restreint à la partie définie par l'adjectif
```

Le cas des adverbes et des adjectifs est un peu spécial : il faut, par l'intermédiaire du lexique, pouvoir préciser les modifications qu'ils entraînent sur leur régissant (c'est-à-dire sur le nom ou le verbe auquel ils se rattachent). Pour cela, nous donnons une définition de ce régissant en indiquant ses catégories syntaxiques possibles et nous utilisons le terme de "régissant" pour faire référence à l'entité référencée par ce dernier. C'est le module d'analyse des relations actancielles qui sera chargé de déterminer avec précision ce régissant (il utilise pour cela les liens fournis par l'analyseur morpho-syntagmatique). Ceci nous donne, par exemple, pour *lentement* (les parties en *italique* correspondent à des commentaires) :

lentement
{ `construction (1)`
`schéma {`
`régissant { v_conj, v_inf, v_p_passe, v_p_pres }`
`instructions { vitesse(régissant) = "petite" } } }`
l'action représentée par le verbe (c'est-à-dire l'événement rattaché au verbe) se fait à une vitesse qui est petite

L'ensemble des spécifications données ci-dessus pour le lexique (aussi bien pour les verbes que pour les autres mots) correspondent aux données qui sont utilisées par le module d'analyse des relations actancielles. Cependant, ce lexique contiendra en plus, pour chacune de ses entrées, un certain nombre de données nécessaires aux autres modules syntaxico-sémantiques (la classe aspectuelle du verbe pour le module aspectuo-temporel, ...). Ces données ne sont pas encore intégrées au lexique car les modules mis en cause ne sont pas encore implémentés.

9.2 Principe de fonctionnement

Le module d'analyse des relations actancielles a pour but de déterminer l'ensemble des relations et entités marquées par le texte. Pour cela, il travaille, contrairement au module effectuant l'analyse morpho-syntagmatique, proposition par proposition et utilise, en plus du résultat de l'analyseur morpho-syntagmatique, les données contenues dans le lexique. Son premier travail consiste à déterminer les constructions syntaxico-sémantiques utilisées ainsi que les segments de la phrase auxquels elles s'appliquent. Cette étape, qui est la plus importante dans le fonctionnement de ce module, consiste à déterminer pour chaque entrée lexicale la construction à prendre en compte en utilisant les données lexicales sur les autres segments présents dans la proposition afin de pouvoir, ensuite, choisir le schéma de sens à prendre en compte et en déduire les relations et entités liées aux procès. Le principe de fonctionnement du module d'analyse des relations actancielles peut être rapproché de celui de WEP [54]. Tout au long de ce processus, le module d'analyse des relations actancielles peut faire appel aux experts pour résoudre des problèmes bien spécifiques.

L'appel au module d'identification des entités est effectué pour tout segment de type nominal et pour les unités morphologiques de catégorie clitique ou pronom, ainsi que pour les entités dont la création est explicitement demandée par un schéma de sens (les entités se trouvant dans la partie "entités" d'un schéma de sens). De même, les instructions faisant référence au module d'identification des entités (les instructions `identifier`, par exemple), entraînent un appel à ce module. Dans le cas des segments nominaux où l'entrée lexicale correspondant au nom est marquée comme ne nécessitant pas de création d'entité (c'est le cas de *partie* dans le cadre de la construction "partie de" ou de *allure* dans le cadre de la construction "à <adj> allure"), il n'y a pas d'appel au module d'identification des entités : ces segments nominaux sont considérés comme étant des modifieurs des segments qui leur sont liés ou comme des locutions adverbiales et ne doivent donc pas donner lieu à une création ni à une recherche d'entité.

L'appel au module d'analyse des relations spatio-temporelles peut être déclenché soit par une catégorie sémantique de type spatiale (lieu, localisation, verbe de mouvement, ...), soit par un appel explicite demandé par un schéma associé à une construction sous forme d'une instruction.

L'appel au module d'analyse des relations aspectuo-temporelles est effectué systématiquement pour chaque verbe conjugué afin de déterminer les intervalles temporels à lui associer (IP et IM). Un appel à ce module peut aussi être déclenché soit par une catégorie sémantique de type temporelle (complément circonstanciel de temps, ...), soit par un appel explicite demandé par un schéma associé à une construction.

Comme nous venons de le voir, lors de l'ensemble du processus, le module d'analyse des relations actancielles est confronté à un certain nombre de choix (choix de construction quand il en existe deux qui sont équivalentes, de segment à associer à un composant de construction, de schéma de sens, ...). Pour effectuer ces choix nous avons mis en place un certain nombre d'heuristiques qui sont données dans la section suivante. Ces heuristiques ne sont, bien entendu, pas complètes et seront amenées à

54. Word Expect Parsing, [Small, 1980] et [Adriaens & Small, 1988]

évoluer au fur et à mesure du développement du système. De plus, elles peuvent toujours être remises en cause par le sous-système de construction de la scène quand, comme nous le verrons dans la section 11.2, elles génèrent une incohérence.

Afin de préciser le fonctionnement du module d'analyse des relations actancielles, nous allons maintenant donner un exemple d'analyse simple (un exemple plus compliqué et plus détaillé sera donné en section 10.1). Prenons la phrase *je marchais sur la chaussée*.

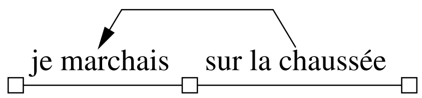


FIG. 9.1 - *Analyse morpho-syntagmatique de "Je marchais sur la chaussée."*

```
moi :
 {  construction (1) (catsem humain)
     schéma (1) { entités { (identass s, typeident personne) } } }
marcher :
 {  construction (1) (voie active) (emploi intransitif)
     sujet  {  ident SN1 groupe {SN !(prep), clitique (cas nominatif),
                q_pron (cas nominatif)} catsem humain }
     cc?    {  ident SN2 groupe {SN (prep), SAdv} catsem lieu }
     schéma (1) {
            entités { (identass e, typeident déplacement) }
            relations { e.acteur = SN1 } } }
 {  construction (2) (voie active) (emploi intransitif)
     sujet  {  ident SN1 groupe {SN !(prep), clitique (cas nominatif),
                q_pron (cas nominatif)} catsem mécanisme }
     schéma (2) {
            entités { (identass e, typeident fonctionnement) }
            relations { e.acteur = SN1 } } }
sur : { construction (1) (catsem lieu) }
le (le1) :
chaussée :
 {  construction (1) (catsem lieu)
     schéma (1) { entités { (identass r, typeident route) }}}
```

TAB. 9.3 - *Entrées lexicales servant à interpréter "Je marchais sur la chaussée."*

À partir du résultat de l'analyse morpho-syntagmatique (cf. figure 9.1) et des entrées lexicales associées (cf. table 9.3) la recherche des constructions valides nous permet de déterminer que seule la `construction 1` de *marcher* est utilisable. Il découle de cette construction l'utilisation du `schéma 1` et donc la présence d'une entité de type "déplacement" associée au verbe (`identass e, typeident déplacement`) et que l'acteur de ce déplacement est l'entité référencée par le sujet (`e.acteur = SN1`). Les appels effectués au module d'identification des entités permettent d'associer à *je* l'entité S_0 de type "personne" et à *sur la chaussée* l'entité R_0 de type "route". Les appels effectués au module d'analyse des relations spatiales permettent de spécifier que S_0 se trouve sur la route R_0. Les appels effectués au module d'analyse des relations aspectuo-temporelles permettent de déterminer les intervalles liés à *marcher* (IP_1 et IM_1) et les relations entre ces différents intervalles (IM_1 est inclus dans IP_1 et IM_1 est avant l'intervalle d'énonciation IE_0).

Procès n° 1 :

TEXTE : `''je marchais sur la chaussée''`

OBJETS :

S_0 : personne, R_0 : route, E_0 : déplacement, IE_0 : intervalle,

IP_1 : intervalle, IM_1 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IP_1 \subset IM_1$ & $IM_2 < IE_0$

RELATIONS ÉVÉNEMENTIELLES :

E_0.acteur = S_0 &	*L'acteur du déplacement est* S_0
[IP_1] appartient_dim2(S_0, R_0)	*L'entité* S_0 *se trouve sur la route* R_0

9.3 Les heuristiques utilisées

Les heuristiques utilisées par ce module se situent à plusieurs niveaux de l'analyse dont, entre autres, au niveau du choix de la construction. Prenons le cas de *partie*. Ce mot admet plusieurs constructions dont une avec *de* (utilisée par exemple dans *partie droite de la chaussée*) et une "vide" (utilisée par exemple dans *il a gagné la partie*). Dans le cas de *partie droite de la chaussée*, les deux constructions définies pour *partie* sont valides car une construction "vide" s'applique quel que soit le contexte. Cependant, il nous a semblé plus judicieux de n'appliquer une construction "vide" que lorsqu'aucune autre ne s'applique. Une des heuristiques que nous avons mises en place consiste donc à considérer que, si une entrée lexicale admet plusieurs constructions et que l'une d'entre elles est vide, alors la construction "vide" ne sera pas prise en considération.

Le fait d'avoir choisi une construction n'implique pas nécessairement que les composants de celle-ci soient correctement déterminés. En effet, il existe un certain nombre de cas où, à un composant, correspondent plusieurs segments alors que la construction n'en nécessite qu'un seul. Pour résoudre ce problème, nous avons mis en place des heuristiques de choix de segments. Ces heuristiques sont basées sur les rattachements possibles des segments (rattachements déterminés par l'analyseur morpho-syntagmatique) et sur le nombre de constructions dans lesquelles les segments sont pris et ont la forme suivante :

- si, parmi les segments associés à un composant, il en existe un dont le seul rattachement possible est celui pris en compte par la construction alors considérer celui-là comme composant de la construction;
- si, parmi les segments associés à un composant, il en existe un n'appartenant qu'à une seule construction alors considérer celui-là comme composant de la construction;
- dans les autres cas, si le composant de la construction n'est pas le sujet (nous verrons le cas du sujet section 9.4 lors du traitement des ambiguïtés) garder le segment le plus "proche" comme composant de la construction.

Ces heuristiques nous permettent de définir le segment qui est probablement un composant de la construction mais n'entraînent pas nécessairement l'élimination des autres : ces derniers ne sont plus considérés comme compléments intégrés mais peuvent être considérés comme compléments rattachés.

Pour choisir parmi les schémas de sens quand il en existe plusieurs qui sont rattachés à la même construction (c'est le cas, par exemple, des schémas 2 et 3 du verbe *rouler*), l'heuristique que nous avons choisie consiste à utiliser le modèle des attentes du lecteur. En effet, comme nous l'avons vu précédemment (section 6.1, page 88), les données lexicales sur les éléments de la proposition ne permettent pas d'effectuer un choix parmi les différents sens liés à une construction donnée : des données contextuelles (portant, entre autres, sur le sujet du verbe) sont alors nécessaires. De plus, l'analyseur des relations actancielles dispose d'informations sur les entités et relations qui sont susceptibles de se trouver dans le texte (le modèle des attentes du lecteur). Nous avons donc choisi d'utiliser ces informations pour déterminer le schéma de sens le plus probable. Dans le cas d'un texte commençant par *J'ai roulé tout droit*, c'est grâce au fait que le modèle des attentes du lecteur spécifie que l'on s'attend à trouver un véhicule que nous choisirons le schéma de sens associer à "se déplacer à l'aide d'un véhicule à roues" et non celui associer à "se déplacer en tournant sur soi-même".

9.4 Les ambiguïtés

L'analyseur linguistique peut, malgré l'ensemble des connaissances auxquelles il a accès et les heuristiques employées, se trouver confronté à un certain nombre d'ambiguïtés qu'il est incapable de résoudre. Dans cette section, nous donnerons dans un premier temps un aperçu des différentes ambiguïtés auxquelles l'analyseur des relations actancielles peut être confronté ainsi qu'un "classement" des ces dernières en fonction de leurs répercussions sur la structure de représentation des procès. Dans un second temps nous indiquerons la représentation que nous avons utilisée pour les transmettre au sous-système de construction de la scène. Parmi les ambiguïtés que nous allons répertorier (un répertoire plus complet et plus étoffé est disponible dans [Fuchs, 1996]), il en existe qui ne proviennent pas directement de l'analyse des relations actancielles mais qui sont générées par les modules experts. Cependant, comme ces différentes ambiguïtés doivent être prises en compte par le module d'analyse des relations actancielles lors de l'intégration des résultats fournis par les modules experts au résultat de l'analyse linguistique, nous les traiterons en même temps que celles dues au module d'analyse des relations actancielles. Nous ne tiendrons pas compte des ambiguïtés détectées par l'analyse morpho-syntagmatique car celles-ci ont généré plusieurs chemins qui sont traités séparément au niveau de l'analyse des relations actancielles et donnent lieu à la transmission de plusieurs résultats d'analyse en parallèle.

9.4.1 Les différentes sources d'ambiguïtés

Nous allons maintenant donner, à l'aide de quelques exemples, différentes ambiguïtés auxquelles le module d'analyse des relations actancielles peut être confronté.

Une des ambiguïtés rencontrée provient du module d'identification des entités. Prenons le cas de la phrase suivante : *je n'ai pu éviter la voiture qui arrivait en face. la voiture* peut faire référence, soit à une voiture dont on a parlé précédemment, soit à une nouvelle voiture qui serait spécifiée à ce moment là uniquement par la relative. Dans ce cas seul le sous-système de construction de la scène (en s'aidant éventuellement du résultat de l'analyse du schéma) peut savoir de quelle voiture il est question. Ce type d'ambiguïté, que nous appellerons par la suite une "ambiguïté d'association", provient du fait qu'à un segment donné sont associées deux entités différentes. Cependant, au niveau de la structure de représentation du procès, le fait de faire référence à tel ou tel véhicule n'est pas fondamentalement différent et ne nécessite pas la création de deux représentations différentes, cette ambiguïté entraînera donc une "alternative locale" au niveau de la structure de représentation.

Le module d'identification des entités peut générer non seulement des ambiguïtés d'association mettant en cause deux entités de même type mais aussi des ambiguïtés d'association mettant en cause deux entités de types différents (ceci se produit, en général, lors de la recherche de l'antécédent d'un pronom). Dans ce cas, l'ambiguïté d'association entraîne aussi une "ambiguïté de construction". En effet, prenons la phrase *il roulait devant moi* avec deux antécédents possibles pour *il* (un de catégorie "véhicule", un autre de catégorie "humain"). Le fait que *rouler* admette, entre autres, les constructions suivantes : "sujet de catégorie sémantique humain + complément circonstanciel de lieu (construction 2)" et "sujet de catégorie sémantique objet_mobile + complément circonstanciel de lieu (construction 3)" entraîne deux interprétations possibles et donc deux représentations différentes pour un même procès. Ce type d'ambiguïtés nécessite donc ce que nous avons appelé une "alternative globale" au niveau de la structure du résultat fourni. Une phrase comme "le conducteur du véhicule roulant devant moi" peut elle aussi générer une "ambiguïté de construction". En effet, la catégorie sémantique de *conducteur* étant "humain" et celle de *véhicule* étant "véhicule", les constructions (2) et (3) de *rouler* sont possibles : une ayant comme sujet *conducteur*, l'autre ayant comme sujet *véhicule*.

Le module d'analyse des relations spatio-temporelles peut être lui aussi confronté à des ambiguïtés. Prenons, par exemple, le morceau de phrase suivant : *sur la partie droite de la chaussée*. Dans ce cas, l'adjectif *droit* qui précise quelle est la partie de route à prendre en considération peut avoir deux significations (marquées dans le lexique par deux schémas de sens différents) : soit il faut prendre en compte la partie rectiligne de la route, soit il faut prendre en compte le bord droit de la route. Ce choix, entre les deux significations, ne peut être fait que par le sous-système de construction de la scène. Le fait que le segment de route à prendre en considération soit rectiligne ou situé sur le bord droit de la route n'intervient que dans les relations précisant l'aspect du segment de route et ne modifie pas la structure de représentation du procès. Ceci nous a conduit à représenter ce type d'ambiguïté (que nous appellerons par la suite une “ambiguïté de relation”) par une alternative locale.

Dans les constructions associées aux verbes, le sujet est un composant un peu particulier et les ambiguïtés sur celui-ci sont à traiter avec particulièrement d'attention. En effet, prenons la phrase *le chat de la voisine traversant le carrefour est noir*. Le *chat* et la *voisine* étant deux “éléments” susceptibles de *traverser*, ils sont donc tous les deux sujets possibles du verbe *traverser*. Le problème, dans ce cas, vient du fait que, contrairement aux compléments circonstanciels, le sujet doit être unique et par conséquent l'analyseur des relations actancielles doit effectuer un choix entre les deux sujets. Malheureusement celui-ci ne peut pas effectuer ce choix car il ne dispose pas des connaissances nécessaires, il doit donc fournir les deux interprétations au sous-système de construction de la scène qui, si il le peut, effectuera le choix. De plus, les deux interprétations étant fondamentalement différentes (le sujet de *traverser* étant l'actant, changer de sujet correspond à changer d'actant), nous représentons cette ambiguïté (que nous avons appelée “ambiguïté de structure syntaxique” car les deux interprétations correspondent à deux constructions syntaxiques différentes) par une alternative globale.

De même qu'il existe des ambiguïtés dues au choix de construction, il en existe qui sont dues au choix de schéma de sens (ambiguïtés que nous appellerons “ambiguïtés de schémas”). En effet, comme nous avons pu le voir pour le verbe *rouler*, il est possible d'avoir, pour une même construction, plusieurs schémas de sens (cf. construction 2 de *rouler*). Comme nous l'avons vu dans la partie précédente, le choix de favoriser un schéma de sens plutôt qu'un autre se fait en fonction du contexte d'énonciation et du contexte linguistique (contenus dans le modèle des attentes du lecteur). Cependant, il est tout à fait possible que ce modèle ne contienne aucune information permettant de choisir un schéma, dans ce cas (comme dans les précédents), les deux interprétations sont fondamentalement différentes et donnent lieu à une représentation par l'intermédiaire d'une alternative globale.

Cette liste des ambiguïtés n'est pas exhaustive mais donne un aperçu des problèmes auxquels nous avons été confrontés. Cela nous a aussi permis de nous apercevoir qu'il y avait en fait deux grands types d'ambiguïtés : des ambiguïtés majeures qui correspondent à des interprétations fondamentalement différentes et qui demandent la création d'une représentation de procès par interprétation possible et des ambiguïtés mineures pour lesquelles il n'est pas nécessaire de créer plusieurs représentations. L'intérêt de cette distinction réside dans la complexité du résultat à fournir au sous-système de construction de la scène : en ne créant pas de représentation supplémentaire nous évitons de fournir au sous-système de construction de la scène un trop grand nombre de données.

9.4.2 La représentation des ambiguïtés dans le résultat de l'analyseur linguistique

Comme nous avons pu le voir, il existe des ambiguïtés nécessitant la création d'autant d'interprétations qu'il existe de possibilités (les ambiguïtés nécessitant une représentation par l'intermédiaire d'une alternative globale) et d'autres, moins gênantes au niveau de la structure de la représentation, qui ne nécessitent pas de créer plusieurs interprétations (celles qui sont représentables par une alternative locale). Nous allons voir, maintenant, pour chacun de ces grands types d'alternatives, la représentation utilisée.

Les ambiguïtés représentables par une "alternative locale"

Pour représenter les ambiguïtés d'association simples (celles ne générant pas d'autres ambiguïtés), nous avons décidé de représenter l'ensemble des entités mises en cause par un nom commun et d'associer à chacune d'entre elles un numéro d'hypothèse. Nous obtenons dans ce cas, et dans le cadre de notre exemple (*je n'ai pu éviter la voiture qui arrivait en face*), une description des objets de la forme suivante : `V : (1)` $\mathtt{V}_1$ `: véhicule, (2)` $\mathtt{V}_2$ `: véhicule` (V peut faire référence soit à l'objet $\mathtt{V}_1$ soit à l'objet $\mathtt{V}_2$). Toutes les relations faisant intervenir un élément pour lequel il existe une ambiguïté d'association feront référence à l'élément générique ce qui nous donne par exemple : `à_roues(V)`.

Pour représenter les ambiguïtés de relation, nous avons choisi de ne transmettre qu'une seule représentation du procès dans lequel les différentes possibilités seront repérées par un numéro d'hypothèse. Si nous reprenons notre exemple (*sur la partie droite de la chaussée*), nous aurons donc 2 hypothèses : une indiquant que le segment de route référent de *sur la partie* est droit, l'autre signalant que ce segment est à droite d'un certain repère qui sera ajouté au niveau des entités. Ces relations seront définies de la manière suivante : (1) droit(SR$_0$) (2) { lie(Rep$_0$, R$_0$) & inf(position_y(SR$_0$, Rep$_0$), "0")}. Le repère n'étant à prendre en considération que pour la seconde hypothèse, il sera défini de la façon suivante : (2) Rep$_0$: repère.

Les ambiguïtés nécessitant une représentation par une "alternative globale"

Comme nous l'avons vu précédemment, dans le cas des ambiguïtés de structure syntaxique, des ambiguïtés de construction ou des ambiguïtés de schémas, il faut fournir au sous-système de construction de la scène, plusieurs interprétations différentes tout en lui indiquant qu'elles sont à traiter en parallèle. Pour cela, nous avons nommé les représentations de procès de la façon suivante : un nom est composé d'une part du numéro de procès, d'autre part du numéro de version du procès (ce qui nous donne des noms de la forme <numéro de procès>_<numéro de version>). Deux représentations de procès devant être traitées en parallèles auront le même numéro de procès mais pas le même numéro de version. Une ambiguïté de structure syntaxique (comme celle générée par la phrase *le chat de la voisine traversant le carrefour est noir*) donnera donc lieu à la création de deux représentations de procès : une de nom 1_0 où l'actant de *traverser* sera l'entité associée à l'expression *de la voisine* et une autre de nom 1_1 où l'actant de *traverser* sera l'entité associée à l'expression *le chat*.

9.5 Implémentation

À partir des principes de fonctionnement et des heuristiques exposés ci-dessus nous avons mis en place un algorithme de fonctionnement du module d'analyse des relations actancielles divisé en 6 étapes qui sont les suivantes :

1. Recherche dans le lexique des unités lexicales correspondant aux unités morphologiques utilisées.

2. Pour les pronoms, faire appel au module d'identification des entités afin de les préciser. Cette opération fait appel au traitement des anaphores et est réalisée dès le début de l'analyse afin de réduire le nombre de choix possibles par la suite.

3. Pour chacune des entrées lexicales trouvées, rechercher l'ensemble des constructions qui peuvent s'appliquer en fonction des segments régis par le segment auquel appartient cette entrée, supprimer les constructions vides puis déterminer le schéma de sens à prendre en compte.

4. Pour chacune des constructions ainsi obtenues, si à un composant correspondent plusieurs éléments faire un choix parmi ces éléments en appliquant les heuristiques données dans la section 9.3, page 172.

5. Pour chacune des constructions, supprimer, parmi les liens précédemment établis par l'analyseur morpho-syntagmatique ceux qui sont inutiles en se basant sur les composants des constructions (pour tout élément intégré à une construction supprimer les liens non pris en compte).

6. Déterminer les procès contenus dans le texte puis, pour chacun des éléments appartenant au procès (aussi bien au niveau unité lexicale qu'un niveau segment), faire appel aux modules spécialisés (analyse des relations aspectuo-temporelles, identification des entités et analyse des relations spatio-temporelles) en fonction des caractéristiques de l'élément étudié et des constructions et schémas de sens choisis pour cet élément.

Pour effectuer la recherche des constructions valides pour une entrée lexicale donnée, nous utilisons l'ensemble des rattachements possibles déterminés lors de l'analyse morpho-syntagmatique. En effet, nous considérons qu'un segment ne peut être composant d'une construction donnée que s'il existe un lien entre le segment considéré et le segment auquel appartient l'unité lexicale pour laquelle est effectuée la recherche.

Ces différentes étapes sont pour l'instant réalisées à l'aide de règles ou d'heuristiques qui, contrairement à celles de l'analyseur morpho-syntagmatique, sont données de façon procédurale. Une des améliorations à réaliser dans le cadre d'une nouvelle implémentation serait d'implémenter ces règles et heuristiques de façon déclarative.

Ces étapes, ainsi que leur nombre, sont bien sûr modifiables mais nous pensons que le principe de base du fonctionnement (rechercher les constructions acceptables à partir du résultat de l'analyseur morpho-syntagmatique et du lexique afin de déterminer les schémas de sens) est à garder.

Chapitre 10

Résultats obtenus

Afin de donner un aperçu des capacités de l'analyseur linguistique nous allons détailler son fonctionnement sur un exemple (*je roulais à vive allure sur la partie droite de la chaussée*). Cette phrase ayant déjà été traitée au niveau de l'analyseur morpho-syntagmatique (cf. section 8.6, page 155), nous ne détaillerons que la partie syntaxico-sémantique (analyse des relations actancielles et utilisation des 3 experts).

10.1 Analyse détaillée

L'analyse syntaxico-sémantique s'effectue à partir du résultat de l'analyse morpho-syntagmatique et est centrée, comme nous l'avons vu précédemment, autour du module d'analyse des relations actancielles. Pour effectuer son analyse ce module utilise, en plus du résultat de l'analyse morpho-syntagmatique et du lexique, le modèle des attentes du lecteur (modèle résultant les conditions d'énonciation liées au domaine des constats d'accidents de la route) dont nous donnons ici un extrait :

OBJETS :	s_0 personne, s_1 personne, v_0 véhicule, v_1 véhicule[55]
RELATIONS :	conducteur(v_0, s_0), conducteur(v_1, s_1), véhicule(s_0, v_0), véhicule(s_1, v_1), énonciateur(s_0).

55. s_0 et s_1 sont différents, de même que v_0 et v_1.

Nous allons maintenant voir, étape par étape, le déroulement de l'analyse effectuée par le module d'analyse des relations actancielles sur la phrase : *je roulais à vive allure sur la partie droite de la chaussée.*

L'étape 1 permet d'extraire du lexique les unités lexicales suivantes :

```
moi : { construction (1) (catsem humain)
        schéma (1) { entités { (identass s, typeident personne) }}}
rouler : { aux avoir }
  { construction (1) (voie active) (emploi transitif)
      sujet   { ident SN1 groupe {SN !(prep), clitique (cas nominatif),
                q_pron (cas nominatif)} catsem humain }
      cd      { ident SN2 groupe {SN !(prep), clitique (cas accusatif),
                q_pron (cas accusatif)} catsem objet_physique }
      schéma (1) { déplacer quelque chose en le faisant tourner sur lui-même
        entités { (identass e, typeident déplacement) }
        relations { e.acteur = SN1, e.objet = SN2}}}
  { construction (2) (voie active) (emploi intransitif)
      sujet   { ident SN1 groupe {SN !(prep), clitique (cas nominatif),
                q_pron (cas nominatif)} catsem humain }
      cc?     { ident SN1 groupe {SN (prep), SAdv} catsem lieu }
      schéma (2) { se déplacer à l'aide d'un véhicule à roues
        entités { (ident v, typeident véhicule)
                  (identass e, typeident déplacement)}
        relations { à_roues(v), e.acteur = v}
        instructions { identifier(IP, conducteur(v, SN1))
                       identifier(IP, véhicule(SN1, v))} }
      schéma (3) { se déplacer en tournant sur soi-même
        entités { (identass e, typeident déplacement) }
        relations { e.acteur = SN1 }}}
  { construction (3) (voie active) (emploi intransitif)
      sujet   { ident SN1 groupe {SN !(prep), clitique (cas nominatif),
                q_pron (cas nominatif)} catsem véhicule }
      cc?     { ident SN1 groupe {SN (prep), SAdv} catsem lieu }
      schéma (4) { se déplacer en parlant d'un véhicule à roues
        entités { (identass e, typeident déplacement) }
        relations { e.acteur = SN1}}}
à : {}
```

vif (vif1) : {}
allure :
  { construction (**1**)
    schéma (**1**) { entités { *association d'une entité à allure*}}}
  { construction interne (**2**) (catégorie SAdv)
    0 { ident SN1 groupe {SN (prep) & (adj) & (prep.lemme = à)} } }}
sur : { construction (**1**) (catsem lieu) }
le (le1) : {}
partie :
  { construction (**1**)
    schéma { entités { *association d'une entité à partie*}}}
  { construction (**2**)
    catégorie outils
    0?{ ident SA1 groupe {SAdj} catsem localisation forme}
    1 { ident SN1 groupe {SN (prep) & prep.lemme = de &
      nom.catsem = lieu }
    schéma { instructions { restriction_spatiale(IP, SN1, SA1)}}}
droit :
  { construction (**1**) (catsem localisation)
    schéma (**1**) { régissant { nom }
      entités { (ident rep, typeident repère) }
      relations { inf(position_y(régissant, rep))}}}
  { construction (**2**) (catsem forme)
    schéma (**2**) { régissant { nom } relations { droit(régissant)}}}
de : {}
chaussée :
  { construction (**1**) (catsem lieu)
    schéma (**1**) { entités { (identass r, typeident route) }}}

Du fait de la catégorie morphologique des unités considérées, l'étape 2 (recherche des catégories sémantiques pour les pronoms) n'entraîne aucune demande auprès du module d'identification des entités.

Dans l'étape 3, il faut déterminer, pour chaque unité lexicales, les constructions valides en fonction du reste de la proposition. Parmi les unités utilisées, seules *rouler*, *allure* et *partie* contiennent des constructions. La recherche des composants d'une construction pour une entrée lexicale se fait parmi les groupes régis par cet élément.

Pour *rouler*, la recherche des composants s'effectue parmi les groupes suivants : *à vive allure*, *sur la partie*, *droite* et *de la chaussée* auxquels on ajoute le groupe courant *je roulais*. La construction 1 de rouler est impossible car il n'y a pas de complément d'objet direct (cd) possible. La construction 3 de rouler est impossible car il n'y a pas de sujet possible. Pour la construction 2 il n'y a pas d'ambiguïté sur le sujet (*je*) mais pour le complément de lieu (cc) il existe plusieurs possibilités : *sur la partie* qui peut être de catégorie sémantique "lieu" à cause de *sur* et *de la chaussée* qui est de catégorie sémantique "lieu" à cause de *chaussée*.

La construction 2 admet plusieurs schémas qui correspondent à deux sens différents de *rouler* : "se déplacer à l'aide d'un véhicule à roues" (schéma 2) et "rouler sur soi-même" (schéma 3). Le premier schéma demande la présence (ou la création si celle-ci est possible) d'un véhicule tel que celui-ci ait comme conducteur et comme responsable le sujet du verbe *rouler*, le second schéma n'a aucun implicite. Par conséquent pour choisir entre ces deux schémas, nous devons déterminer s'il existe un véhicule vérifiant les conditions du schéma 2. Il y a alors trois possibilités : le véhicule existe (le sujet de *rouler* est alors conducteur) et alors nous prendrons en priorité le schéma 2, le véhicule n'existe pas et il est impossible d'en créer un (parce que le sujet de *rouler* est indiqué comme ne pouvant pas être un conducteur) alors nous choisirons le schéma 3 et enfin, la troisième possibilité, le véhicule n'existe pas mais il est possible de le créer (il n'y a aucune indication sur le sujet de *rouler*), dans ce cas nous serons obligé de traiter en parallèle les deux schémas. Pour vérifier les conditions portant sur le véhicule implicite au schéma 2, le module d'analyse des relations actancielles fait appel au module d'identification des entités qui dispose des entités et relations liées au contexte d'énonciation et de celles issues de l'analyse des phrases précédentes (entités et relations contenues dans le modèle des attentes du lecteur). La phrase en cours d'étude étant la première d'un texte, les seules entités dont dispose ce module sont celles issues des conditions d'énonciation. Le fait que nous soyons dans un cadre de constats d'accidents de la route, implique que le locuteur (donc *je*) est conducteur d'un véhicule (voir définition du modèle initial des attentes du lecteur, page 92 ou page 181). Par conséquent, le schéma choisi, dans ce contexte, est le schéma 2 et l'entité associée au véhicule implicite de ce schéma est v_0. En cas de problème lors de l'analyse du reste du texte il sera néanmoins toujours possible de revenir sur ce choix et de prendre le second schéma.

Pour *partie*, la recherche des composants s'effectue parmi les groupes suivants : *droite* et *de la chaussée* auxquels on ajoute le groupe courant *sur la partie*. La construction 1 qui est une construction vide (permettant d'indiquer que *partie* peut s'utiliser sans construction : *il a gagné la partie*) est une construction valide. La construction 2 ne pose pas de problème : au composant 0 est associé *droite* et au composant 1 est associé *de la chaussée*. Le fait que, dans cette construction, *partie* soit de catégorie "outils" entraîne que ce mot est alors considéré comme un modifieur des segments qui s'y rattachent : ce segment ne nécessitera pas de recherche d'entité contrairement aux segments nominaux ordinaires.

Pour *allure*, la recherche des composants s'effectue parmi les groupes suivants : *sur la partie*, *droite* et *de la chaussée* auxquels on ajoute le groupe courant *à vive allure*. La construction 1 qui est une construction vide (permettant d'indiquer que *allure* peut s'utiliser sans construction : *il a de l'allure*) est une construction valide. La construction 2 est une construction spéciale : elle n'agit que sur le segment auquel appartient *allure* et n'est valide que si il existe des composants de ce segment qui vérifie les conditions `(prep) & (adj) & (prep.lemme = à)` (ce segment doit contenir une préposition - à - et un adjectif). Dans notre cas, la construction 2 est donc valide. Le fait que cette construction soit de catégorie "SAdv" (le segment n'est plus considéré comme un segment nominal mais comme un segment adverbial) entraîne qu'il n'y aura pas de recherche d'entité pour ce segment.

Au cours de cette étape, l'heuristique appliquée pour choisir les constructions (suppression des constructions vides) entraîne la suppression des constructions vides de *partie* et de *allure*.

Au début de l'étape 4 on se trouve donc dans la situation décrite dans la figure 10.1 où les traits en gras représentent les liens qui peuvent être utilisés par les composants des constructions.

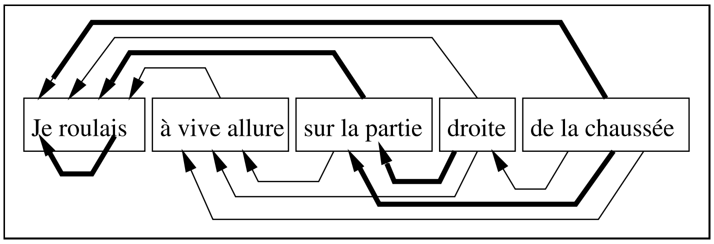


FIG. 10.1 - *Exemple : composants possibles des constructions*

Le seul composant pour lequel il existe plusieurs possibilités est le complément de lieu de *rouler*. En effet, à ce composant peut être associé soit le segment représentant *sur la partie*, soit celui représentant *de la chaussée*. Or *de la chaussée* est le seul segment possible pour le composant 1 de la construction choisie pour *partie* (la construction "partie + de" ayant été détectée comme probable, et le composant 1 de cette construction étant obligatoire il est obligatoire de lui associer un élément), il est donc pris par la construction de *partie* et n'est donc plus considéré comme composant de *rouler*. Le complément de lieu de *rouler* est donc *sur la partie*. Ce rattachement pourrait aussi être effectué par le module d'analyse des relations spatio-temporelles qui devrait être capable de refuser une construction du style "*rouler* + de" si une autre, plus simple, existe.

On obtient alors l'ensemble des liens représenté sur la figure 10.2 (les traits en gras représentent les segments réellement pris dans les constructions) :

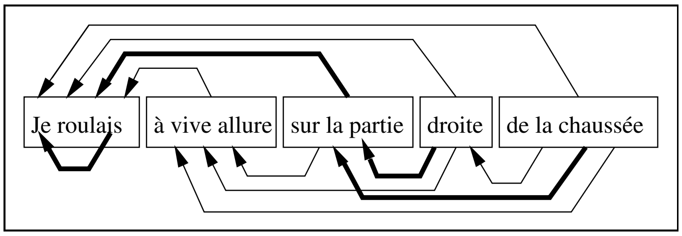


FIG. 10.2 - *Exemple : composants des constructions après vérification des liens*

Après l'étape 5 (suppression, pour les segments intégrés à des constructions, des liens non pris en compte), on obtient l'ensemble des liens réellement utilisés et visualisés sur la figure 10.3.

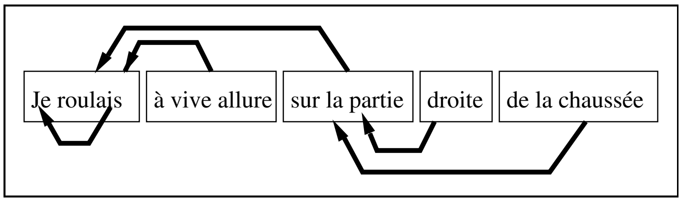


FIG. 10.3 - *Exemple : ensemble des liens réellement utilisés*

Cet ensemble regroupe les liens pris en compte dans les constructions ainsi que, pour les éléments qui ne sont inclus dans aucune construction, l'ensemble des liens déterminés par l'analyseur morpho-syntagmatique (lien *à vive allure* → *je roulais*).

L'étape 6 consiste à faire appel aux divers modules spécialisés. Dans le cadre de notre exemple, les appels au module d'identification des entités sont les suivants :

élément fourni au module d'identification des entités	raison de l'appel	entité rendue
création déplacement	entité demandée par *rouler*	$\mathtt{E}_0$ déplacement
je	unité de type clitique	$\mathtt{S}_0$ personne
de la chaussée	segment de type SN	$\mathtt{R}_0$ route

Du fait des indications au niveau des constructions utilisées pour *partie* et *allure* (`catégorie outils` et `catégorie SAdv`), il n'y a pas d'appel effectué au module d'identification des entités pour les segments *sur la partie* et *à vive allure*.

Dans notre exemple, le schéma associé à *partie* fait explicitement appel au module d'analyse des relations spatio-temporelles par l'intermédiaire de l'instruction `restriction_spatiale`. L'expression *partie droite de la chaussée* permet de caractériser une partie de la route. Cette caractérisation peut prendre deux formes : la partie de route considérée est soit droite, soit à droite d'un certain repère lié à la route définie par *chaussée*. L'appel effectué lors de l'exécution de l'instruction `restriction_spatiale` contenue dans le schéma de *partie*, entraîne donc l'ajout, au niveau du procès, des entités et relations suivantes : $\mathtt{SR}_0$ `: segment_route, ` $\mathtt{Rep}_0$ `: repère, droit(`$\mathtt{SR}_0$`), lie(`$\mathtt{Rep}_0$`, `$\mathtt{R}_0$`), inf(position_y(`$\mathtt{SR}_0$`, `$\mathtt{Rep}_0$`), "0") et inclus(`$\mathtt{SR}_0$`, `$\mathtt{R}_0$`)`. Les entités $\mathtt{SR}_0$ et $\mathtt{Rep}_0$ sont demandées directement par le module spatio-temporel au module d'identification des entités. $\mathtt{SR}_0$ correspond au segment de route décrit par *partie droite*, $\mathtt{Rep}_0$ correspond au repère auquel réfère cette même expression pris dans le sens de “partie de route se trouvant à droite d'un certain repère”. La relation `lie(`$\mathtt{Rep}_0$`, `$\mathtt{R}_0$`)` permet de préciser que le repère de référence n'est pas totalement indéterminé mais lié à la route $\mathtt{R}_0$ (référent de *chaussée*). La relation `inf(position_y(`$\mathtt{SR}_0$`, `$\mathtt{Rep}_0$`), "0")` quant à elle indique la position du segment de route par rapport à ce repère : “être à droite d'un certain repère” signifie avoir une position sur l'axe des ordonnées de ce repère qui soit inférieure à 0. Ces deux relations ainsi que le repère sont créés en utilisant le schéma 1 de *droit*. L'utilisation du schéma 2 de *droit* entraîne quant à lui l'ajout de la relation `droit(`$\mathtt{SR}_0$`)` qui indique que le segment de route concerné est droit. Les deux sens de *droit*, qui correspondent à deux hypothèses différentes, forment ce que nous avons appelé précédemment une “ambiguïté mineure”. Ils ne donnent donc pas lieu à la création de deux interprétations différentes mais sont repérés par des numéros d'hypothèse. La relation `inclus(SR0, R0)` est valable quel que soit le sens pris pour *droit* et signifie que le segment de route défini par *partie droite* est un élément de la route R_0 (référent de *chaussée*). Toutes les relations caractérisant le segment de route considéré sont indépendantes du temps.

Les constituants de la phrase qui impliquent un appel au module d'analyse des relations spatiales du fait de leur catégorie sémantique sont les suivants : *rouler* (verbe de mouvement) et *sur la partie* (complément de lieu auquel on ajoute ses composants *droite* et *de la chaussée* ainsi que le verbe auquel il est lié). Pour l'instant, seul le second appel génère un résultat qui permet de préciser la position du sujet du verbe *rouler* par rapport à la partie de route décrite. Cette relation, qui dépend du temps, s'exprime de la façon suivante (où $[IP_1]$ indique l'intervalle sur lequel elle est vérifiée) : $[IP_1]$ `appartient_dim2(S0, SR0)`.

Dans notre exemple, les schémas ne contiennent aucune instruction devant faire appel au module d'analyse des relations aspectuo-temporelles. Les seuls appels effectués sont ceux de début de procès dont les résultats sont les intervalles IP_1, IM_1 et IE_0 et celui déclenché par le verbe *rouler* dont le résultat est une série de relations entre les différents intervalles définis pour ce procès : $IM_1 < IE_0$ & $IM_1 \subset IP_1$.

Le résultat rendu par l'analyseur linguistique (qui est la concaténation des résultats obtenus au cours des différentes étapes de l'analyse) représente le procès déterminé par l'analyseur des relations actancielles. Dans notre cas, il est de la forme suivante :

Numéro : 1_0

TEXTE : ``je roulais à vive allure sur la partie droite de la chaussée''

OBJETS :

S_0 : personne V_0 : véhicule SR_0 : segment_route R_0 : route

(1) Rep_0 : repère E_0 : déplacement IE_0 : intervalle IP_1 : intervalle

IM_1 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$

RELATIONS ÉVÉNEMENTIELLES :

$[IP_1]$ véhicule(S_0, V_0) & $[IP_1]$ conducteur(v0, S_0) & à_roues(V_0) &
E_0.acteur = V_0 & $[IP_1]$ appartient_dim2(S_0, SR_0) &
{ (1) { inf(position_y(SR_0, Rep_0), "0") & lie(Rep_0, R_0) } |
(2) droit(SR_0) } & inclus(SR_0, R_0) & énonciateur(S_0)

Où (1) et (2) indiquent les deux hypothèses envisagées pour *droit*.

10.2 Analyse des résultats

Actuellement cet analyseur est capable de traiter l'ensemble du corpus de textes simplifiés (16 textes construits à partir du corpus réel et présentés, avec les résultats obtenus, dans l'annexe B.2, page XIV). Ces phrases sont non seulement analysées par l'analyseur linguistique mais, pour les phrases composées d'une seule proposition, le résultat de l'analyse est aussi intégré au niveau du sous-système de construction de la scène et donne lieu à la construction d'une représentation visuelle de la scène. Il serait possible, au niveau de l'analyseur linguistique, de traiter des phrases plus variées (avec des verbes différents, par exemple) mais cela nécessiterait de travailler, non seulement au niveau de l'analyseur linguistique mais aussi au niveau du sous-système de construction de la scène, pour établir le lexique correspondant.

Dans l'état actuel de nos travaux, l'analyseur linguistique que nous proposons est partiellement implémenté. En effet, les modules d'analyse des relations spatio-temporelles [Mathet, 1995] et d'identification des entités [Dupont, 1995] sont encore dans une phase d'étude linguistique et leur comportement est donc simulé au niveau de l'analyseur linguistique. Le module d'analyse des relations aspectuo-temporelles [Legendre, 1995] est pratiquement terminé mais n'a pas encore été intégré au reste de l'analyseur. Au niveau de l'analyseur linguistique, le comportement de ce module est donc lui aussi simulé mais, contrairement aux deux précédents, les résultats fournis par la simulation correspondent exactement à ceux donnés par une première implémentation de ce module.

Nous avons utilisé des textes simplifiés d'une part dans le but de limiter la quantité des données lexicales nécessaires, d'autre part dans le but d'étudier des problèmes précis tant au niveau de l'analyseur linguistique qu'au niveau du sous-système de construction de la scène. Au niveau de l'analyseur linguistique, les problèmes que nous avons étudiés concernent non seulement la représentation lexicale des verbes de mouvement (étudiée avec le sous-système de construction de la scène) comme “rouler”, “arriver”, “percuter”, “circuler”, etc., mais aussi la représentation lexicale des adverbes et adjectifs ainsi que l'utilisation de ces données au niveau de l'analyse linguistique. Au niveau du sous-système de construction de la scène, les problèmes étudiés (en plus de ceux liés à la représentation dans le lexique qui sont des problèmes communs au sous-système de construction de la scène et à l'analyseur linguistique) concernent essentiellement la représentation spatiale des procès. Sur l'ensemble de ces textes simplifiés, nous obtenons, au niveau de l'analyseur linguistique, des résultats similaires à ceux donnés en exemple tout au long de cette partie et qui sont donnés dans l'annexe B.2.

La description des procès formant le résultat de l'analyse linguistique est transmise telle quel au sous-système de construction de la scène. Chacune de ces descriptions introduit un événement (qui peut être de type déplacement, choc, etc.) représentant l'action décrite par le texte et donnant une indication "directrice" sur la nature du procès. Lors de la réception des résultats de l'analyseur linguistique, le module de construction de la scène utilise ces événements et effectue, en fonction de leur nature, un certain nombre d'inférences pour préciser et compléter la description. A titre d'exemple, la relation `[IP] appartient_dim2(V, SR)` (qui signifie que l'entité V se trouve située, pendant l'intervalle de temps IP, sur la surface définie par SR) est interprétée par le module de construction de la scène pour obtenir des contraintes géométriques plus spécifiques sur la position du véhicule (il en déduira, par exemple, que le support de déplacement du véhicule V est le segment de route SR). Toutes les descriptions ainsi obtenues sont utilisées en fin de traitement pour générer une visualisation de la scène.

Les parties actuellement réalisées ainsi que les résultats obtenus nous permettent d'affirmer qu'une telle architecture est valide pour effectuer de l'interprétation automatique de textes. Cependant, comme nous l'avons vu tout au long de cette partie, il reste encore beaucoup de points à développer tant au niveau du processus d'analyse en lui même et des heuristiques choisies (traitement de la coordination par exemple) qu'au niveau des données (développement du lexique entre autres). Le but de cette implémentation n'étant pas de mettre en place un système définitif mais une maquette nous permettant d'évaluer la faisabilité d'un tel système, nous considérons que le but a été atteint : les résultats obtenus sont encourageants et permettent d'envisager un développement plus approfondi du système. Bien entendu, lors de ce développement un certain nombre de points pourront (et même devront) être rediscutés (c'est le cas, par exemple, du traitement des mots inconnus au niveau de l'analyse morpho-syntagmatique ou des différentes étapes du fonctionnement de l'analyseur des relations actancielles) mais nous pensons que la structure globale de l'analyseur (communications entre modules et architecture globale de l'analyseur linguistique) ainsi que les principes généraux des différents modules (découpage en segments maximaux sûrs avec reconnaissance des propositions et extraction des rattachements possibles au niveau de l'analyseur morpho-syntagmatique, construction des représentations de procès basée sur le résultat de l'analyse morpho-syntagmatique et sur l'utilisation d'un lexique dans lequel les différents sens associés à un mot sont caractérisés par des schémas de sens rattachés à des constructions syntaxico-sémantiques acceptables) pourront être conservés.

Quatrième partie

Mise en place du système global

Comme nous l'avons vu dans la partie I, le fonctionnement général du système est basé sur le principe suivant : chacun des deux analyseurs effectue son analyse sur la partie du document qui le concerne et envoie le résultat obtenu au sous-système de construction de la scène qui, à partir d'un raisonnement sur ces données et grâce au modèle du monde de la route dont il dispose, va construire une représentation de la scène décrite par l'ensemble du document.

Cependant, les analyses effectuées par les deux sous-systèmes d'analyse ne sont pas totalement fiables et peuvent être cause d'incohérences au moment de la construction de la scène. En effet, d'une part, celles-ci sont effectuées avec très peu connaissance sur le monde de la route et peuvent donc se trouver en contradiction avec ce dernier, d'autre part, comme les analyseurs n'utilisent pas les mêmes représentations des connaissances, ils ne peuvent communiquer entre eux et les deux analyses obtenues peuvent donc différer sur certains points. Afin de pouvoir gérer ces problèmes nous avons dû mettre en place un principe de résolution des incohérences permettant de gérer les incohérences détectées par le sous-système de construction de la scène.

C'est ce principe, ainsi que ses conséquences sur les communications entre les différents sous-système et sur l'implémentation du système global, que nous allons présenter dans cette partie.

Chapitre 11

Principes et implémentation

Avant de donner des détails sur le principe mis en place pour résoudre les incohérences, nous allons tout d'abord rappeler quelques principes du fonctionnement général du système.

11.1 Fonctionnement global

Comme nous l'avons déjà indiqué en section 2.1, nous n'avons pas voulu faire comme dans le système développé par [Abe *et al.*, 1981] ou comme dans PICTION ([Srihari & Burhans, 1994] et [Srihari *et al.*, 1994]) et contraindre l'analyse effectuée par un des analyseurs grâce au résultat du deuxième analyseur à cause de la répercussion possible d'une erreur effectuée dans une des analyses sur l'autre analyse. C'est pour cette raison que nous n'avons pas voulu, alors que cela aurait été possible, établir le modèle initial des attentes du lecteur à partir du résultat de l'analyse du schéma mais nous avons préféré mettre en place, au niveau du sous-système de construction de la scène, un modèle initial "typique" décrivant le contexte d'énonciation "classique" lié à un constat d'accident.

L'inconvénient de ce principe de fonctionnement, comme nous l'avons déjà indiqué en section 2.1, est qu'il peut exister des incohérences entre les résultats fournis par les deux analyseurs. Pour résoudre ce problème nous avons donc mis en place un système de résolution des incohérences qui a pour but (comme nous le verrons section 11.2) de traiter, non seulement les incohérences pouvant exister entre les deux analyses mais aussi celles qui peuvent se produire lors de l'intégration d'une des analyses dans le monde de la route. La gestion des incohérences nécessitant de nombreuses communications entre les trois composants du système global, nous avons dû établir un protocole de communication entre les analyseurs et le sous-système chargé d'effectuer la construction de la scène, protocole qui sera plus amplement détaillé dans la section 11.3.1. Ce système de communications sert, non seulement pour résoudre les incohérences, mais aussi pour transmettre le modèle des attentes du lecteur ainsi que les résultats des diverses analyses.

Le fonctionnement du système global est donc basé sur un fonctionnement en parallèle des deux analyseurs ainsi qu'une interprétation et une comparaison des résultats obtenus au niveau du sous-système de construction de la scène. Pour pouvoir détecter le plus rapidement possible une incohérence entre l'analyse linguistique et le monde de la route (et donc éviter d'effectuer inutilement l'analyse d'une partie du texte), nous avons imposé à l'analyseur linguistique de transmettre régulièrement les résultats obtenus au sous-système chargé d'effectuer la construction de la scène. L'analyseur linguistique procédant phrase par phrase nous avons choisi d'effectuer la transmission des résultats à la fin de chacune des phrases. Lors de la réception d'un résultat intermédiaire, le sous-système chargé d'effectuer la construction de la scène doit le vérifier, l'exploiter et, si aucune incohérence n'a été trouvée, envoyer à l'analyseur linguistique une validation afin que celui-ci puisse continuer son analyse. L'avantage d'effectuer pour chaque phrase une transmission des résultats obtenus est de permettre une meilleure évolution du modèle des attentes du lecteur. En effet, l'intégration, au niveau du sous-système de construction de la scène, des résultats phrase par phrase permet de déterminer les attendus pour la suite de l'analyse du texte et donc le nouveau modèle des attentes du lecteur.

Nous allons maintenant voir les différentes incohérences auxquelles peut se trouver confronté le sous-système de construction de la scène ainsi que les principes élaborés pour les résoudre.

11.2 Les incohérences

Nous ne voulons pas, ici, étudier le cas des incohérences inhérentes à l'une ou l'autre des analyses, celles-ci étant strictement du domaine de chacun des analyseurs et n'influençant pas le fonctionnement de l'ensemble du système. Dans cette partie, nous considérons donc comme incohérence possible, toute différence entre analyse du texte et analyse du schéma ainsi que tout désaccord entre une analyse (du texte ou du schéma) et modèle du monde de la route.

Du fait de l'architecture choisie et des différences de représentation entre les deux analyseurs, une incohérence ne peut être détectée que par le sous-système de construction de la scène. En effet, d'une part, il est le seul à avoir connaissance des résultats des deux analyseurs, d'autre part c'est lui qui détient les informations relatives au monde de la route.

Afin de mettre en place un protocole de résolution de ces incohérences, nous avons répertorié un ensemble de cas possibles en distinguant chacun des grands types d'incohérence : incohérence entre les deux analyses et incohérence avec le monde de la route.

11.2.1 Incohérences entre les deux analyses

L'analyse du schéma ne comportant aucune donnée temporelle, les incohérences entre analyse du schéma et analyse du texte ne peuvent porter que sur les entités et relations entre entités mises en évidence par les deux analyses. Une incohérence portant sur une entité correspond à la détection, par un des deux analyseurs, d'une entité qui n'a pas été détectée par l'autre analyseur. Une incohérence au niveau des relations correspond soit à l'absence d'une relation dans une des deux analyses (un véhicule n'ayant pas été détecté comme étant dans une intersection, par exemple), soit à une différence entre la relation déterminée par un des analyseurs et celle déterminée par l'autre (deux véhicules considérés comme face à face par un des analyseurs mais considérés comme se suivant par l'autre).

Les entités qui, actuellement, sont détectées par les deux analyseurs (et donc susceptibles de générer une incohérence) sont : les véhicules, les routes et les intersections (carrefour simple aussi bien que rond-point). En ce qui concerne les véhicules, les incohérences peuvent porter non seulement sur le nombre des entités détectées mais aussi sur leur nature : un analyseur détecte la présence d'un camion mais l'autre ne détecte que des voitures est une incohérence sur la nature des véhicules. Dans le cas des routes il peut y avoir, non seulement une incohérence sur le nombre d'entités, mais aussi une incohérence sur la constitution de la route, c'est-à-dire sur la nature (courbe ou droite) des différents segments qui la composent. De même, pour les intersections, les incohérences peuvent porter soit sur le nombre d'entités détectées, soit sur la nature (carrefour classique ou rond-point) des entités.

Les relations actuellement déterminées à partir de chacune des analyses sont les relations "véhicule-véhicule" (position relative d'un véhicule par rapport à un autre; ce type de relation n'est pas fourni directement par l'analyseur de schémas mais peut être déduit par le sous-système de construction de la scène à partir des positions et orientations des véhicules), les relations "véhicule-route" (position d'un véhicule par rapport à une route) et les relations "véhicule-intersection" (position d'un véhicule par rapport à une intersection).

Cette liste des incohérences possibles entre les deux analyseurs n'est bien sûr pas exhaustive mais correspond aux capacités actuelles des deux analyseurs. Le fait d'augmenter les capacités d'un des analyseurs (ajouter, par exemple, à l'analyseur de schémas, la reconnaissance de nouveaux objets) entraînera une augmentation des possibilités d'incohérence mais celles-ci porteront toujours sur la détection ou non d'une entité (et de son type) ou d'une relation entre entités. Il est évident, étant donné que le schéma ne représente qu'un instant alors que le texte correspond à l'ensemble du déroulement de l'accident, qu'une différence entre analyse du texte et analyse du schéma n'est pas nécessairement une incohérence. Une des tâches à effectuer lors de la résolution des incohérences sera de déterminer si une différence est bien une incohérence.

11.2.2 Incohérences avec le monde de la route

Les incohérences avec le monde de la route sont détectées lors de l'intégration d'une analyse dans le modèle du monde de la route. Ceci est vrai aussi bien pour l'analyse du texte que pour l'analyse du schéma. En effet, un véhicule détecté par l'analyseur de schémas comme étant hors d'une route est, le plus souvent, une incohérence avec le monde de la route.

Afin de préciser les incohérences qui peuvent exister entre le résultat de l'analyse linguistique et le monde de la route, nous allons maintenant donner quelques exemples.

Soit la phrase suivante :

Je traversais le carrefour sur le passage pour piétons quand un véhicule m'a percuté.

Si l'analyse de cette phrase est effectuée avec un modèle initial des attentes contenant deux véhicules ainsi qu'une relation indiquant que le locuteur est le conducteur d'un véhicule (cas de modèle des attentes utilisé dans le cadre des constats d'accidents), le résultat obtenu aura la signification suivante : "le locuteur traverse le carrefour sur un passage pour piétons tout en étant dans un véhicule". Ce résultat est en contradiction avec le monde de la route car, en général (excepté, à la rigueur, le cas du vélo), un véhicule ne traverse pas sur le passage pour piétons. Dans cet exemple, le fait que le locuteur soit conducteur d'un véhicule (source de l'incohérence) est issu du modèle des attentes. Les incohérences entre analyse linguistique et monde de la route peuvent donc provenir de ce modèle.

Prenons un autre exemple :

Je circulais à bord de ma moto quand un véhicule m'a percuté. Sous le choc j'ai roulé jusque dans le fossé.

L'analyse de la seconde phrase s'effectue avec un modèle des attentes contenant, entre autres, un véhicule dont le conducteur est le locuteur. La présence de ce véhicule entraîne, lors de l'analyse de la seconde phrase, la sélection, comme schéma de sens de *rouler*, du schéma signifiant "se déplacer à l'aide d'un véhicule à roues". Or dans ce texte, le sens le plus vraisemblable serait plutôt : "se déplacer en tournant sur soi-même" et nécessiterait donc de considérer que le choc a éjecté le conducteur de la moto. Le véhicule étant linguistiquement marqué dans la première phrase du texte, l'incohérence entre analyse du texte et monde de la route ne provient pas du modèle initial des attentes du lecteur mais d'un non-dit (le conducteur a été éjecté) non détecté lors de l'analyse de la première phrase et donc non intégré au niveau du modèle des attentes du lecteur utilisé pour effectuer l'analyse de la seconde phrase.

Nous nous trouvons donc confrontés à deux types d'incohérences entre l'analyse linguistique et le monde de la route : celles qui sont issues du modèle initial des attentes du lecteur et celles qui proviennent de non-dits non détectés.

Cette liste des incohérences, comme la précédente, n'est bien sûr pas exhaustive et ce, d'autant plus que, contrairement aux incohérences entre les deux analyses, les incohérences entre analyse linguistique et le monde de la route font non seulement intervenir les entités et les relations qu'elles entretiennent mais aussi les relations temporelles.

11.2.3 Principe de résolution des incohérences

Le principe général de résolution des incohérences est basé sur l'exécution, par un ou plusieurs analyseurs, d'une nouvelle analyse dirigée par des informations venant du sous-système de construction de la scène. En effet, lorsqu'il a repéré une incohérence, le sous-système de construction de la scène ne dispose pas de toutes les informations nécessaires à la modification des analyses mises en cause : il ne connaît pas les conséquences exactes que peut avoir l'ajout ou la suppression d'un élément ou d'une relation. Par conséquent, il doit demander aux analyseurs d'effectuer une nouvelle analyse en la paramétrant : il donne des indications sur le (ou les) élément(s) à supprimer ou à ajouter.

Afin de préciser ce principe général, nous allons le détailler pour chacun des grands types d'incohérence décrits ci-dessus (incohérence entre les deux analyses et incohérence avec le monde de la route).

Cas d'une incohérence avec le monde de la route

Dans le cas d'une incohérence avec le monde de la route, une seule analyse est remise en cause. Le sous-système de construction de la scène doit alors déterminer les éléments susceptibles d'être modifiés pour effectuer une nouvelle analyse. Pour cela, il doit procéder à une étape de vérification des données auprès de l'analyseur mis en cause afin de déterminer les parties de l'analyse à refaire. Une fois les causes de l'incohérence cernées, le sous-système de construction de la scène peut demander à l'analyseur de refaire une analyse en lui donnant les éléments à modifier.

En ce qui concerne l'analyseur de schémas, comme nous avons choisi de ne transmettre au sous-système de construction de la scène que le résultat final de l'analyse, toute incohérence entre analyse du schéma et monde de la route obligera ce dernier à demander à l'analyseur de refaire complètement son analyse. Seul l'analyseur de schémas pourra déterminer, en fonction des modifications apportées, quelle partie de l'analyse (catégorisation des éléments fermés, extraction des routes, etc.) doit être refaite. Dans le cas de l'analyseur linguistique, le problème est différent. En effet, nous avons choisi de transmettre le résultat de cettte analyse non pas en une seule fois mais à la fin de chaque phrase. L'intégration dans le modèle du monde de la route étant effectuée à chaque transmission, le sous-système de construction de la scène peut détecter à partir de quelle phrase (voire à partir de quel procès) l'analyse est à recommencer. Il nous a donc semblé important au niveau de la complexité de l'ensemble du processus de ne refaire que les parties nécessaires de l'analyse. Ceci nous donne donc, au niveau du sous-système de construction de la scène, l'algorithme de résolution d'incohérences suivant :

```
-  demander à l'analyseur mis en cause d'effectuer un certain
   nombre de vérifications sur l'analyse qui vient d'être effectuée
   (les vérifications demandées sont en relation directe avec les
   éléments cause de l'incohérence)
-  déterminer, à partir des réponses obtenues, la partie de
   l'analyse à refaire
-  demander à l'analyseur mis en cause de refaire son analyse en
   tenant compte des modifications imposées
```

Comme nous l'avons vu précédemment, une incohérence entre analyse linguistique et monde de la route peut provenir soit du modèle initial des attentes du lecteur (c'est le cas dans le premier exemple), soit de la non détection d'un non-dit entraînant la "suppression" d'une entité ou d'une relation (comme dans le second exemple où le non-dit relatif au fait que le conducteur a été éjecté entraîne la "suppression" de la relation `conducteur`). Dans le cas où l'incohérence est issue du modèle initial des attentes du lecteur, l'analyse est à reprendre depuis le début avec un modèle initial modifié. Dans les autres cas, elle est à reprendre à partir du procès où elle a été détectée. Pour déterminer à quel moment l'analyse linguistique doit être recommencée, le sous-système de construction de la scène se sert des renseignements fournis par l'analyseur linguistique lors de l'étape de vérification : si l'incohérence porte sur une entité ou une relation non explicitée dans le texte déjà lu, l'analyse est à reprendre depuis le début (une telle entité ou relation est en général issue du modèle initial des attentes du lecteur), si l'incohérence porte sur une entité ou relation mentionnée explicitement dans le texte déjà lu, l'analyse est à reprendre à partir du moment où a été détectée l'incohérence (une telle entité ou relation a été "supprimée" par un non-dit).

Nous allons maintenant donner quelques exemples afin de préciser ces relations entre le fait qu'une entité ou une relation soit explicite et le moment à partir duquel l'analyse doit être recommencée.

Prenons le texte suivant :

Je me déplaçais sur la chaussée. Durant ma marche, j'ai été percuté par un véhicule.

Dans cet exemple, l'entité $\texttt{v}_0$ et les relations `conducteur(`$\texttt{v}_0$`, `$\texttt{s}_0$`)` et `véhicule(`$\texttt{s}_0$`, `$\texttt{v}_0$`)` données par l'analyseur linguistique à l'issue de l'analyse de la première phrase sont implicites et liées au contenu du modèle initial des attentes. La présence de ces entités et relations dans le résultat de l'analyse de la seconde phrase génère une incohérence avec le monde de la route : il est impossible d'être à la fois conducteur d'un véhicule et de marcher[56]. L'incohérence n'est détectée que lors de l'analyse de la seconde phrase mais les relations et entités mises en cause ne doivent pas apparaître dans l'analyse de la première phrase : il faut donc recommencer l'analyse depuis le début en supprimant du modèle initial des attentes les relations et entités causes de l'incohérence.

Regardons le texte suivant :

Je circulais à bord de ma moto quand un véhicule m'a percuté. Sous le choc, j'ai roulé jusqu'au fossé avant de me relever.

Dans ce texte, la relation `conducteur(`$\texttt{v}_0$`, `$\texttt{s}_0$`)` est validée par *Je circulais à bord de ma moto* mais est en contradiction avec le reste du texte (*j'ai roulé jusqu'au fossé avant de me relever*, phrase dans laquelle le sens le plus vraisemblable pour *rouler* est : "rouler sur soi-même"). L'incohérence est détectée lors de l'intégration de l'analyse de la seconde phrase dans le monde de la route mais le fait que la relation impliquée ait été validée auparavant entraîne que l'analyse ne doit être recommencée qu'à partir du procès où a été détectée l'incohérence, c'est-à-dire à partir de la seconde phrase.

Comme nous venons de le voir, il existe un lien entre le fait qu'une entité soit explicite ou implicite et le moment auquel il faut reprendre l'analyse. Le sous-système de construction de la scène ne disposant pas des informations relatives au caractère implicite ou explicite des entités et relations, ces informations doivent lui être transmises par l'analyseur linguistique. Les informations recueillies durant la phase de vérification serviront donc d'une part à déterminer les éléments modifiables de l'analyse, d'autre part à déterminer à partir de quel moment l'analyse doit être recommencée.

56. pris ici dans le sens "se déplacer à l'aide de ses jambes"

Cas d'une incohérence entre les deux analyses

Dans le cas d'une incohérence entre les résultats des deux analyseurs, le principe de fonctionnement du sous-système de construction de la scène est similaire à celui employé précédemment. La seule différence réside dans le fait que ce sous-système doit non seulement déterminer les éléments modifiables de chacune des analyses et le niveau auquel celle-ci doit être reprise mais aussi quelle analyse doit être remise en cause. En effet, prenons le cas où l'analyseur de schémas a détecté un seul véhicule alors que l'analyseur linguistique en a détecté deux. Cette incohérence peut avoir deux causes : soit l'analyseur linguistique a supposé la présence d'un véhicule qui en fait n'existe pas, soit un des véhicules présents sur le schéma n'a pas été détecté par l'analyseur de schémas. La solution de facilité consisterait à recommencer systématiquement les deux analyses mais elle a un inconvénient majeur : elle peut générer l'incohérence inverse à savoir, dans le cadre de notre exemple, la reconnaissance par l'analyseur de schémas de deux véhicules mais la détection, au niveau de l'analyse linguistique, d'un seul véhicule. C'est pour éviter de tels problèmes que nous avons décidé d'effectuer une recherche sur la validité d'une analyse. Cette recherche de l'analyse à remettre en cause est basée, comme celle du moment auquel il faut reprendre l'analyse, sur le caractère implicite ou explicite des entités et relations. Elle est donc effectuée à partir des informations fournies par les deux analyseurs lors de la phase de vérification. Cette phase de vérification est donc effectuée en parallèle auprès des deux analyseurs ce qui nous donne l'algorithme de fonctionnement suivant :

```
-  demander aux deux analyseurs d'effectuer un certain nombre
   de vérifications au niveau des analyses effectuées (les
   vérifications demandées sont en relation directe avec les
   éléments cause de l'incohérence)
-  à partir des réponses obtenues, déterminer l'analyse qui est
   susceptible d'être incorrecte et la partie qui est à refaire
-  demander à l'analyseur mis en cause de refaire son analyse en
   tenant compte des modifications imposées
```

11.3 Implémentation

11.3.1 Le protocole de communication

Le protocole de communication mis en place dans le cadre du système global est le même que celui mis en place dans le cadre de l'analyseur linguistique et qui est décrit section 7.3.1, page 112. Il est donc composé de 4 types de messages différents (les indications, les requêtes, les réponses et les réponses intermédiaires) auxquels sont associées 4 fonctions différentes : les indications permettent de signaler un événement (début d'analyse, nom du fichier où se trouve un résultat d'analyse, ...), les requêtes permettent de poser une question et demandent nécessairement une réponse, les réponses et les réponses intermédiaires permettent d'envoyer une réponse à une requête précise (nous rappelons que les réponses intermédiaires sont utilisées pour envoyer une liste de réponses à une requête donnée et que cette liste doit systématiquement être terminée par une réponse de façon à indiquer la fin de la liste).

11.3.2 Le système global

Pour implémenter l'architecture décrite en partie I, nous devions respecter un certain nombre de contraintes. Tout d'abord une contrainte de génie logiciel : les trois sous-systèmes devaient pouvoir être indépendants de façon à être testés et mis au point (du moins en grande partie) séparément. D'autre part, ces trois sous-systèmes devaient pouvoir fonctionner simultanément. De plus, le sous-système chargé d'effectuer la construction de la scène étant développé en Life[57] [Aït Kaci *et al.*, 1994] un langage de programmation logique (cf. [Pied, 1994] et [Pied *et al.*, 1996]) et les autres, comme nous l'avons vu précédemment en `C++`, nous ne pouvions regrouper tous ces modules en un seul programme “multi-thread” comme cela a été réalisé pour l'analyseur linguistique. Nous avons donc choisi d'avoir trois programmes différents fonctionnant simultanément. Ces trois programmes sont lancés par un quatrième (le contrôleur) qui a, ensuite, pour seule fonction de transmettre les messages de l'expéditeur au destinataire. Les trois sous-systèmes sont donc reliés au contrôleur par un pipe Unix par lequel s'effectuent les communications. Dans le cas présent, cette architecture ne pose

57. langage alliant à la fois les représentations orientées objets et programmation logique (unification basée sur les ψ-termes).

pas les mêmes problèmes que pour l'analyseur linguistique car il n'y a pas de données partagées et donc pas de mémoire partagée à gérer, l'échange de données se faisant par l'intermédiaire de messages et/ou de fichier.

Comme nous pouvons le voir dans la figure 11.1 (les flèches indiquent les flots de données entre les différents programmes), un échange de messages entre les deux analyseurs est tout à fait possible mais n'a jamais lieu du fait de la différence de représentation des données existant entre ces deux sous-systèmes.

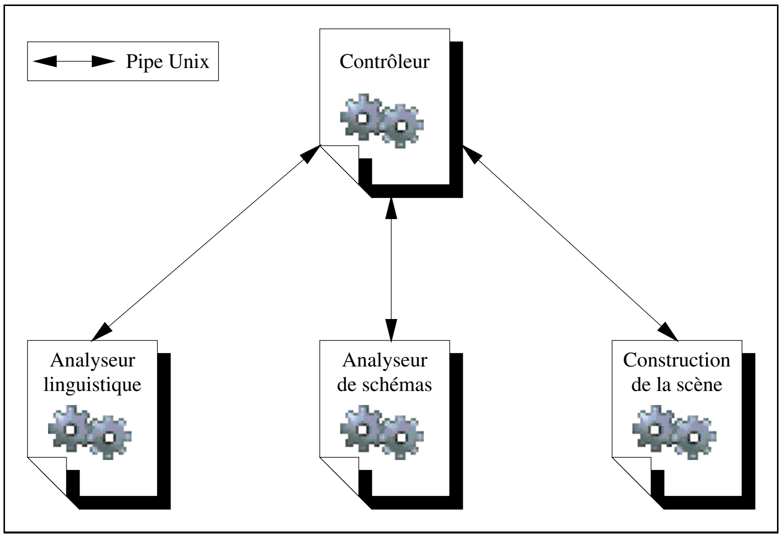


FIG. 11.1 - *Flots de données dans le projet global*

Afin de faciliter l'utilisation du système, une interface graphique a été mise en place (interface dont un aperçu de la fenêtre principale est donnée dans la figure 11.2 et pour laquelle plus de détails sont donnés en annexe C). Celle-ci permet à l'utilisateur de paramétrer et de lancer facilement une analyse ainsi que de visualiser les différentes étapes et résultats fournis. Cette interface (développée à partir des outils réalisés par R. Cazoulat, [Cazoulat, 1996]) est en fait un programme qui vient s'ajouter aux quatre déjà mis en place, qui lance le contrôleur et communique avec les autres sous-systèmes par l'intermédiaire de ce dernier (cf. figure 11.3). Le paramétrage de l'analyse permet, entre autres, de spécifier les fichiers contenant les connaissances à utiliser et ce, aussi bien pour les différents des sous-systèmes que pour les modules de l'analyseur linguistique. Ce système permet de travailler avec des connaissances réduites et de tester l'apport de nouvelles connaissances à l'analyse et ce, de façon conviviale.

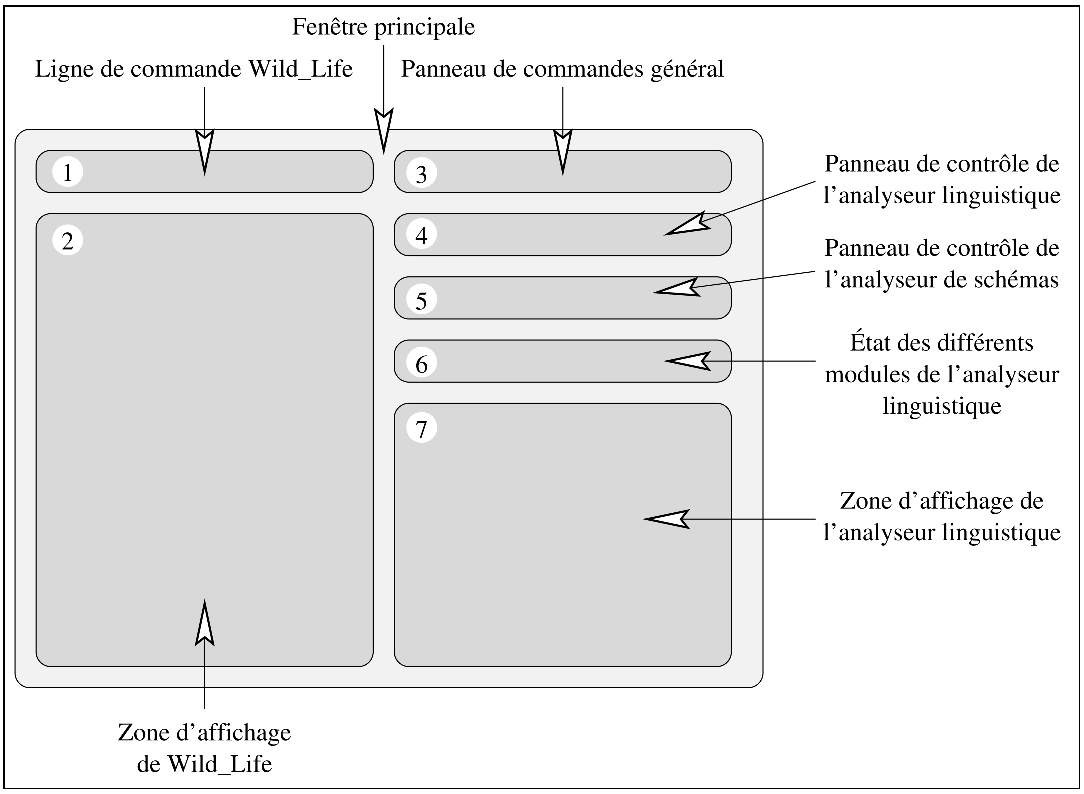


FIG. 11.2 - *La fenêtre principale de l'interface graphique*

Dans ce système, et comme nous l'avons vu précédemment (chapitre 5, page 79 et chapitre 10, page 181), une première maquette a été réalisée pour chacun des analyseurs. Même si ces maquettes sont incomplètes (il reste encore un certain nombre de problèmes qu'elles ne sont pas encore capables de traiter), la partie communication a été mise en place et elles sont capables d'utiliser les réponses fournies par le sous-système de construction de la scène et de répondre à un certain nombre de questions posées par ce dernier. Le sous-système de construction de la scène étant encore dans une phase de développement, nous avons utilisé, pour effectuer nos tests relatifs au problème des incohérences, un système simulant son fonctionnement. Ce sous-système fonctionne cependant correctement dans le cadre d'une analyse sans incohérence.

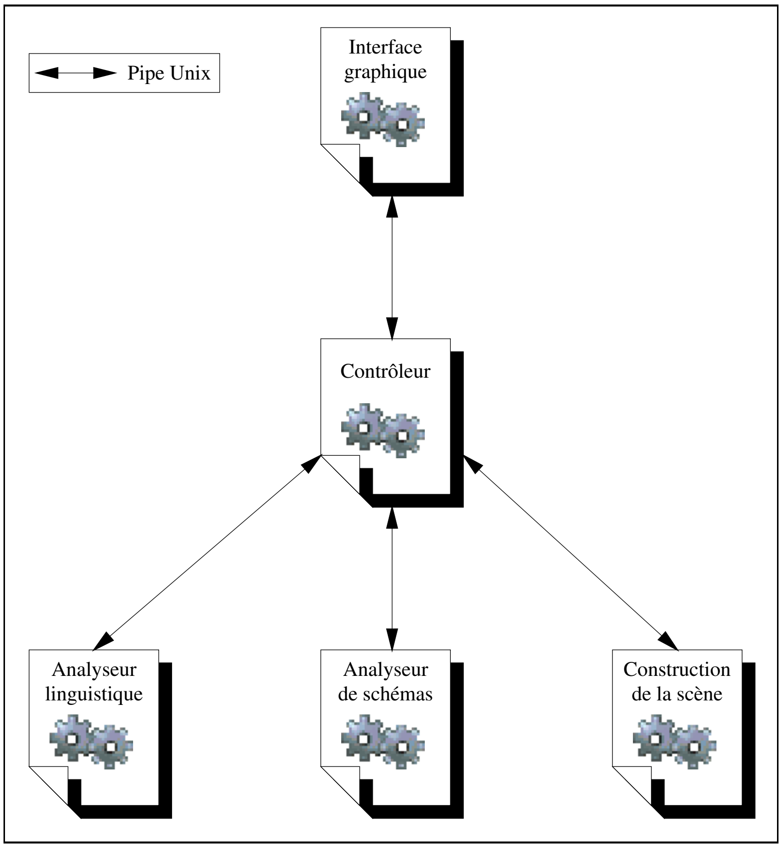


FIG. 11.3 - *Flots de données dans le projet global avec interface graphique*

Chapitre 12

Résultats obtenus

Nous allons maintenant, à travers quelques exemples, voir les communications ayant lieu entre les différents sous-systèmes et ce, dans différentes configurations (analyse avec ou sans incohérence).

12.1 Exemples d'analyse

Dans cette section, nous donnons 3 exemples d'ensemble de messages échangés par les trois sous-systèmes composant le système d'interprétation. Ces différents exemples ont été entièrement fabriqués et ne font pas référence à des constats réels. Notre choix s'est porté sur ces différents exemples d'une part pour leur relative simplicité au niveau de l'analyse du texte, d'autre part parce qu'ils faisaient intervenir des notions ou des exemples déjà vus. Pour les exemples où il existe une incohérence (sections 12.1.2 et 12.1.3), nous indiquons en plus le résultat de l'analyseur linguistique avant détection de l'incohérence ainsi que celui obtenu après une seconde analyse. Les descriptions de procès fournies contiennent non seulement les entités et relations extraites du texte mais aussi les relations contenues dans le modèle des attentes du lecteur où apparaît une des entités extraites du texte ainsi que les entités en relation avec une des entités extraites. En considérant le modèle des attentes du lecteur utilisé dans le cadre des constats d'accidents (2 personnes, 2 véhicules, 1 choc, 2 relations conducteur, 1 relation indiquant que l'énonciateur est un des conducteurs et 1 relation indiquant que les événements se situent dans le passé), le fait de détecter la présence, dans le texte, de l'énonciateur entraîne, au niveau de la représentation du procès la présence de l'entité représentant le conducteur (`s`$_0$ `: personne`), des relations existant entre le conducteur et son véhicule (`véhicule(s`$_0$`, v`$_0$`) et conducteur(v`$_0$`, s`$_0$`)`) et de l'entité représentant le véhicule (`v`$_0$ `: véhicule`).

Dans les tables 12.1, 12.2 et 12.3 nous donnons les échanges réalisés ainsi que les actions qu'ils déclenchent (E correspond à l'expéditeur, D au destinataire et N au numéro de requête ou de réponse). Ces ensembles de messages seront sûrement amenés à évoluer du fait, entre autres, des échanges qui auront lieu avec les modules experts de l'analyseur linguistique. Dans ces tableau, CS correspond au sous-système de construction de la scène, AL à l'analyseur linguistique et AS à l'analyseur de schémas. Dans le cas de l'implémentation réalisée, le seul module syntaxico-sémantique réalisé étant le module d'analyse des relations actancielles, les communications avec l'analyseur linguistique correspondent uniquement à des communications avec ce module.

12.1.1 Analyse sans incohérence

La table 12.1 contient l'ensemble des messages échangés dans le cadre de l'analyse du texte suivant et du schéma de la figure 12.1[58].

Je roulais à vive allure sur la partie droite de la chaussée.

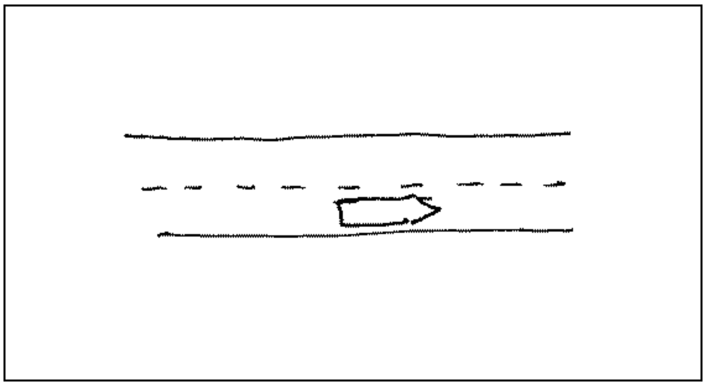


FIG. 12.1 - *Schéma associé à "Je roulais à vive allure sur la partie droite de la chaussée."*

Du fait du résultat de l'analyse linguistique (1 procès où apparaissent, entre autres,

58. Cette phrase correspond à celle qui a été analysée section 8.6 et section 10.1 et dont le résultat est donné page 188. Le schéma est constitué d'une partie d'un schéma réel qui a été modifié pour les besoins de l'exemple.

1 véhicule, 1 personne, 1 route, 1 relation conducteur) et d'autre part du résultat de l'analyse du schéma (1 véhicule, 1 route, 1 relation véhicule-route), l'interprétation des deux analyses ne génère pas d'incohérence.

E	D	Type	N	Message	Explication	Actions déclenchées
AS	CS	indication		lit_schema ("schema")	Envoi par l'analyseur de schémas du résultat de son analyse (résultat contenu dans le fichier donné en paramètre)	CS : intégration de l'analyse, vérification de sa cohérence d'une part avec le monde de la route, d'autre part avec le résultat de l'analyse linguistique (si elle existe)
AS	CS	requête	0	finanalyseschema	Demande de validation de l'analyse effectuée	
AL	CS	requête	0	modeledesattentes	Demande du modèle des attentes	CS : calcul du modèle des attentes du lecteur
CS	AL	réponse	0	modeledesattentes modele	Envoi du modèle initial des attentes du lecteur	AL : début de l'analyse du texte
AL	CS	indication		lit_proces("1_0")	Envoi par l'analyseur linguistique d'un résultat intermédiaire (1er procès de la 1ère phrase)	CS : intégration de l'analyse, vérification de sa cohérence avec le monde de la route et construction du nouveau modèle des attentes
AL	CS	requête	44	finphrase	Demande de validation de l'analyse de la phrase	
CS	AL	réponse	44	modeledesattentes modele1	Validation de l'analyse de la phrase : envoi du modèle des attentes	AL : poursuite de l'analyse
AL	CS	requête	50	fintexte	Demande de validation de l'analyse du texte	CS : vérification de la cohérence de l'analyse du texte avec celle du schéma
CS	AS	réponse	0	ok	Validation analyse	
CS	AL	réponse	50	ok	Validation analyse	

12.1.2 Analyse avec incohérence entre les deux analyses

La table 12.2 contient l'ensemble des messages échangés dans le cadre de l'analyse du texte suivant et du schéma de la figure 12.2a.

Je traversais le carrefour quand un véhicule m'a percuté.

Le résultat de l'analyse du schéma (cf. figure 12.2b) est composé des éléments suivants : 1 véhicule, 1 personne, 2 routes, 1 intersection, 2 relations véhicule-route (le véhicule appartient aux deux routes), 1 relation véhicule-intersection et une relation personne-route. Cette dernière relation ainsi que l'entité de type `personne` ont été rajoutées au niveau du résultat de l'analyseur de schémas, celui-ci étant, pour l'instant, incapable de détecter les piétons.

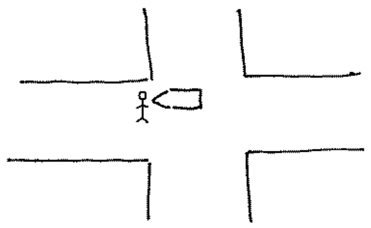
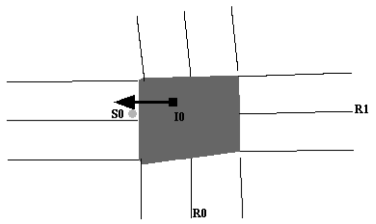
a : Image initiale	b : Résultat de l'analyse du schéma
	

FIG. 12.2 - *Schéma associé à "Je traversais le carrefour quand un véhicule m'a percuté."*

Le résultat de la première analyse du texte effectuée est composé des deux procès suivants :

Numéro : 2_0

TEXTE : ''je traversais le carrefour (quand)''

OBJETS : S_0 : personne V_0 : véhicule E_1 : déplacement C_1 : carrefour
IP_2 : intervalle IM_2 : intervalle IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_2 < IE_0$ & $IM_2 \subset IP_2$

RELATIONS ÉVÉNEMENTIELLES :
[IP_2] véhicule(S_0, V_0) & [IP_2] conducteur(V_0, S_0) &
E_1.acteur = S_0 & [IP_2] appartient_dim2(S_0, C_1)

Numéro : 3_0

TEXTE : ''quand un véhicule m ' a percuté''

OBJETS : V_1 : véhicule S_1 : personne S_0 : personne V_0 : véhicule
E_2 : choc IP_3 : intervalle IM_3 : intervalle IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_2 \mid IM_3$ & $IM_3 = IP_3$ & $IM_3 < IE_0$

RELATIONS ÉVÉNEMENTIELLES :
[IP_3] conducteur(V_1, S_1) & [IP_3] véhicule(S_1, V_1) &
[IP_3] véhicule(S_0, V_0) & [IP_3] conducteur(V_0, S_0) & E_2.acteur = V_1

Comme nous pouvons le voir sur les procès donnés ci-dessus, l'analyseur linguistique, du fait du modèle initial des attentes du lecteur, détermine la présence de deux véhicules (v_0 et v_1) contrairement à l'analyseur de schémas ce qui génère une incohérence portant sur le nombre de véhicules mis en cause dans l'accident. Il existe aussi une différence entre les résultats des deux analyses au niveau du nombre de routes (2 pour l'analyse du schéma, aucune pour l'analyse linguistique). Cependant, cette différence ne génère pas d'incohérence du fait des règles par défaut liées au monde de la route : une intersection entraîne la présence d'au moins deux routes.

E	D	Type	N	Message	Explication	Actions déclenchées
AS	CS	indication		lit_schema ("schema")	Envoi par l'analyseur de schémas du résultat de son analyse (résultat contenu dans le fichier donné en paramètre)	CS : intégration de l'analyse, vérification de sa cohérence d'une part avec le monde de la route, d'autre part avec le résultat de l'analyse linguistique
AS	CS	requête	0	finanalyseschema	Demande de validation de l'analyse effectuée	
AL	CS	requête	0	modeledesattentes	Demande du modèle des attentes	CS : calcul du modèle des attentes du lecteur
CS	AL	réponse	0	modeledesattentes modele	Envoi du modèle initial des attentes du lecteur	AL : début de l'analyse du texte
AL	CS	indication		lit_proces("2_0") lit_proces ("3_0")	Envoi par l'analyseur linguistique d'un résultat intermédiaire (les deux procès de la 1$^{\text{ère}}$ phrase)	CS : intégration de l'analyse, vérification de sa cohérence avec le monde de la route et construction du nouveau modèle des attentes
AL	CS	requête	50	finphrase	Demande de validation de l'analyse de la phrase	
CS	AL	réponse	50	modeledesattentes modele1	Validation de l'analyse de la phrase : envoi du modèle des attentes	AL : poursuite de l'analyse
AL	CS	requête	56	fintexte	Demande de validation de l'analyse du texte	CS : vérification de la cohérence de l'analyse du texte avec celle du schéma
Le module logique a détecté une incohérence entre les deux analyses au niveau du nombre de véhicules : demande de vérification envoyée aux deux analyseurs - l'analyseur de schémas doit rechercher un véhicule en plus - l'analyseur linguistique doit essayer de supprimer un véhicule (pour cela on cherche à déterminer la pertinence de chacun des véhicules afin de savoir lequel on peut supprimer).						
CS	AS	requête	0	verification vehiculesup	Demande à l'analyseur de schémas de rechercher un nouveau véhicule	AS : recherche d'un nouveau véhicule en élargissant la catégorie véhicule
suite page suivante ...						

suite de la page précédente						
E	D	Type	N	Message	Explication	Actions déclenchées
CS	AL	requête	1	verificationentite v_1	Demande de la pertinence de v_1	AL : calcul de la pertinence de v_1
AS	CS	réponse	0	aucun	Aucun véhicule supplémentaire au niveau de l'analyse du schéma	CS : l'analyse du schéma n'a pas lieu, pour l'instant d'être recommencée
AL	CS	réponse	1	verificationentite v_1 1 3_0	Présence de v_1 certaine (procès 3_0)	CS : passer au véhicule suivant
CS	AL	requête	2	verificationentite v_0	Demande de la pertinence de v_0	AL : calcul de la pertinence de v_0
AL	CS	réponse	2	verificationentite v_0 0	Présence de v_0 incertaine	
CS	AL	requête	3	reinitanalyse	Demande de réinitialisation de l'analyse (depuis le début)	AL : réinitialisation de l'analyse
AL	CS	requête	59	modeledesattentes	Demande du modèle des attentes	
CS	AL	réponse	59	modeledesattentes modele2	Envoi d'un nouveau modèle des attentes	
AL	CS	réponse	3	reinitok	Réinitialisation terminée	
CS	AL	réponse	56	analyse	Réponse envoyée à l'analyseur linguistique à sa demande de validation de l'analyse du texte : recommencer l'analyse	AL : Refaire l'analyse en tenant compte des modifications contenues dans le nouveau modèle des attentes
AL	CS	indication		lit_proces("5_0") lit_proces ("6_0")	Envoi par l'analyseur linguistique d'un résultat intermédiaire (les deux procès de la 1ère phrase)	CS : intégration de l'analyse, vérification de sa cohérence avec le monde de la route et construction du nouveau modèle des attentes
AL	CS	requête	116	finphrase	Demande de validation de l'analyse de la phrase	
CS	AL	réponse	116	modeledesattentes modele1	Validation de l'analyse de la phrase : envoi du modèle des attentes	AL : poursuite de l'analyse
AL	CS	requête	122	fintexte	Demande de validation de l'analyse du texte	CS : vérification de la cohérence de l'analyse du texte avec celle du schéma
CS	AS	réponse	0	ok	Validation analyse	
CS	AL	réponse	122	ok	Validation analyse	

TAB. 12.2 - *Communications dans le cadre d'une analyse avec incohérence entre les résultats des deux analyseurs*

Le modèle des attentes transmis à l'analyseur linguistique lors de la réinitialisation de l'analyse contient deux personnes mais ne comporte qu'un seul véhicule et une seule relation de type "conducteur". Il contient aussi une relation de type "véhicule" pour S_0 mais sous la forme suivante : `véhicule(S0, 0)`. Cette relation a pour but d'indiquer à l'analyseur linguistique que la personne S_0 ne peut pas être responsable d'un véhicule (ce n'est donc pas un conducteur). Le fait de faire figurer cette relation dans le modèle des attentes du lecteur influe sur le résultat de l'analyse : l'énonciateur n'étant pas considéré comme un conducteur ni comme responsable d'un véhicule, les relations véhicule et conducteur qui auparavant le mettaient en cause n'apparaîtront pas dans le résultat de la seconde analyse. La seconde analyse effectuée par l'analyseur linguistique donne comme résultat, les procès suivants :

Numéro : 5_0

TEXTE : ''je traversais le carrefour (quand)''

OBJETS :

S_0 : personne E_1 : déplacement C_1 : carrefour IP_5 : intervalle

IM_5 : intervalle IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_5 < IE_0$ & $IM_5 \subset IP_5$

RELATIONS ÉVÉNEMENTIELLES :

E_1.acteur = S_0 & [IP_5] appartient_dim2(S_0, C_1)

Numéro : 6_0

TEXTE : ''quand un véhicule m ' a percuté''

OBJETS :

V_1 : véhicule S_1 : personne S_0 : personne E_2 : choc IP_6 : intervalle

IM_6 : intervalle IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_5 \mid IM_6$ & $IM_6 = IP_6$ & $IM_6 < IE_0$

RELATIONS ÉVÉNEMENTIELLES :

[IP_6] conducteur(V_1, S_1) & [IP_6] véhicule(S_1, V_1) & E_2.acteur = V_1

12.1.3 Analyse avec incohérence entre le résultat de l'analyse linguistique et le modèle du monde

La table 12.3 contient l'ensemble des messages échangés dans le cadre de l'analyse du texte

Je circulais en moto sur la rn 13 quand un véhicule m'a percuté. Sous le choc j'ai roulé sur la chaussée.

Le résultat de la première analyse du texte effectuée est composé des trois procès suivants :

Numéro : 2_0

TEXTE : ``je circulais en moto sur la rn 13 (quand)''

OBJETS :

S_0 : personne V_0 : véhicule E_1 : déplacement R_1 : route IP_2 : intervalle IM_2 : intervalle IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_2 < IE_0$ & $IM_2 \subset IP_2$

RELATIONS ÉVÉNEMENTIELLES :

[IP_2] véhicule(S_0, V_0) & [IP_2] conducteur(V_0, S_0) &
E_1.acteur = S_0 & [IP_2] appartient_dim2(S_0, R_1) &
R_1.nom = ``13'' & $R_1$.nature = ``nationale''

Numéro : 3_0

TEXTE : ``quand un véhicule m ' a percuté''

OBJETS :

V_1 : véhicule S_1 : personne S_0 : personne V_0 : véhicule E_2 : choc IP_3 : intervalle IM_3 : intervalle IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_2 \mid IM_3$ & $IM_3 = IP_3$ & $IM_3 < IE_0$

RELATIONS ÉVÉNEMENTIELLES :

[IP_3] conducteur(V_1, S_1) & [IP_3] véhicule(S_1, V_1) &
[IP_3] véhicule(S_0, V_0) & [IP_3] conducteur(V_0, S_0) & E_2.acteur = V_1

Numéro : 4_0

TEXTE : ``sous le choc j ' ai roulé sur la chaussée''

OBJETS :

E_2 : choc S_0 : personne V_0 : véhicule E_3 : déplacement R_1 : route IP_4 : intervalle IM_4 : intervalle IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_4 < IE_0$ & $IM_4 = IP_4$ & $IM_3 \mid IM_4$

RELATIONS ÉVÉNEMENTIELLES :

[IP_4] véhicule(S_0, V_0) & [IP_4] conducteur(V_0, S_0) & à_roues(V_0) &
équivalent(E_3.acteur, V_0) & [IP_4] appartient_dim2(S_0, R_1)

Comme nous pouvons le voir sur les procès donnés ci-dessus, l'analyseur linguistique, dans le cas du verbe *rouler* considère que le locuteur conduit un véhicule à roues (schéma (2) de *rouler* - cf. table 9.2, page 167). Cependant, le fait qu'il y ait eu un choc et que le véhicule du locuteur soit une moto implique que celui-ci ait été éjecté de son véhicule; il ne peut plus être conducteur de la moto. Ce non-dit non détecté lors de l'analyse de la première phrase entraîne donc une incohérence entre le résultat de l'analyse de la seconde phrase et le monde de la route.

E	D	Type	N	Message	Explication	Actions déclenchées
AL	CS	requête	0	modeledesattentes	Demande du modèle des attentes	CS : calcul du modèle des attentes du lecteur
CS	AL	réponse	0	modeledesattentes modele	Envoi du modèle initial des attentes du lecteur	AL : début de l'analyse du texte
AL	CS	indication		lit_proces("2_0") lit_proces ("3_0")	Envoi par l'analyseur linguistique d'un résultat intermédiaire (les deux procès de la 1ère phrase)	CS : intégration de l'analyse, vérification de sa cohérence avec le monde de la route et construction du nouveau modèle des attentes
AL	CS	requête	50	finphrase	Demande de validation de l'analyse de la phrase	
CS	AL	réponse	50	modeledesattentes modele1	Validation de l'analyse de la phrase : envoi du modèle des attentes	AL : poursuite de l'analyse
AL	CS	indication		lit_proces("4_0")	Envoi par l'analyseur linguistique d'un résultat intermédiaire (2nd phrase)	CS : intégration de l'analyse, vérification de sa cohérence avec le monde de la route et construction du modèle des attentes
AL	CS	requête	90	finphrase	Demande de validation de l'analyse de la phrase	
Le module logique a détecté une incohérence avec lors de l'intégration de l'analyse dans le monde de la route (choc en moto ⇒ conducteur éjecté) : demande de vérification des relations détectées lors de l'analyse linguistique.						
CS	AL	requête	0	verificationentite v_0	Demande de la pertinence de v_0	AL : calcul de la pertinence de v_0
AL	CS	réponse	0	verificationentite v_0 1 2_0	Présence de v_0 certaine (procès 2_0)	CS : étudier les relations mises en cause
CS	AL	requête	1	verificationrelation véhicule(s_0, v_0)	Demande de la pertinence de la relation	AL : recherche de la pertinence de la relation
AL	CS	réponse	1	verificationrelation 1 2_0	Relation certaine (procès 2_0)	CS : passer à la relation suivante
suite page suivante ...						

suite de la page précédente						
E	D	Type	N	Message	Explication	Actions déclenchées
CS	AL	requête	2	verificationrelation conducteur(v_0., s_0)	Demande de la pertinence de la relation	AL : recherche de la pertinence de la relation
AL	CS	réponse	2	verificationrelation 1 2_0	Relation certaine (procès 2_0)	
CS	AL	requête	3	reinitanalysephrase	Relations et véhicule certains ⇒ demande de réinitialisation de l'analyse de la dernière phrase	AL : réinitialisation de l'analyse de la dernière phrase
AL	CS	requête	98	modeledesattentes	Demande du modèle des attentes	
CS	AL	réponse	98	modeledesattentes modele2	Envoi du modèle des attentes	
AL	CS	réponse	3	reinitok	Réinitialisation terminée	
CS	AL	réponse	90	analyse	Réponse envoyée à l'analyseur linguistique à sa demande de validation de l'analyse du texte : recommencer l'analyse	AL : Refaire l'analyse de la dernière phrase en tenant compte des modifications contenues dans le nouveau modèle des attentes
AL	CS	indication		lit_proces("5_0")	Envoi par l'analyseur linguistique d'un résultat intermédiaire (2^{nd} phrase)	CS : intégration de l'analyse, vérification de cohérence avec le monde de la route et construction du modèle des attentes
AL	CS	requête	129	finphrase	Demande de validation de l'analyse de la phrase	
CS	AL	réponse	129	modeledesattentes modelele3	Validation de l'analyse de la phrase : envoi du modèle des attentes	AL : poursuite de l'analyse
AL	CS	requête	135	fintexte	Demande de validation de l'analyse du texte	CS : vérification de la cohérence de l'analyse du texte avec celle du schéma
CS	AL	réponse	135	ok	Validation analyse	

TAB. 12.3 - *Communications dans le cadre d'une analyse avec incohérence entre analyse linguistique et modèle du monde*

Le modèle des attentes transmis à l'analyseur linguistique lors de la réinitialisation de l'analyse contient deux personnes, deux véhicules (la moto est toujours à prendre en compte), deux relations de type "véhicule" (le locuteur, étant éjecté de son véhicule, n'en est plus le conducteur mais en est toujours responsable). Il contient aussi deux relations de type "conducteur" : une pour la relation existant entre v_1 et s_1 et une pour v_0. La relation de type "conducteur" portant sur v_0 est donnée sous la forme :

`conducteur(v0, 0)`. Cette relation a pour but d'indiquer à l'analyseur linguistique que le véhicule v_0 ne peut pas avoir de conducteur. Le fait de faire figurer cette relation dans le modèle des attentes du lecteur influe sur le choix du schéma de sens à prendre en considération pour *rouler* (cf. section 10.1, page 184 - explication des principes du choix de schéma dans le cadre de l'analyse détaillée) lors de la seconde analyse. En effet, comme v_0 ne peut avoir de conducteur, le schéma (2) de *rouler* ne peut s'appliquer (il est impossible de créer le véhicule). Par conséquent, lors de la seconde analyse effectuée par l'analyseur linguistique, seul le schéma (3) de *rouler* sera pris en compte et le sens associé à *rouler* sera celui correspondant à "se déplacer en tournant sur soi-même". Nous obtenons alors comme résultat de l'analyse de la seconde phrase, le procès suivant :

Numéro : 5_0

TEXTE : ``sous le choc j ' ai roulé sur la chaussée''

OBJETS :

E_2 : choc S_0 : personne V_0 : véhicule E_4 : déplacement R_1 : route

IP_5 : intervalle IM_5 : intervalle IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_5 < IE_0$ & $IM_5 = IP_5$ & $IM_3 \mid IM_5$

RELATIONS ÉVÉNEMENTIELLES :

[IP_5] véhicule(S_0, V_0) & E_4.acteur = S_0 &

[IP_5] appartient_dim2(S_0, R_1)

A partir de ces trois exemples, nous pouvons voir la complexité des interactions existant entre les différents sous-systèmes ainsi que l'influence, sur les communications, du principe mis en place pour résoudre les incohérences.

12.2 Analyse des résultats

Les résultats obtenus lors des différents tests effectués sur le système global (analyse sans incohérence, avec incohérence entre les résultats des deux analyseurs ou avec incohérence entre analyse linguistique et modèle du monde) montrent qu'un tel système est réalisable. Il reste cependant beaucoup de travail à faire tant au niveau des analyseurs que du sous-système de construction de la scène ainsi qu'un travail commun à l'analyseur linguistique et au sous-système de construction de la scène : le développement du lexique. Cependant, le fait d'avoir déjà réalisé une première implémentation des analyseurs et de l'ensemble de l'architecture nous permet de poursuivre le développement du système dans de bonnes conditions et de voir les problèmes liés à l'analyse des documents composites que sont les constats amiables d'accidents automobiles.

Conclusion et perspectives

Les objectifs de cette thèse étaient de concevoir l'architecture d'un système de "compréhension limitée" de documents composites et d'en implémenter une première maquette ayant pour fonction d'une part de valider l'architecture définie, d'autre part de montrer la faisabilité d'un tel système. Cette première maquette constitue une première étape nécessaire au développement d'un système plus complet et plus performant. L'architecture que nous avons conçue s'appuie sur la structure même du document (document composé de deux parties distinctes : un texte et un schéma) ainsi que sur une conception particulière de la sémantique : nous considérons qu'un certain nombre d'informations sont apportées par le texte (ou le schéma) et que d'autres doivent être déduites des conditions d'énonciation et des connaissances sur le monde auquel fait référence le document. Cette architecture a donc la particularité de répartir le travail d'interprétation du document en trois sous-systèmes : analyseur de schémas, analyseur linguistique et construction d'une représentation de la scène à partir des données fournies par les deux analyseurs et en s'appuyant sur une description du domaine lié au document et sur les conditions d'énonciation. Le travail réalisé au cours de cette thèse a consisté (en plus de la participation à la conception de l'architecture globale du système) d'une part, à concevoir et à réaliser l'analyseur de schémas ainsi qu'une partie de l'analyseur linguistique, d'autre part à implémenter la structure globale de l'architecture.

Dans notre architecture, le travail de l'analyseur de schémas a été volontairement restreint à une tâche d'extraction des contraintes géométriques portant sur les divers éléments composant le schéma à partir uniquement de connaissances perceptives. Cette limitation du travail de l'analyseur vient de la distinction que nous avons voulu effectuer, à travers cette architecture, entre les informations mentionnées dans le document et celles déduites des conditions d'énonciation ou des connaissances encyclopédiques.

L'analyseur de schémas réalisé (et qui a été testé sur un corpus de 77 schémas) n'est pas capable de traiter tous les problèmes liés aux schémas de constat d'accident (ce n'est qu'une première maquette ayant pour but une étude de faisabilité) mais est capable de déterminer les objets principaux des schémas (les véhicules, les routes et les intersections). L'implémentation réalisée a prouvé qu'il était possible de réaliser un tel analyseur et nous a permis de déterminer un ensemble de problèmes spécifiques à l'analyse de schémas auxquels se trouvera forcément confronté un analyseur grandeur nature.

De même que le travail de l'analyseur de schémas a été restreint à l'extraction de contraintes géométriques, celui de l'analyseur linguistique a été restreint à l'extraction de contraintes sémantiques. Cette extraction s'effectue à l'aide non seulement de connaissances linguistiques mais aussi grâce à une coopération mise en place entre l'analyseur linguistique et le sous-système chargé d'effectuer la construction de la scène. Cette coopération prend deux formes : d'une part, une forme "statique" par l'intermédiaire du lexique (celui-ci contenant à la fois des données linguistiques et des données liées au modèle), d'autre part une forme "dynamique" par l'intermédiaire de ce que nous avons appelé le "modèle des attentes du lecteur" (modèle qui contient l'ensemble des entités et relations que l'on s'attend à trouver dans une phrase en fonction, d'une part des conditions d'énonciation, d'autre part de la lecture des phrases précédentes et qui est mis à jour à la fois par l'analyseur linguistique et par le sous-système chargé d'effectuer la construction de la scène).

L'analyseur linguistique réalisé (et qui a été testé sur un ensemble de 16 textes simplifiés composés à partir de textes du corpus réel) n'a pas pour vocation de traiter tous les problèmes liés à la compréhension de la langue (ce n'est qu'une maquette ayant pour but une étude de faisabilité) mais est capable de traiter correctement les phrases simples sur lesquelles il a été testé. L'implémentation partielle réalisée (seuls trois des modules de cet analyseur ont été implémentés, deux sont encore dans une phase d'étude linguistique) a prouvé qu'il était possible de réaliser un tel analyseur linguistique et a permis de déterminer un principe global de fonctionnement de cet analyseur, principe qui devra être affiné lors de la seconde étape du processus de conception et de réalisation d'un système de compréhension de documents composites.

L'implémentation de l'architecture globale effectuée, bien qu'incomplète, a montré qu'un système de compréhension limitée basé sur une distinction entre informations apportées par le document et informations déduites des conditions d'énonciation et des connaissances sur le monde auquel fait référence le document était réalisable. La mise au point d'un système opérationnel de compréhension limitée demandera cependant encore beaucoup de temps. En effet, il faut non seulement compléter les différents sous-systèmes (les deux analyseurs et aussi le sous-système chargé d'effectuer la construction de la scène) mais aussi développer le lexique utilisé pour pouvoir traiter des textes complets. Du fait de l'importance de ce lexique au niveau de l'analyseur linguistique (et, comme nous l'avons vu, au niveau du résultat fourni par l'analyseur linguistique) ce travail demande beaucoup d'attention et doit être effectué grâce à une collaboration entre concepteurs de l'analyseur linguistique et concepteurs du sous-système de construction de la scène. La suite du développement de ce système consistera donc à développer d'avantage la maquette en tenant compte des résultats obtenus et des difficultés rencontrées au cours de la première étape, de manière à traiter un nombre un peu plus conséquent de documents du corpus.

L'architecture du système que nous venons de présenter dans ce mémoire est destinée à s'adapter à chaque application spécifique pour laquelle elle sera mise en œuvre. Comme nous l'avons indiqué dans l'introduction, nous ne pensons pas qu'il soit possible de construire un système universel capable de traiter n'importe quel type de textes, mais nous pensons que, suivant les corpus, il est possible de mettre en place différents systèmes ayant un certain nombre de caractéristiques en commun.

En ce qui concerne l'analyseur linguistique nous estimons que le passage à d'autres corpus ne devrait pas nécessiter une refonte totale de l'analyseur mais uniquement un certain nombre de réglages ainsi qu'un augmentation de ses capacités. Pour cela, nous nous basons sur l'idée qu'il existe, en sémantique grammaticale, une certaine unité de la langue : les marques aspectuo-temporelles, par exemple, forment un système que l'on peut décrire relativement indépendamment d'un type de corpus donné même si, suivant le type de corpus, certains réglages seront nécessaires du fait de l'emploi massif de tel temps verbal ou de tel adverbe aspectuel. Dans le cadre d'un changement de corpus, l'analyseur linguistique peut être confronté à un changement de type de textes (textes très stéréotypés ou alors plus élaborés que ceux rencontrés lors de l'analyse du corpus de constats d'accidents) et donc à des formes linguistiques nouvelles ou à des emplois nouveaux de formes déjà traitées. Nous estimons que ce changement de type de textes conduira, non pas à une refonte totale du système, mais à un ajout de nouvelles connaissances sans pour autant invalider les connaissances déjà introduites et ce, même si certains réglages seront nécessaires (au niveau des priorités introduites dans les grammaires, par exemple). Nous estimons donc que l'évolution de l'analyseur linguistique se fera par accumulation de connaissances linguistiques au fur et à mesure des passages de corpus différents.

Le cas de l'analyseur de schémas est un peu différent. En effet, un certain nombre d'opérateurs pourront être réutilisés (fermeture de contours, fusion de deux contours, etc.) mais leur ordre d'application devra sans doute être modifié en fonction de la nature des schémas à étudier. De même, la partie "catégorisation des éléments fermés" devra être ajustée aux caractéristiques des schémas. Cependant, le principe même de l'analyseur (découpage de l'image en deux parties distinctes, traitées en parallèle) est, à notre avis, à conserver. Les différents opérateurs développés dans la cadre de ce corpus (ainsi que ceux qui devront être développés pour d'autres corpus) peuvent être vus comme les composantes d'une bibliothèque d'opérateurs utilisables pour toute application ultérieure.

Pour le sous-système de construction de la scène, et du fait des caractéristiques de ce dernier (ce sous-système dépend du domaine étudié), le passage à d'autres corpus (et surtout le passage à d'autres domaines) devrait entraîner un remplacement quasi complet du sous-système avec, entre autres, une reconstruction complète du monde et du lexique. Cependant, la méthodologie acquise lors de la construction du lexique lié à l'analyse des constats d'accidents automobiles pourra être réutilisée lors de la mise en place de nouveaux lexiques et facilitera ce travail.

Afin de vérifier la possible réutilisation de la partie "interprétation du texte" du système que nous avons développé (analyseur linguistique plus sous-système de construction de la scène) nous comptons, dans le cadre d'un projet du GIS[59] Sciences de la Cognition ("Traitements et modalités d'accès aux grands fonds documentaires") et en collaboration avec le LIPN, travailler sur un corpus constitué d'annonces judiciaires et légales extraites du "journal spécial des sociétés françaises par actions" et plus particulièrement sur les annonces de constitution de sociétés. L'objectif de ce travail est d'extraire des textes non seulement des informations permettant de constituer des listes d'adresses ou de produire des statistiques sur les lieux d'implantation, de montant de ressources déclarées. etc. mais aussi des renseignements corrects et à peu près complets sur l'activité de la société. Pour obtenir ces derniers renseignements, une simple analyse de forme ne peut suffire : il faut effectuer une analyse sémantique approfondie de la déclaration de création de la société et plus spécialement de son "objet social" ce dont est capable notre système.

Cette réutilisation de notre architecture pourrait être aussi vérifiée à partir de textes du même type que les nôtres (textes narratifs) mais portant sur un autre domaine : les

59. Groupement d'Intérêt Scientifique.

comptes-rendus médicaux. L'avantage d'utiliser un tel corpus serait (en plus du fait de garder le même type de textes) de pouvoir profiter de l'important travail effectué dans le cadre du projet MENELAS sur l'étude de ce corpus et donc de gagner beaucoup de temps.

Une autre perspective concernant cette architecture serait de l'utiliser dans le cadre du projet du GREYC concernant l'interprétation d'atlas géographiques. L'avantage de ce corpus par rapport aux deux précédents est qu'il est composé à la fois de textes et d'images (qui peuvent être des figures statistiques, des croquis ou des photos) et qu'il nous permettrait de vérifier la validité non seulement de la partie interprétation du texte mais aussi celle de la totalité de l'architecture.

Bibliographie

[Abe *et al.*, 1981] Abe, N., Soga, I. et Tsuji, S. (1981). A plot understanding system on reference to both image and langage. Dans *Proceedings of IJCAI-81*, pages 77–84, Vancouver, Canada.

[Abraham, 1994] Abraham, M. (1994). Formalisation de la sémantique lexicale des verbes *circuler* et *éviter*. *Traitement Automatique des Langues : Approches sémantiques*, 35(1):131–146.

[Abraham *et al.*, 1993] Abraham, M., Cordier, F., Coulon, D., Desclés, J. P., Enjalbert, P., Espéret, E., Estival, D., Flageul, V., Francois, J., Fuchs, C., Gayral, F., Grandemange, P., Jayez, J., Kayser, D., Levy, F., Ligozat, G., Moeschler, J., Oh, H., Richard, J. F., Victorri, B. et Zweigenbaum, P. (1993). Déclaration d'intention pour le prc "sciences cognitives" 1993.

[Abraham *et al.*, 1992] Abraham, M., Coulon, D., Desclés, J. P., Enjalbert, P., Estival, D., Fuchs, C., Gayral, F., Jayez, J., Kayser, D., Levy, F., Ligozat, G., Maire-Reppert, D., Oh, H. et Saint-Dizier, P. (1992). Rapport d'activité juin 1990-juin 1992 du groupe inter-prc "sémantique des langues naturelles". LIPN.

[Abraham, 1995] Abraham, M. Y. (1995). *Analyse sémantico-cognitive des verbes de mouvement et d'activité. Contribution méthodologique à la constitution d'un dictionnaire informatique des verbes.* Thèse de doctorat, École Nationale Supérieure des Télécommunications de Bretagne.

[Adriaens & Small, 1988] Adriaens, G. et Small, S. L. (1988). Word expect parsing revisited in a cognitive science perspective. Dans Small, S., Cottrell, G. et Tanenhaus, M., éditeurs, *Lexical Ambiguity Resolution*, pages 13–43. Morgan Kaufmann.

[Aikins, 1980] Aikins, J. (1980). *Prototypes and productions rules : a knowledge representation for computer consultations.* Stanford University. STAN-CS-80-814 (HPP-80-17).

[Allen, 1983] Allen, J. F. (1983). Maintainning knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

[Antoine, 1991] Antoine, D. (1991). *Interprétation des plans cadastraux français à partir d'un modèle.* Thèse de doctorat, Institut National Polytechnique de Lorraine.

[Ariel, 1990] Ariel, M. (1990). *Accessing noun phrases antecedents.* Routledge, London.

[Asher, 1993] Asher, N. (1993). *Reference to Abstract Objects in Discourse.* Kluwer Academic Publishers.

[Aït Kaci *et al.*, 1994] Aït Kaci, H., Dumant, B., Meyer, R., Podelski, A. et Van Roy, P. (1994). *The wild Life Handbook.* Digital.

[Aurnague *et al.*, 1994] Aurnague, M., Sablayrolles, P. et Jayez, J. (1994). Les informations spatio-temporelles dans les constats d'accidents : représentation du contenu sémantique et raisonnement. *Traitement Automatique des Langues : Approches sémantiques*, 35(1):107–130.

[Becher, 1996] Becher, G. (1996). First order modal temporal logic with generalized intervals. Submitted to the Third International Workshop on Temporal Representation and Reasoning.

[Belaïd & Tombre, 1992] Belaïd, A. et Tombre, K. (1992). Analyse de documents : de l'image à la sémantique. Dans *Actes de CNED'92 : Traitement de l'écriture et des documents.*

[Bley, 1984] Bley, H. (1984). Segmentation and preprocessing of electrical schematics using picture graphs. *Computer Vision, Graphics and Image Processing*, 28:271–288.

[Boatto *et al.*, 1992] Boatto, L., Consorti, V., Del Buono, M., Di Zenzo, S., Eramo, V., Esposito, A., Melcarne, F., Meucci, M., Morelli, A., Mosciatti, M., Scarci, S. et Tucci, M. (1992). An interpretation system for land register maps. *IEEE computer*, pages 23–33.

[Bobrow *et al.*, 1977] Bobrow, D., Kaplan, J., Norman, D. et Winograd, T. (1977). Gus : a frame-driven dialog system. *Artificial Intelligence*, 8:155–173.

[Bobrow & Winograd, 1977] Bobrow, D. et Winograd, T. (1977). An overview of krl, a knowledge representation language. *Cognitive Science*, 1:3–46.

[Bos *et al.*, 1996] Bos, J., Gambäck, B., C., L., Mori, Y., Pinkal, M. et Worm, K. (1996). Compositional semantics in verbmobil. Dans *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, Copenhague.

[Broggi & Bertè, 1995] Broggi, A. et Bertè, S. (1995). Vision-based road detection in automotive systems : A real-time expection-driven approach. *Artificial Intelligence Research*, 3:325–348.

[Cazoulat, 1991] Cazoulat, R. (1991). *Simulation de comportements avec des réseaux de neurones*. Rapport de stage de DEA, Université de Caen.

[Cazoulat, 1996] Cazoulat, R. (1996). *Modélisation et simulation de la dynamique de population d'agents*. Thèse de doctorat, Université de Caen.

[Chomsky, 1957] Chomsky, N. (1957). *Syntactic Structure*. La Haye, Mouton.

[Chomsky, 1969] Chomsky, N. (1969). *Structures Syntaxiques*. Le Seuil, Paris.

[CNED, 1994] (1994). *Troisième Colloque National sur l'Écrit et le document (CNED'94)*, Rouen.

[CNED, 1996] (1996). *Quatrième Colloque National sur l'Écrit et le document (CNED'96)*, Nantes.

[Collin, 1992] Collin, S. (1992). *Interprétation de la cotation des dessins techniques par analyse syntaxique*. Thèse de doctorat, Institut National Polytechnique de Lorraine.

[Coulon & François, 1996] Coulon, D. et François, J. (1996). La gauche et la droite dans un corpus de déclarations d'accidents de la circulation. à paraître - EUROSEM 96.

[Cowie & Lehnert, 1996] Cowie, J. et Lehnert, W. (1996). Information extraction. *Communications of the ACM*, 39(1):80–91.

[Crémilleux, 1996] Crémilleux, B. (1996). A pruning method for decision trees in uncertain domains : Applications in medicine. Dans *proceedings of the workshop Intelligent Data Analysis in Medicine and Pharmacology, ECAI 96*, pages 15–20, Budapest, Hungary.

[Crémilleux & Robert, 1997] Crémilleux, B. et Robert, C. (1997). A theoretical framework for decision trees in uncertain domains : application to medical data sets. proceedings of Sixth European Conference of AI in Medicine Europe (AIME 97), à paraître dans Lecture Note Series in Artificial Intelligence.

[Crouch & Pulman, 1993] Crouch, R. S. et Pulman, S. G. (1993). Time and modality in a natural language interface to a planning system. 63:265–304.

[Descles, 1990a] Descles, J. P. (1990a). Catégories grammaticales et opérations cognitives. Dans *Presses Universitaires de Vincennes*, volume 1, pages 33–53. Paris.

[Descles, 1990b] Descles, J. P. (1990b). *Langages applicatifs, langues naturelles et cognition*. Hermès, Paris.

[Desclés, 1991] Desclés, J. P. (1991). Archétypes cognitifs et types de procès. Dans *Travaux de linguistique et de philologie*, numéro XXIX, pages 171–195. Klincksieck, Paris.

[Desclés *et al.*, 1994] Desclés, J. P., Maire-Reppert, D., Oh, H. et Berri, J. (1994). Traitement informatique du temps et de l'aspect pour l'analyse de textes. *Traitement Automatique des Langues : Approches sémantiques*, 35(1):83–105.

[Dorna & Emele, 1996] Dorna, M. et Emele, M. (1996). Semantic-based transfer. Dans *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, Copenhague.

[Dupont, 1995] Dupont, M. (1995). *En TAL : le calcul de la référence nécessite la gestion d'un modèle des attentes du lecteur*. Rapport de stage de DEA, Université de Caen.

[Dupont, 1996] Dupont, M. (1996). Le modèle des attentes du lecteur. Dans *Actes de ILN'96*, Nantes.

[Enjalbert & Victorri, 1994] Enjalbert, P. et Victorri, B. (1994). Du langage au modèle. *Traitement Automatique des Langues : Approches sémantiques*, 35(1):37–64.

[Erman *et al.*, 1980] Erman, L. D., Hayes-Roth, F., Lesser, V. et Raj Reddy, D. (1980). The hearsay-ii speech understanding system : integration knowledge to resolve uncertainly. *Computing surveys*, 12(2):213–253.

[Estival & Gayral, 1994] Estival, D. et Gayral, F. (1994). Contexte et inférence. *Traitement Automatique des Langues : Approches sémantiques*, 35(1):19–36.

[Etemadi *et al.*, 1991] Etemadi, A., Schmidt, J. P., Matas, G., Illingworth, J. et Kittler, J. (1991). Low_level grouping of straight line segments. Dans *British Machine Vision Conference*.

[Fahns *et al.*, 1988] Fahns, C. S., Wang, J. F. et Lee, J. Y. (1988). A topology-based component extractor for understanding electronic circuit diagrams. *Computer Vision, Graphics and Image Processing*, 44:119–138.

[Fauconnier, 1984] Fauconnier, G. (1984). *Les espaces mentaux*. Éditions de minuit, Paris.

[Fraczak, 1996] Fraczak, L. (1996). Modélisation de la description d'itinéraires : aspects linguistique et conceptuel. Dans *Actes de CLIM'96*, pages 73–79, Montréal.

[François, 1996] François, J. (1996). Le cadrage cognitif des prédications de contact dans un corpus de déclarations d'accidents de la route : effets du contexte et du co-texte. Dans *Actes du colloque de l'ELSAP*.

[Fuchs *et al.*, 1993] Fuchs, C., Danlos, L., Lacheret-Dujour, A. et Victorri, B. (1993). *Linguistique et Traitements Automatiques des Langues*. Hachette Supérieur. Hachette, Paris.

[Fuchs, 1996] Fuchs, C. (1996). *Les ambiguïtés du français*. Coll. L'essentiel français. Ophrys.

[Gaio, 1994] Gaio, M. (1994). *Une architecture d'aide à la construction de croquis d'interprétation géographique : contribution à l'interprétation des images aériennes*. Thèse de doctorat, Université de Caen.

[Gaio *et al.*, 1994] Gaio, M., Madelaine, J., Poirier, C. et Victorri, B. (1994). Identification des structures linéaires prégnantes dans des images télédétectées. Dans *Actes EGIS/MARI'94*.

[Garnesson, 1991] Garnesson, P. (1991). *Messie: un système d'analyse de scène application à l'imagerie aérienne*. Thèse de doctorat, Université de Nice.

[Gayral *et al.*, 1994] Gayral, F., Grandemange, P., Kayser, D. et Lévy, F. (1994). Interprétation des constats d'accidents : représenter le réel et le potentiel. volume 35, pages 65–82.

[Georgiev, 1991] Georgiev, H. (1991). English algorithmic grammar. *Applied Computer Translation*, 1(3):29–48. Sigma Press, U.K.

[Gnu, 1992] (1992). User's guide to the gnu `c++` library.

[Gorcy, 1996] Gorcy, G. (1996). Le trésor de la langue française (tlf), trente ans après, bilan et perspectives. *Études de linguistique appliquée*, (85/86).

[Gosselin, 1992] Gosselin, L. (1992). Une grammaire calculatoire du temps et de l'aspect en français. Rapport interne de l'ELSAP, Université de Caen.

[Gosselin, 1993] Gosselin, L. (1993). *Subjectivité et temporalité : une approche cognitive.* Numéro 9 dans Elseneur. Presses Universitaires de Caen.

[Gosselin, 1995] Gosselin, L. (1995). Sémantique de la temporalité en français, un modèle calculatoire et cognitif. Habilitation à diriger des recherches. Université de Caen.

[Haralick & Queeney, 1982] Haralick, M. et Queeney, D. (1982). Understanding engineering drawings. *Computer Vision, Graphics and Image Processing*, 20:244–258.

[Haton, 1989] Haton, J. P. (1989). Panorama des systèmes multi-agents. Dans *Architectures avancées pour l'IA*, Paris. EC2.

[Herzog *et al.*, 1993] Herzog, G., Maaß, W. et Wazinski, P. (1993). Vitra guide : Utilisation du langage naturel et de représentations graphiques pour la description d'itinéraires. Dans *Actes du Colloque Interdisciplinaire du Comité National "Images et Langages : Multimodalité et Modélisation Cognitive"*, Paris.

[Herzog & Rollinger, 1991] Herzog, O. et Rollinger, C. R. (1991). Text understanding in lilog. Dans *Lectures Notes in Artificial Intelligence*, numéro 546. Springer Verlag.

[Hwang & Schubert, 1992] Hwang, C. H. et Schubert, L. (1992). Tense trees as the "fine structure" of discourse. Dans *Actes 30^{me} Annual Meeting of the Association for Computational Linguistics*, pages 232–240, Newark.

[Johnson-Laird, 1983] Johnson-Laird, P. N. (1983). *Mental models*. Harvard University Press, Cambridge.

[Kamp, 1981] Kamp, H. (1981). A theory of truth and semantic representation. Dans *Formal methods in the study of language*. Groenendijk and Janssen and Stokhof, Amsterdam, Mathematical centre Tracs.

[Kamp & Reyle, 199] Kamp, H. et Reyle, U. (199). *From Discourse to Logic : An Introduction to Modeltheoretic Semantics of Natural Language*. Kluwer Academic Publishers.

[Kaplan, 1973] Kaplan, R. (1973). A general syntactic processor. *Natural Language Processing*, pages 193–241. Randall Rustin (ed.), Algorithmics Press, New York.

[Katsuhiro & Shogo, 1995] Katsuhiro, T. et Shogo, N. (1995). Implementation and refinement of decision trees using neural networks for hybrid knowledge acquisition. *Artificial Intelligence in Engineering*, 9:265–275.

[Kay, 1973] Kay, M. (1973). The mind system. *Natural Language Processing*, pages 155–18. Randall Rustin (ed.), Algorithmics Press, New York.

[Klein, 1994] Klein, W. (1994). *Time in Language*. Routeledge, London/New York.

[Kripke, 1963] Kripke, S. A. (1963). Semantic analysis of modal logic l : Normal propositional calculi. *Zeit. Math. Logik Grund. math*, 9:67–96.

[Krönert *et al.*, 1987] Krönert, G., Lauber, G., Lörscher, J., Maynieux, E., Postl, W., Schneider, U., Seisen, S. et Tombre, K. (1987). Document editing and entry based on the standardised office document architecture. Dans *ESPRIT'86 : Results and Achievements*, Bruxelles.

[Legendre, 1995] Legendre, Y. (1995). *Conception d'un module d'analyse linguistique des relations aspectuo-temporelles*. Rapport de stage de DEA, Université de Caen.

[Lenat & Guha, 1990] Lenat, D. et Guha, R. (1990). *Building large knowledge based systems : representation and inference in the CYC project*. Reading, Addison Wesley.

[Mariani, 1982] Mariani, J. (1982). *Esope : Un système de compréhension de la parole continue*. Thèse de doctorat, Université de Paris 6.

[Martin, 1983] Martin, R. (1983). *Pour une logique du sens.* Linguistique Nouvelle. P.U.F., Paris. Réédition revue et corrigée, 1992.

[Mathet, 1995] Mathet, Y. (1995). *Sémantique spatiale.* Rapport de stage de DEA, Université de Caen.

[McDermott, 1982] McDermott, D. (1982). A temporal logic for reasoning about processes and plans. *Cognitive science*, 6:101–155.

[Mel'čuk *et al.*, 1995] Mel'čuk, I., Clas, A. et Polguère, A. (1995). *Introduction à la lexicologie explicative et combinatoire.* Coll. Champs linguistiques. Éditions Duculot.

[MENELAS, 1995] (1995). *MENELAS : An Acces System Records using Natural Language.* P. Zweigenbaum ed., AIM Project A2023, Final Report.

[Mercier *et al.*, 1977] Mercier, G., Quinton, P. et Vives, R. (1977). Dialogue homme-machine avec keal. *Recherches acoustiques*, 4:15–22.

[MUC, 1991] (1991). *Proceedings of the 3d Message Understanding Conference (MUC-3)*, San Diego, California. DARPA, Morgan Kauffman.

[MUC, 1992] (1992). *Proceedings of the 4th Message Understanding Conference (MUC-4)*, McLean, Virginia. DARPA, Morgan Kauffman.

[Nagasamy & Langrana, 1990] Nagasamy, V. et Langrana, N. A. (1990). Engineering drawing processing and vectorisation system. *Computer Vision, Graphics and Image Processing*, 49:379–397.

[Palmer *et al.*, 1993] Palmer, M. S., Passonneau, R. J., Weir, C. et Finin, T. (1993). The kernel text understanding system. *Artificial Intelligence*, 63:17–68.

[Picand, 1993] Picand, Y. (1993). *Analyse morpho-syntaxique de la phrase française.* Rapport de stage de DEA, Université de Caen.

[Pied, 1994] Pied, F. (1994). *Modélisation logique du monde de la route.* Rapport de stage de DEA, Université de Caen.

[Pied *et al.*, 1996] Pied, F., Poirier, C., Enjalbert, P. et Victorri, B. (1996). From langage to model. Dans *Proceedings Workshop ECAI-96 on Corpus-Oriented Semantic Analysis*, pages 51–56.

[Pierrel, 1975] Pierrel, J. M. (1975). *Contribution à la compréhension automatique du discours continu.* Thèse de doctorat, Université de Nancy 1.

[Pierrel, 1981] Pierrel, J. M. (1981). *Étude et mise en oeuvre de contraintes linguistiques en compréhension automatique du discours continu.* Thèse d'état, Université de Nancy 1.

[Pierrel & Sabah, 1991] Pierrel, J. M. et Sabah, G. (1991). Dialogue en langage naturel écrit ou oral : bilan des approches du crin et du limsi. Dans *Actes des secondes journées du PRC Communication homme-machine*, Paris. EC2.

[Poirier & Pied, 1996a] Poirier, C. et Pied, F. (1996a). Analyse de constats amiables d'accident automobile. Dans *Actes de CLIM'96*, pages 235–243, Montréal.

[Poirier & Pied, 1996b] Poirier, C. et Pied, F. (1996b). Un système d'analyse de constats amiables d'accident automobile. Dans *Actes des Rencontres Nationales des Jeunes Chercheurs en Intelligence Artificielle (RJC-IA'96)*, page 273. Présenté sous forme de poster.

[Poirier, 1993] Poirier, C. (1993). *Interprétation de schémas (constats d'accidents fournis par la MAIF).* Rapport de stage de DEA, Université de Caen.

[Poirier, 1994] Poirier, C. (1994). Analyse de schémas de constats amiables. Dans *Actes de CNED'94*, pages 175–184, Rouen.

[Postaire, 1987] Postaire, J. G. (1987). *De l'image à la décision.* Bordas, Paris.

[Quinlan, 1986] Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1):81–106.

[Quinlan, 1993] Quinlan, J. R. (1993). *Programs for machine learning.* Morgan Kaufmann, San Mateo, CA.

[Rady, 1983] Rady, M. (1983). *L'ambiguïté du langage naturel est-il la source du non déterminisme des procédures de traitement ?* Thèse de doctorat, Université Pierre et Marie Curie.

[Ramsay, 1985] Ramsay, A. (1985). Effective parsing with generalized phrase structure grammar. Dans *ACL Proceedings, Second European Conference*, pages 57–61.

[Reyle, 1993] Reyle, U. (1993). Dealing with ambiguities by underspecification : Construction, representation and deduction. *Journal of Semantics*, 10(2):123–179.

[Roussanaly & Pierrel, 1992] Roussanaly, A. et Pierrel, J. M. (1992). Dialogue oral homme-machine en langage naturel: le projet dial. *Technique et science informatiques*, 11(2):45–91.

[Sabah, 1989] Sabah, G. (1989). *L'intelligence artifielle et le langage : processus de compréhension*, volume 1. Hermès, 2^{nd} édition.

[Sabah, 1990a] Sabah, G. (1990a). Caramel : un système multi-experts pour le traitement automatique des langues. *Modèles linguistiques*, XII(1):95–118. Lille.

[Sabah, 1990b] Sabah, G. (1990b). *L'intelligence artifielle et le langage : représentation des connaissances*, volume 2. Hermès.

[Sablayrolles, 1993] Sablayrolles, P. (1993). The semantics of motion in french expressed in a theory of space-time. Dans *Proceedings Workshop Donostia-Toulouse on Semantics and pragmatics of Natural Langage : Logical and Computationnal Aspects*, pages 127–156.

[Sablayrolles, 1995] Sablayrolles, P. (1995). *Sémantique formelle de l'expression du mouvement. De la sémantique lexicale au calcul de la structure du discours en français.* Thèse de doctorat, Institut de Recherche en Informatique de Toulouse, Université Paul Sabatier.

[Seligman, 1985] Seligman, L. (1985). *Intégration de la syntaxe, de la sémantique et de la pragmatique dans un analyseur de textes.* Thèse de doctorat, Université Pierre et Marie Curie.

[Simon, 1969] Simon, H. (1969). *The science of artificial.* MIT Press, Cambridge.

[Small, 1980] Small, S. L. (1980). *Word Expert Parsing : A Theory of Distributed Word-Based Natural Language Understanding.* Thèse de doctorat, Dept. of Computer Science, University of Maryland.

[Srihari & Burhans, 1994] Srihari, R. K. et Burhans, D. T. (1994). Visual semantics : Extracting visual information from text accompanying pictures. Dans *Proceedings of AAAI'94*, Seattle.

[Srihari *et al.*, 1994] Srihari, R. K., Chopra, R., Burhans, D., Venkataraman, M. et Govindaraju, V. (1994). Use of collateral text in image interpretation. Dans *Proceedings of the ARPA Image Understanding Workshop*, Monterey.

[Stroustrup, 1989] Stroustrup, B. (1989). *Le langage C++*. InterEditions.

[Sun, 1993] (1993). Sunos5.2 guide to multi-thread programming. Part n 801-3176-10.

[TAL, 1994] (1994). *Approches sémantiques*, volume 35, n°1 de *Traitement Automatique des Langues*.

[Tanaka *et al.*, 1995] Tanaka, M., Aoyama, N., Sugiura, A. et Koseki, Y. (1995). Integration of multiple knowledge representation for classification problems. *Artificial Intelligence in Engineering*, 9:243–251.

[TLF, 1971] (1971). *Trésor de la Langue Française : dictionnaire la la langue du XIXe et du XXe siècle*. CNRS, Institut natinal de la langue française, Éditions Gallimard. dictionnaire en 16 Volumes.

[Tombre, 1987] Tombre, K. (1987). *La saisie automatisée de documents composites : reconnaissance, codage et interprétation des parties graphiques*. Thèse de doctorat, Institut National Polytechnique de Lorraine.

[Van Dijk & Kintsch, 1983] Van Dijk, T. A. et Kintsch, W. (1983). *Strategies of discourse comprehension*. Academic Press, New York.

[Vandeloise, 1986] Vandeloise, C. (1986). *L'espace en français : sémantique des prépositions spatiales*. Seuil, Paris.

[Vaxivière, 1995] Vaxivière, P. (1995). *Interprétation de dessins techniques mécaniques*. Thèse de doctorat, Institut National Polytechnique de Lorraine.

[Vergne, 1990] Vergne, J. (1990). A parser without a dictionnary as a tool for research into french syntax. Dans *Proceedings Colin90 Intenational Conference on Computational Linguistics*.

[Vergne, 1994] Vergne, J. (1994). A non-recursive sentence segmentation, applied to parsing of linear complexity in time. Dans *Proceedings NeMLaP 94 (International Conference on New Methods in Language Processing)*, Manchester.

[Vergne, 1995] Vergne, J. (1995). Esquisse d'une syntaxe des langues concrètes. *Les cahiers du GREYC*, 11. Université de Caen.

[Victorri & Fuchs, 1996] Victorri, B. et Fuchs, C. (1996). *La polysémie*. Hermès, Paris.

[Victorri *et al.*, 1992] Victorri, B., Thomazo, L., Boyreau, G., Madelaine, J., Coulon, J. F., Le Crosnier, H. et Girollet, D. (1992). L'anté-serveur : une interface intelligente avec l'univers documentaire. Dans *Actes du colloque Informatique 92*, Montpellier. EC2.

[Wirén, 1992] Wirén, M. (1992). Studies in incremental natural-language analysis. Dissertation n°292. Linköping Studies in Science and Technology, University of Linköping, Suède.

[Wolf & Woods, 1980] Wolf, J. et Woods, W. A. (1980). The hwim speech understanding system. *Prentice-Hall*, pages 316–339.

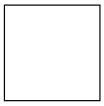
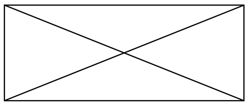
Annexes

Annexe A

L'analyseur de schémas

A.1 Types de dessins de véhicules

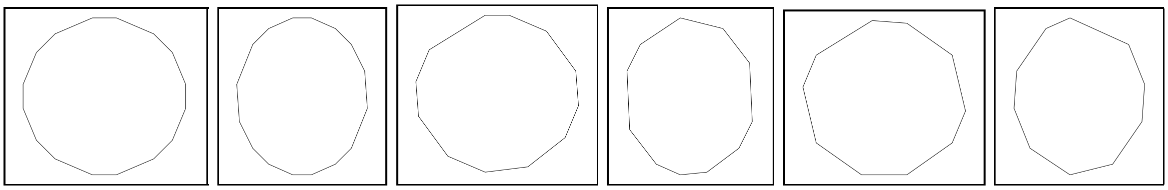
L'ensemble des représentations de véhicules rencontrées lors de l'enquête effectuée au GREYC est donné dans le tableau suivant (143 dessins effectués par 16 scripteurs différents). Dans ce tableau, on remarque que les dessins peuvent être divisés en deux groupes quasiment égaux : les véhicules "non orientés" (51 %), c'est à dire les véhicules dont l'orientation n'est pas graphiquement mentionnée, et les véhicules "orientés" (49 %). C'est la présence d'un grand nombre de véhicules non orientés dans les dessins et la nécessité de ne transmettre que des données certaines qui nous a conduit à avoir deux classes distinctes (la classe des rectangles et celle des véhicules) pour représenter les véhicules. Parmi les véhicules orientés, seuls ceux où l'orientation est représentée par une flèche ne sont pas détectés comme tels du fait de la non reconnaissance des flèches. Le fait d'ajouter à l'analyseur de schémas un module de reconnaissance des flèches nous permettrait sans doute, en utilisant la collaboration entre les modules 2 et 3 de l'analyseur, de pouvoir classer correctement ce type de véhicule.

	Type de dessin	Pourcentage de véhicules de ce type
véhicules non orientés		45 %
		3 %
		3 %
véhicules orientés		12 %
		17 %
		20 %

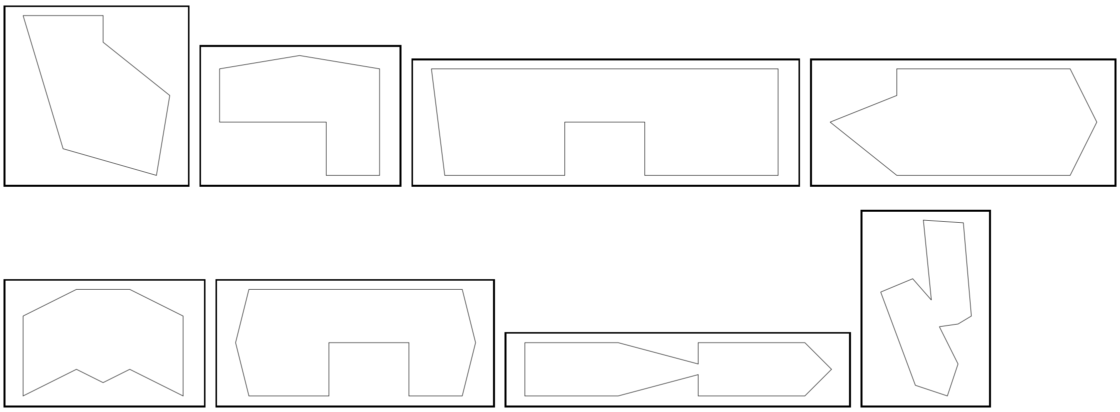
A.2 Une partie de l'échantillon d'apprentissage

Dans cette annexe, nous ne donnons qu'une partie des éléments formant l'échantillon d'apprentissage : uniquement les éléments que nous avons jugés les plus pertinents.

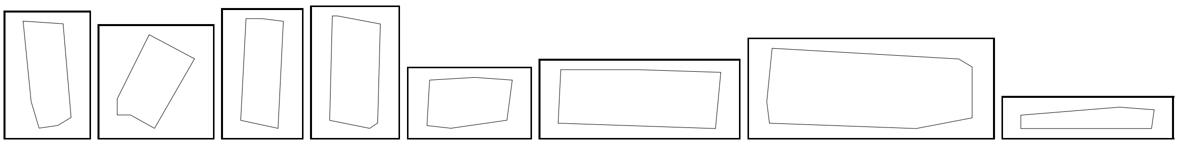
Les éléments de la classe "cercle"

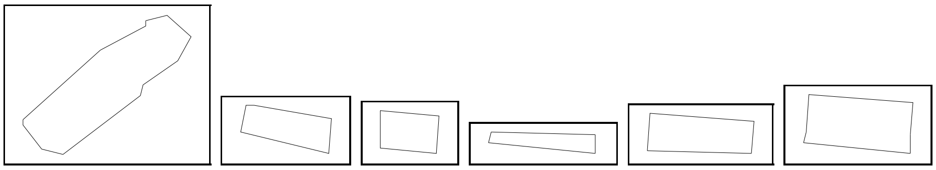


Les éléments de la classe "inconnu"

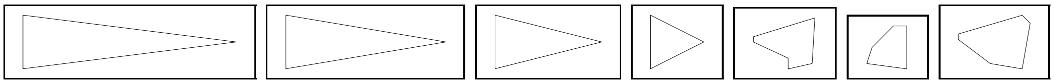


Les éléments de la classe "rectangle"

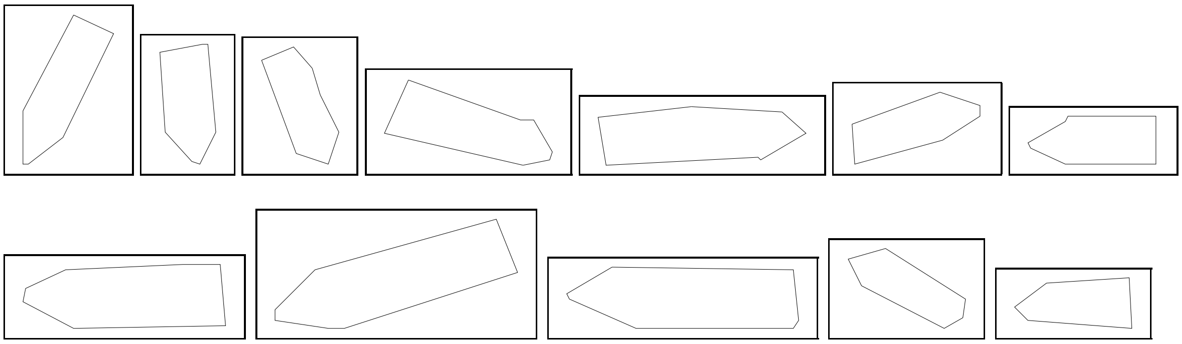




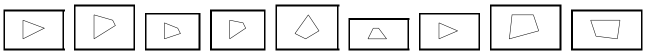
Les éléments de la classe “triangle”



Les éléments de la classe “véhicule”



Les éléments de la classe “petit”



A.3 Arbre de décision utilisé pour la catégorisation des éléments fermés

L'arbre suivant (arbre de décision déterminé par C4.5 à partir d'un échantillon de 293 éléments fermés qui sont soit extraits des schémas des corpus 1 et 2, soit des “éléments types"), nous donne une visualisation de la classification effectuée (les nombres entre parenthèses correspondent aux nombres d'éléments de l'échantillon ayant servi à établir l'arbre).

- surface ≤ 0.0009 : *petit* (81.0)
- surface > 0.0009 :
  - compacité > 0.818675 : *cercle* (13.0)
  - compacité ≤ 0.818675 :
    - rapport ≤ 0.712164 :
      - nombre de segments = 2: *rectangle* (1.0)
      - nombre de segments = 3: *triangle* (10.0)
      - nombre de segments = 4: *triangle* (10.0)
      - nombre de segments = 5: *triangle* (0.0)
      - nombre de segments = 7: *inconnu* (1.0)
      - nombre de segments = 8: *inconnu* (1.0)
      - nombre de segments = 9: *inconnu* (2.0)
      - nombre de segments = 10: *inconnu* (6.0)
      - nombre de segments = 6:
        - rapport ≤ 0.662353 : *inconnu* (2.0)
        - rapport > 0.662353 : *rectangle* (2.0)
    - rapport > 0.712164 :
      - profil gauche ≤ 0.2 : *rectangle* (43.0)
      - profil gauche > 0.2 :
        - profil gauche ≤ 0.7 :
          - nombre de segments = 2: *véhicule* (0.0)
          - nombre de segments = 3: *véhicule* (2.0)
          - nombre de segments = 6: *véhicule* (8.0)
          - nombre de segments = 8: *véhicule* (1.0)
          - nombre de segments = 9: *inconnu* (1.0)
          - nombre de segments = 10: *inconnu* (4.0)
          - nombre de segments = 4:
            - compacité ≤ 0.486504 : *véhicule* (4.0)
            - compacité > 0.486504 :
              - profil droit ≤ 0.81 : *rectangle* (14.0)
              - profil droit > 0.81 :
                - rapport ≤ 0.84866 : *véhicule* (5.0)

| | | | | | | | rapport > 0.84866 : *rectangle* (3.0)
| | | | | nombre de segments = 5:
| | | | | | rapport ≤ 0.861136 : *véhicule* (27.0)
| | | | | | rapport > 0.861136 :
| | | | | | | profil droit $\leq$ -0.5 : *rectangle* (2.0)
| | | | | | | profil droit $>$ -0.5 : *véhicule* (7.0)
| | | | | nombre de segments = 7:
| | | | | | compacité ≤ 0.673596 : *inconnu* (2.0)
| | | | | | compacité > 0.673596 : *véhicule* (5.0)
| | | | profil gauche > 0.7 :
| | | | | compacité > 0.420878 : *rectangle* (30.0)
| | | | | compacité ≤ 0.420878 :
| | | | | | compacité ≤ 0.257297 : *rectangle* (4.0)
| | | | | | compacité > 0.257297 : *inconnu* (2.0)

Annexe B
L'analyseur linguistique

B.1 Corpus de textes réels

Texte A1

Me rendant à Beaumont sur Oise depuis Cergy, je me suis retrouvée à un carrefour juste après la sortie Beaumont sur Oise. J'étais à un stop avec 2 voitures devant moi tournant à droite vers Mours. Alors que la première voiture passait ce stop je fis mon contrôle à gauche et je démarrais mais je percutais la deuxième voiture qui n'avait pas encore passé le stop.

Texte A2

Voulant dépasser un semi-remorque clignotant à droite, ce dernier tourna à gauche m'obligeant à braquer à gauche pour l'éviter. La voiture a dérapé sur la chaussée mouillée et a percuté un trottoir puis un mur de clôture en face. Le conducteur du camion avait bien mis son clignotant à gauche mais sa remorque inversait le signal sur la droite. Ne m'ayant pas touché le conducteur s'est déclaré hors de cause et n'a pas voulu établir de constat. Ayant quitté ma voiture pour appeler un dépanneur j'ai retrouvé celle-ci avec la portière arrière droite enfoncée sans coordonnées du responsable.

Texte A3

Fort trafic à 17 h 15 Bd Sébastopol. Je roulais entre deux files de voitures arrêtées quand l'une des voitures à ma gauche a ouvert sa porte avant droite. Pour l'éviter, j'ai fait un écart qui m'a fait toucher le véhicule B avec l'arrière de ma moto ce qui a provoqué ma chute. Vu l'importance du trafic à cette heure là nous avons juste échangé nos assurances et noms ce qui explique que mon constat amiable ne soit signé que par moi.

Texte A4

Véhicule B venant de ma gauche, je me trouve dans le carrefour, à faible vitesse environ 40 km/h, quand le véhicule B, percute mon véhicule, et me refuse la priorité à droite. Le premier choc atteint mon aile arrière gauche, sous le choc, et à cause de la chaussée glissante, mon véhicule dérape, et percute la protection métallique d'un arbre, d'où un second choc frontal.

Texte A5
J'étais arrêté à l'intersection désirant emprunter la route où la circulation intense s'effectue à sens unique sur deux voies; lorsque le dernier véhicule du flot arrivait, j'ai voulu m'engager sur la deuxième file, lui laissant libre la première. Au moment où je démarrais, j'ai entendu le choc arrière; je ne m'attendais pas à ce qu'un usager désire me dépasser car il n'y avait pas deux voies matérialisées sur la portion de route où je me trouvais à l'arrêt.

Texte A6
Mr C. Delon, abordant le carrefour, laissait le passage aux véhicules roulant sur la voie abordée, car d'ordinaire se trouve un feu à ce carrefour (hors fonctionnement ce jour-là). Venant de derrière moi, roulant dans le même sens, dans la même file, Mr Oms n'a pas vu que j'étais arrêté et a percuté fortement mon véhicule, l'abîmant gravement. De ce fait, j'ai subi (C.Delon) le coup du lapin; le siège conducteur a été endommagé; les gendarmes se sont rendus sur place; j'ignore s'ils ont établi un rapport.

Texte A7
Étant arrêté momentanément sur la file de droite du Boulevard des Italiens j'avais mis mon clignotant j'étais à l'arrêt et m'apprêtant à changer de file. Le véhicule B arrivant sur ma gauche m'a serré de trop près et m'a abîme tout le côté avant gauche.

Texte A8
Je roulais sur la partie droite de la chaussée quand un véhicule arrivant en face dans le virage a été complètement déporté. Serrant à droite au maximum, je n'ai pu éviter la voiture qui arrivait à grande vitesse.

Texte A9
Nous roulions en ville sur une portion de route à deux voies où la vitesse est limitée à 45 km/h. Je clignotais et m'apprêtais à tourner à gauche vers le chemin de Condos. A ce moment, le véhicule B a doublé à grande vitesse notre véhicule et s'est immobilisé sur le trottoir gauche de la chaussée après m'avoir touché.

Texte A10
Nous roulions sur une route à 90 km/h. Un véhicule plein phares blancs nous aveuglant nous a percutés à grande vitesse à l'arrière du véhicule, nous entraînant dans une série de tonneaux avant l'immobilisation du véhicule dans un fossé.

Texte A11
Le conducteur du véhicule B me doublant par la droite a accroché mon pare-choc avant droit

et m'a entraîné vers le mur amovible du pont de Genevilliers que j'ai percuté violemment. D'après les dires du témoin le conducteur du véhicule B slalomait entre les voitures. qui me suivait, Après m'avoir heurtée, il a pris la fuite et n'a pu être rejoint par le témoin cité.

Texte A12

Je circulais à bord de mon véhicule A sur la file de droite réservée aux véhicules allant tout droit. Le véhicule B circulait sur la voie de gauche réservée aux véhicules allant à gauche (marquage au sol par des flèches). Celui-ci s'est rabattu sur mon véhicule A me heurtant à l'arrière gauche.

Texte A13

Je roulais dans la rue Pasteur quand une voiture surgit de ma droite; pour l'éviter, je me rabattais à gauche et freinais. Je pus l'éviter et mon rétroviseur heurte le sien. La voiture continue car elle n'eut rien et moi, je heurtais une benne qui stationnait sur le côté de la chaussée. La benne n'a pas été du tout endommagée. Ma voiture a été touchée à l'avant, ainsi qu'au rétroviseur.

Texte A14

Je circulais à environ 45 km/h dans une petite rue à sens unique où stationnaient des voitures de chaque côté. Surgissant brusquement sur ma droite sortant d'un parking d'immeuble, le véhicule de Mme Glorieux était à très peu de distance de mon véhicule; le passage étant impossible: surpris, je freinais immédiatement mais le choc fut inévitable.

Texte A15

Nous étions à Saint-Ouen, j'ai été surprise par la personne qui a freiné devant moi, n'ayant pas la possibilité de changer de voie et la route étant mouillée, je n'ai pu m'arrêter complètement à temps.

Texte A16

Je m'engageais (véhicule A) dans une file de station service. La pompe étant en panne, je reculais pour repartir lorsque j'ai heurté le véhicule B qui s'était engagé également dans la même file pour prendre de l'essence.

Texte A17

La conductrice de l'autre véhicule et moi amorcions le virage sur la gauche dans un carrefour. Nous étions à la même hauteur. Nous nous sommes certainement rapprochées et par conséquent percutées, sa voiture s'emboîtant dans la mienne, son aile gauche dans l'avant latéral droit de ma voiture.

Texte B18

Les deux véhicules A et B roulaient sur un rond-point dans 2 files différentes, le véhicule A sur la voie de gauche, le véhicule B sur celle de droite. A et B ont pris la même sortie du rond-point (sortie à deux voies en sens unique). C'est au moment de s'engager dans cette sortie que le véhicule A est venue percuter le véhicule B au niveau de la portière arrière gauche. Le véhicule B à été projeté dans le trottoir, ce qui a endommagé la roue arrière droite.

Texte B24

Circulant Bd Couturier, le véhicule B était stationné sur ce même Boulevard quand celui-ci a effectué une marche arrière dans la rue Constantinide, puis a redémarré brusquement vers le bd Couturier pour tourner à gauche et m'a percuté au moment où j'arrivais sur le carrefour. Le véhicule B arrivait de la partie gauche de sa chaussée et a coupé sa manœuvre pour tourner à gauche.

Texte B28

Revenant à mon domicile, le chauffeur du véhicule B me précédant a perdu le contrôle de son véhicule à cause d'un brutal verglas. Je n'ai pas pu à mon tour, contrôler mon véhicule qui après 20 mètres a percuté le véhicule de Mme Louvet. Je tiens à signaler qu'il n'y avait pas de verglas ailleurs et que nous avons été beaucoup de véhicules à déraper sur cette rue. Rien ne pouvait laisser prévoir 1 tel verglas.

Texte B29

Je descendais l'avenue du Gal De Gaulle, roulant à 45 km/h je voyais la voiture de Melle Perrin arrêtée au stop. En me rapprochant de l'intersection représentée sur le schéma, j'ai vu sa voiture s'engager sur l'avenue, alors que j'arrivais quasiment à sa hauteur. J'ai immédiatement commencé à freiner. Je ne pouvais pas continuer sur la même trajectoire, pour ne pas percuter la voiture du côté conducteur. Je ne pouvais pas me déporter sur la droite pour l'éviter , à cause du trottoir, des arbres et des panneaux de signalisation. Afin d'éviter le choc, j'ai donc braqué sur la gauche, pensant que Melle Perrin freinerait pour éviter l'accident. Malheureusement, et comme elle me l'a dit par la suite, elle regardait à ce moment sur la droite (je venais de gauche) et n'a pas vu mon véhicule : elle a donc continué à s'engager, regardant toujours sur la droite et a heurté mon véhicule au niveau de l'aile avant droite.

Texte B54

Je roulais à droite avenue de Libération en direction de la rue René Coty où je devais me rendre. Arrivant à cette rue, qui prend à gauche, je me suis rangé à droite le plus possible pour m'assurer qu'aucun véhicule n'arrivait. Je n'ai vu aucun véhicule proche ni devant ni derrière, et j'ai commencé à tourner à gauche. Ai-je mal apprécié la vitesse du véhicule A? Toujours est-il que la conductrice a freiné dans les dernières secondes, mais n'a pas pu éviter le choc. J'avais indiqué mon changement de direction.

Texte B57

Ce jour (le 23-3/93) au moment de m'engager sur une place de parking en sous-sol, j'ai éternué fortement, ce qui me fit relâcher involontairement l'angle de braquage du volant si bien que l'angle AVD de mon véhicule a heurté le pilier à l'angle droit de la place de parking (prière de voir croquis p.1!).Le pilier n'a ni traces du choc, ni dommages.

B.2 Corpus de textes simplifiés extraits des textes réels

Dans cette section, nous présentons les textes simplifiés utilisés pour tester l'ensemble de l'architecture du point de vue interprétation du texte ainsi que les résultats obtenus au niveau de l'analyse linguistique sur chacun de ces différents textes (la description des procès).

Texte S1 : je roulais sur la chaussée.

Procès 1_0 :
TEXTE : ``je roulais sur la chaussée''
OBJETS : S_0 : personne, V_0 : véhicule, E_0 : déplacement, R_0 : route, IE_0 : intervalle, IP_1 : intervalle, IM_1 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$
RELATIONS ÉVÉNEMENTIELLES :
[IP_1] véhicule(S_0, V_0) & [IP_1] conducteur(V_0, S_0) & à_roues(V_0) & énonciateur(S_0) & E_0.acteur = V_0 & [IP_1] appartient_dim2(S_0, R_0)

Texte S2 : je roulais dans le virage.

Procès 1_0 :
TEXTE : ``je roulais dans le virage''
OBJETS : S_0 : personne, V_0 : véhicule, E_0 : déplacement, SR_0 : virage, IE_0 : intervalle, IP_1 : intervalle, IM_1 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$
RELATIONS ÉVÉNEMENTIELLES :
[IP_1] véhicule(S_0, V_0) & [IP_1] conducteur(V_0, S_0) & à_roues(V_0) & énonciateur(S_0) & E_0.acteur = V_0 & [IP_1] appartient_dim2(S_0, SR_0)

Texte S3 : je roulais dans une rue à sens unique.

Procès 1_0 :

TEXTE : ``je roulais dans une rue à sens unique''

OBJETS : S_0 : personne, V_0 : véhicule, E_0 : déplacement, R_0 : route, IE_0 : intervalle, IP_1 : intervalle, IM_1 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$

RELATIONS ÉVÉNEMENTIELLES : énonciateur(S_0) & [IP_1] véhicule(S_0, V_0) & [IP_1] conducteur(V_0, S_0) & à_roues(V_0) & E_0.acteur = V_0 & [IP_1] appartient_dim2(S_0, R_0) & R_0.nature = ``urbaine'' & $R_0$.sens_circulation = ``unilatéral''

Texte S4 : je roulais dans une rue à double sens.

Procès 1_0 :

TEXTE : ``je roulais dans une rue à double sens''

OBJETS :

S_0 : personne, V_0 : véhicule, E_0 : déplacement, R_0 : route, IE_0 : intervalle, IP_1 : intervalle, IM_1 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$

RELATIONS ÉVÉNEMENTIELLES : énonciateur(S_0) & [IP_1] véhicule(S_0, V_0) & [IP_1] conducteur(V_0, S_0) & à_roues(V_0) & E_0.acteur = V_0 & [IP_1] appartient_dim2(S_0, R_0) & R_0.nature = ``urbaine'' & $R_0$.sens_circulation = ``bilatéral''

Texte S5 : je roulais lentement.

Procès 1_0 :

TEXTE : ``je roulais lentement''

OBJETS :

S_0 : personne, V_0 : véhicule, E_0 : déplacement, IE_0 : intervalle, IP_1 : intervalle, IM_1 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$

RELATIONS ÉVÉNEMENTIELLES : énonciateur(S_0) & [IP_1] véhicule(S_0, V_0) & [IP_1] conducteur(V_0, S_0) & à_roues(V_0) & E_0.acteur = V_0 & vitesse(E_0) = ``petite'' &

Texte S6 : j'ai roulé lentement.

Procès 1_0 :
TEXTE : ``j'ai roulé lentement''
OBJETS : S0 : personne, V0 : véhicule, E0 : déplacement,
IE0 : intervalle, IP1 : intervalle, IM1 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : IM1 < IE0 & IM1 | IP1
RELATIONS ÉVÉNEMENTIELLES : énonciateur(S0) &
[IP1] véhicule(S0, V0) & [IP1] conducteur(V0, S0) &
à_roues(V0) & E0.acteur = V0 & vitesse(E0) = ``petite''

Texte S7 : le véhicule b roulait à gauche.

Procès 1_0 :
TEXTE : ``le véhicule b roulait à gauche''
OBJETS : S1 : personne, V1 : véhicule, E0 : déplacement, Rep0 : repère,
IE0 : intervalle, IP1 : intervalle, IM1 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : IM1 < IE0 & IM1 ⊂ IP1
RELATIONS ÉVÉNEMENTIELLES : V1.nom = ``b'' & énonciateur(S0) &
[IP1] véhicule(S1, V1) & [IP1] conducteur(V1, S1) &
E0.acteur = V1 & [IP1] sup(position_y(V1, Rep0), ``0'')

Texte S8 : le véhicule b roulant à gauche arrivait en face.

Procès 1_0 :
TEXTE : ``le véhicule b arrivait en face''
OBJETS : S1 : personne, V1 : véhicule, E0 : déplacement, Rep0 : repère,
IE0 : intervalle, IP1 : intervalle, IM1 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : IM1 < IE0 & IM1 ⊂ IP1
RELATIONS ÉVÉNEMENTIELLES :
[IP1] véhicule(S1, V1) & [IP1] position_y(V1, Rep0) = "petite" &
E0.borneinfip = "horschamp" & V1.nom = ``b'' &
E0.bornesupip = "danschamp" & [IP1] conducteur(V1, S1) &
[IP1] sup(position_x(V1, Rep0), "0") & E0.acteur = V1

Procès 2_0 :

TEXTE : ''le véhicule b roulant à gauche''

OBJETS : s_1 : personne, v_1 : véhicule, E_1 : déplacement, Rep_1 : repère, IE_0 : intervalle, IP_2 : intervalle, IM_2 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_2 < IE_0$ & $IM_2 \subset IP_2$ & $IM_2 \mid IM_1$

RELATIONS ÉVÉNEMENTIELLES :

[IP_2] véhicule(s_1, v_1)& [IP_2] conducteur(v_1, s_1)& v_1.nom = ''b''& E_1.acteur = v_1 & [IP_2] sup(position_y(v_1, Rep_1), ''0'')

Texte S9 : le véhicule b roulait lentement dans le virage à sens unique.

Procès 1_0 :

TEXTE : ''le véhicule b roulait lentement dans le virage à sens unique''

OBJETS : s_1 : personne, v_1 : véhicule, E_0 : déplacement, SR_0 : virage, IE_0 : intervalle, IP_1 : intervalle, IM_1 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$

RELATIONS ÉVÉNEMENTIELLES :

[IP_1] conducteur(v_1, s_1)& [IP_1] véhicule(s_1, v_1)& v_1.nom = "b"& E_0.acteur = v_1 & SR_0.sens_circulation = "unilatéral" & vitesse(E_0) = "petite" & [IP_1] appartient_dim2(v_1, SR_0)

Texte S10 : roulant à droite dans le virage j'arrivais en face.

Procès 1_0 :

TEXTE : ''dans le virage j ' arrivais en face''

OBJETS : SR_0 : virage, s_0 : personne, v_0 : véhicule, E_0 : déplacement, Rep_0 : repère, IE_0 : intervalle, IP_1 : intervalle, IM_1 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$

RELATIONS ÉVÉNEMENTIELLES : [IP_1] position_y(s_0 = Rep_0, "petite")

[IP_1] véhicule(s_0, v_0)& [IP_1] conducteur(v_0, s_0)& E_0.acteur = s_0& [IP_1] appartient_dim2(s_0, SR_0) & E_0.borneinfip = "horschamp" & E_0.bornesupip = "danschamp" & énonciateur(s_0) & [IP_1] sup(position_x(s_0, Rep_0), "0") &

Procès 2_0 :
TEXTE : ‘‘roulant à droite dans le virage j ’ ’’
OBJETS : E_1 : déplacement, V_0 : véhicule, S_0 : personne, Rep_1 : repère,
SR_0 : virage, IE_0 : intervalle, IP_2 : intervalle, IM_2 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : $IM_2 < IE_0$ & $IM_2 \subset IP_2$ & $IM_2 \mid IM_1$
RELATIONS ÉVÉNEMENTIELLES : à_roues(V_0) & E_1.acteur = V_0 & énonciateur(S_0) &
[ip2_0] inf(position_y(S_0, Rep_1), "0") & [ip2_0] conducteur(V_0, S_0)&
[ip2_0] véhicule(S_0, V_0) & [ip2_0] appartient_dim2(S_0, SR_0)

Texte S11 : je roulais sur la partie droite de la chaussée .

Procès 1_0 :
TEXTE : ‘‘je roulais sur la partie droite de la chaussée’’
OBJETS : S_0 : personne, V_0 : véhicule, E_0 : déplacement, SR_1 : segment_route,
(1) Rep_0 : repère, R_1 : route, IP_1 : intervalle, IM_1 : intervalle,
IE_0 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$
RELATIONS ÉVÉNEMENTIELLES : à_roues(V_0) & E_0.acteur = V_0 &
[IP_1] véhicule(S_0, V_0) & [IP_1] conducteur(V_0, S_0) &
[IP_1] appartient_dim2(S_0, SR_1) & (1) [IP_1] lie(Rep_0, R_1) &
(1) [IP_1] inf(position_y(SR_1, Rep_0), "0") &
(2) [IP_1] droit(SR_1) & [IP_1] inclus(SR_1, R_1) & énonciateur(S_0)

Texte S12 : je roulais à droite sur la chaussée quand un véhicule arrivant en face dans le virage m'a percuté.

Procès 1_0 :
TEXTE : ‘‘je roulais à droite sur la chaussée (quand)’’
OBJETS :
S_0 : personne, V_0 : véhicule, E_0 : déplacement, Rep_0 : repère,
R_1 : route, IP_1 : intervalle, IM_1 : intervalle, IE_0 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$
RELATIONS ÉVÉNEMENTIELLES :
[IP_1] véhicule(S_0, V_0) & [IP_1] conducteur(V_0, S_0) &
à_roues(V_0) & E_0.acteur = V_0 & énonciateur(S_0) &
[IP_1] inf(position_y(S_0, Rep_0), "0") & [IP_1] appartient_dim2(S_0, R_1)

Procès 2_0 :

TEXTE : ``quand un véhicule m ' a percuté''

OBJETS : v_1 : véhicule, s_1 : personne, s_0 : personne, v_0 : véhicule,
E_1 : choc, IP_2 : intervalle, IM_2 : intervalle, IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_2 < IE_0$ & $IM_2 \mid IP_2$ & $IM_2 \mid IM_1$

RELATIONS ÉVÉNEMENTIELLES :
[IP_2] conducteur(v_1, s_1)& [IP_2] véhicule(s_1, v_1)& E_1.acteur = v_1&
[IP_2] véhicule(s_0, v_0) & [IP_2] conducteur(v_0, s_0) & énonciateur(s_0)

Procès 3_0 :

TEXTE : ``un véhicule arrivant en face dans le virage''

OBJETS : v_1 : véhicule, s_1 : personne, E_2 : déplacement, Rep_0 : repère,
SR_0 : virage, IP_3 : intervalle, IM_3 : intervalle, IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES :

RELATIONS ÉVÉNEMENTIELLES : [IP_3] position_y(v_1, Rep_0) = "petite"
E_2.acteur = v_1 & E_2.borneinfip = "horschamp" &
E_2.bornesupip = "danschamp" & [IP_3] véhicule(s_1, v_1) &
[IP_3] sup(position_x(v_1, Rep_0), "0") & [IP_3] conducteur(v_1, s_1) &

Texte S13 : je traversais le carrefour quand un véhicule m'a percuté.

Procès 1_0 :

TEXTE : ``je traversais le carrefour (quand)''

OBJETS : s_0 : personne, E_0 : déplacement, c_1 : carrefour,
IP_1 : intervalle, IM_1 : intervalle, IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$

RELATIONS ÉVÉNEMENTIELLES :
E_0.acteur = s_0 & [IP_1] appartient_dim2(s_0, c_1) & énonciateur(s_0)

Procès 2_0 :

TEXTE : ``quand un véhicule m ' a percuté''

OBJETS : v_1 : véhicule, s_1 : personne, s_0 : personne, E_1 : choc,
IP_2 : intervalle, IM_2 : intervalle, IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_2 < IE_0$ & $IM_2 \mid IP_2$ & $IM_2 \mid IM_1$

RELATIONS ÉVÉNEMENTIELLES : [IP_2] conducteur(v_1, s_1) & énonciateur(s_0) &
[IP_2] véhicule(s_1, v_1) &E_1.acteur = v_1 & E_1.objet = s_0

Texte S14 : je traversais le carrefour sur le passage pour piétons quand un véhicule m'a percuté.

Procès 1_0 :
TEXTE : ``je traversais le carrefour sur le passage pour piétons (quand)''
OBJETS : S_0 : personne, E_0 : déplacement, C_1 : carrefour, P_0 : passage,
IP_1 : intervalle, IM_1 : intervalle, IE_0 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$
RELATIONS ÉVÉNEMENTIELLES : E_0.acteur = S_0 & [IP_1] appartient_dim2(S_0, C_1)
[IP_1] appartient_dim2(S_0, P_0) & énonciateur(S_0)

Procès 2_0 :
TEXTE : ``quand un véhicule m ' a percuté''
OBJETS : V_1 : véhicule, S_1 : personne, S_0 : personne, E_1 : choc,
IP_2 : intervalle, IM_2 : intervalle, IE_0 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : $IM_2 < IE_0$ & $IM_2 \mid IP_2$ & $IM_2 \mid IM_1$
RELATIONS ÉVÉNEMENTIELLES : [IP_2] conducteur(V_1, S_1) & énonciateur(S_0) &
E_1.acteur = V_1 & E_1.objet = S_0 & [IP_2] véhicule(S_1, V_1)

Texte S15 : je circulais en moto sur la rn 13 quand un véhicule m'a percuté. sous le choc j'ai roulé sur la chaussée.

Procès 1_0 :
TEXTE : ``je circulais en moto sur la rn 13 (quand)''
OBJETS : S_0 : personne, V_0 : véhicule, E_0 : déplacement, R_1 : route,
IP_1 : intervalle, IM_1 : intervalle, IE_0 : intervalle
RELATIONS ASPECTUO-TEMPORELLES : $IM_1 < IE_0$ & $IM_1 \subset IP_1$
RELATIONS ÉVÉNEMENTIELLES : E_0.acteur = S_0 & énonciateur(S_0) &
[IP_1] véhicule(S_0, V_0) & [IP_1] appartient_dim2(S_0, R_1) & [IP_1]
conducteur(V_0, S_0)

Procès 2_0 :

TEXTE : ``quand un véhicule m ' a percuté''

OBJETS : v_1 : véhicule, s_1 : personne, s_0 : personne, v_0 : véhicule, e_1 : choc, ip_2 : intervalle, im_2 : intervalle, ie_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $im_2 < ie_0$ & $im_2 \mid ip_2$ & $im_2 \mid im_1$

RELATIONS ÉVÉNEMENTIELLES : [ip_2] conducteur(v_1, s_1) & énonciateur(s_0) & [ip_2] véhicule(s_0, v_0) & [ip_2] conducteur(v_0, s_0) & e_1.acteur = v_1 & e_1.objet = s_0 & [ip_2] véhicule(s_1, v_1)

Procès 3_0 :

TEXTE : ``sous le choc j ' ai roulé sur la chaussée''

OBJETS : e_1 : choc, s_0 : personne, v_0 : véhicule, e_2 : déplacement, r_1 : route, ip_3 : intervalle, im_3 : intervalle, ie_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $im_3 < ie_0$ & $im_3 \mid ip_3$

RELATIONS ÉVÉNEMENTIELLES : [ip_3] véhicule(s_0, v_0) & e_2.acteur = s_0 & [ip_3] appartient_dim2(s_0, r_1) & énonciateur(s_0)

Texte S16 : je traversais le carrefour sur le passage pour piétons quand un véhicule m'a percuté. sous le choc j'ai roulé sur la chaussée.

Procès 1_0 :

TEXTE : ``je traversais le carrefour sur le passage pour piétons (quand)''

OBJETS : s_0 : personne, e_0 : déplacement, c_1 : carrefour, p_0 : passage, ip_1 : intervalle, im_1 : intervalle, ie_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $im_1 < ie_0$ & $im_1 \subset ip_1$

RELATIONS ÉVÉNEMENTIELLES : e_0.acteur = s_0 & énonciateur(s_0) [ip_1] appartient_dim2(s_0, p_0) & [ip_1] appartient_dim2(s_0, c_1)

Procès 2_0 :

TEXTE : ``quand un véhicule m ' a percuté''

OBJETS : v_1 : véhicule, s_1 : personne, s_0 : personne, e_1 : choc, ip_2 : intervalle, im_2 : intervalle, ie_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $im_2 < ie_0$ & $im_2 \mid ip_2$ & $im_2 \mid im_1$

RELATIONS ÉVÉNEMENTIELLES : [ip_2] conducteur(v_1, s_1) & énonciateur(s_0) & e_1.acteur = v_1 & e_1.objet = s_0 & [ip_2] véhicule(s_1, v_1)

Procès 3_0 :

TEXTE : ``sous le choc j ' ai roulé sur la chaussée''

OBJETS : E_1 : choc, S_0 : personne, E_2 : déplacement, R_1 : route, IP_3 : intervalle, IM_3 : intervalle, IE_0 : intervalle

RELATIONS ASPECTUO-TEMPORELLES : $IM_3 < IE_0$ & $IM_3 \mid IP_3$

RELATIONS ÉVÉNEMENTIELLES :

E_2.acteur = S_0 & [IP_3] appartient_dim2(S_0, R_1) & énonciateur(S_0)

B.3 Bases de connaissances de l'AMS

B.3.1 Définition des catégories morphologiques

La liste suivante contient l'ensemble des catégories actuellement définies pour l'analyseur morpho-syntagmatique. Cette liste contient, pour chacune des catégories possibles, les divers attributs possibles. Cette liste, du fait de l'état d'avancement de nos travaux, est incomplète et sera amenée à évoluer.

{ nom : genre nbr source }
{ pronomp : pers genre nbr source }
{ det : genre nbr source }
{ v_p_pres : source }
{ prep : introduit source }
{ adj : genre nbr source }
{ clitique : cas pers genre nbr source }
{ v_p_passe : genre nbr source aux }
{ q_adv : source }
{ q_pron : cas nbr genre pers source }
{ v_conj : tps pers nbr source }
{ adv : source }
{ loc_adv : source }
{ v_inf : source }
{ ponct : source }
{ neg : source }
{ nb : }
{ abrev : genre nbr suite }
{ nom_propre : source }

B.3.2 Définition des attributs catégoriels

La liste suivante contient l'ensemble des attributs catégoriels et indique, pour chacun de ses éléments, les valeurs que peut prendre un attribut. L'attribut "source" permet de mentionner le dictionnaire d'où est issu le lemme (DUG : Dictionnaire des Unités Grammaticales ; DUL : dictionnaire des unités lexicales).

L'attribut "introduit" est utilisé en général pour les prépositions et permet de spécifier les catégories morphologiques des mots susceptibles de "suivre" la préposition : *en* introduit en général un participe présent ; *pour* introduit en général un substantif ou un verbe.

{ source : DUG DUL }
{ genre : fem masc }
{ nbr : sing plur }
{ introduit : nom v_inf v_p_pres pronomp }
{ cas : nominatif genitif accusatif datif ablatif }
{ pers : 1 2 3 }
{ tps : imp_ind pres_ind pres_subj }
{ aux : avoir être }
{ suite : nb }

B.3.3 Quelques règles de construction des segments

Construction d'un groupe verbal

Règle permettant de construire des groupes comme *je donne*, *je le lui donne* mais dans laquelle les contraintes globales permettent de ne pas prendre en compte des groupes comme *je le te donne*, groupes pour lesquels l'ordre "accusatif - datif" n'est pas correct. !(2.pers=3 & (3.pers=1 | 3.pers=2)) signifie que l'on ne peut pas avoir en même temps un clitique de cas accusatif à la troisième personne et un clitique de cas datif à la première ou seconde personne.

```
{ SV_conj :
   const [ 0? clitique (cas=nominatif)
           1? neg
           2? clitique(cas=accusatif)
```

```
      3? clitique(cas=datif)
      4? clitique(cas=ablatif)
      5? clitique(cas=genitif)
      6 v_conj ]
  cont  [ { 0.pers = 6.pers } { 0.nbr = 6.nbr }
        {!(2.pers=3 & (3.pers=1 | 3.pers=2))}] }
```

Règle permettant de construire uniquement des groupes comme *je te le donne* ou *le voisin te le donne*.

```
{ SV_conj1 SV_conj :
  const [ 0? clitique (cas=nominatif)
      1? neg
      2 clitique(cas=datif, pers=1 2 )
      3 clitique(cas=accusatif, pers = 3 )
      4 v_conj ]
  cont  [ { 0.pers = 4.pers } { 0.nbr = 4.nbr}] }
```

Construction d'un groupe nominal

Règle de construction des groupes nominaux correspondant à la partie d'un syntagme nominal classique comprise entre le déterminant et le nom (comme par exemple, *la petite voiture* ou *la voiture* mais pas *la voiture verte*).

```
{ SN :
  const [ 0 det
      1* adj
      2 nom ]
  cont  [ { 0.genre = 1.genre = 2.genre } { 0.nbr = 1.nbr =2.nbr }]
  aff   [ { nbr = 2.nbr } { genre = 2.genre } { pers = 3 }] }
```

Règle de construction de segments nominaux sans déterminant (construction moins fréquente que la précédente et donc d'ordre de priorité égal à 1).

```
{ SN1 SN : ponderation 1
  const [ 0* adj
      1 nom ]
  cont  [ { 0.genre = 1.genre } { 0.nbr = 1.nbr }]
  aff   [ { nbr = 1.nbr } { genre = 1.genre } { pers = 3 }] }
```

Règle de construction de segments nominaux prépositionnels.

```
{ SN2 SN :
  const [ 0 prep (introduit=nom)
          1? det
          2* adj
          3 nom]
  cont  [ { 1.genre = 2.genre = 3.genre } { 1.nbr = 2.nbr = 3.nbr }]
  aff   [ { nbr = 3.nbr } { genre = 3.genre } { pers = 3 }] }
```

Règle permettant de construire des groupes comme "sur la rn 13" où rn n'est pas un substantif mais est ce que nous avons appelé une abréviation.

```
{ SN3 SN :
  const [ 0 prep (introduit = nom)
          1 det
          2* adj
          3 abrev (suite = nb)
          4 nb]
  cont  [ { 1.genre = 2.genre = 3.genre } { 1.nbr = 2.nbr = 3.nbr }]
  aff [   { nbr = 3.nbr } { genre =3.genre } { pers = 3 }]}
```

B.3.4 Quelques règles de construction des liens potentiels

Ces règles sont de la forme :

liste de REGI : liste de REGISSANT { conditions }

ce qui signifie : REGI rattaché au REGISSANT si les conditions sont respectées

Rattachements à l'intérieur d'une même proposition

Conditions de rattachement d'un SN ou un SPron à un :
- SV_conj : aucune.
- SV_p_passe, SV_p_pres, SN, SAdj, SAdv ou SV_inf : le régissant est avant le régi et il n'existe aucun verbe conjugué entre les deux.

```
{ SN SPron :
  SV_conj {}
  SV_p_passe SV_p_pres SN SAdj SAdv SV_inf
    { ( REGISSANT < REGI ) &!( REGISSANT < SV_conj < REGI) } }
```

Conditions de rattachement d'un SAdj, un SV_p_passe ou un SV_p_pres à un :

- SN, SV_p_passe, SV_p_pres : le régissant est avant le régi et il n'existe aucun verbe conjugué entre les deux.
- SN non prépositionnel : le régissant est après le régi et il n'existe aucun verbe conjugué entre les deux.
- SV_conj : aucune.

```
{ SAdj SV_p_passe SV_p_pres :
  SN SV_p_passe SV_p_pres
    { ( REGISSANT < REGI ) &!( REGISSANT < SV_conj < REGI ) }
  SN
    {(REGISSANT > REGI) &!REGISSANT.prep &
      !(REGI < SV_conj < REGISSANT)}
  SV_conj {}}
```

Conditions de rattachement d'un SAdv à un :

- SV_conj : aucune.
- SV_p_passe, SV_p_pres : le régissant est avant le régi et il n'existe aucun verbe conjugué entre les deux.

```
{ SAdv :
  SV_conj {}
  SV_p_passe SV_p_pres
    { ( REGISSANT < REGI ) &!( REGISSANT < SV_conj < REGI ) }}
```

Conditions de rattachement d'un SQ_conj, SQ_rel, SV_inf à un :

- SV_conj : aucune.

```
{ SQ_conj SQ_rel SV_inf :
  SV_conj {}}
```

Conditions de rattachement d'un SNpropre à un :

- SN : le régissant est avant le régi et il n'existe aucun verbe conjugué entre les deux.

```
{ SNpropre :
  SN
    { ( REGISSANT < REGI ) &!( REGISSANT < SV_conj < REGI) }}
```

Rattachement d'une proposition

Conditions de rattachement d'un SQ_conj à un :
- SV_conj : aucune.

{ PROP SQ_conj :
SV_conj {}}

Conditions de rattachement d'un SQ_rel à un :
- SN : le régissant est avant le régi et il n'existe aucun verbe conjugué entre les deux.

{ PROP SQ_rel :
SN
{(REGISSANT < REGI) & !(REGISSANT < SV_conj < REGI)}}

Annexe C

L'interface graphique

L'interface graphique permet à l'utilisateur de paramétrer, de façon conviviale, les différentes analyses (changer, entre autres, de fichiers de données pour l'analyseur linguistique) et visualiser les différentes tâches utilisées. Cette interface se compose d'une fenêtre principale par l'intermédiaire de laquelle il est possible d'accéder aux fenêtres associées à chacun des sous-systèmes composant l'ensemble de l'architecture.

C.1 La fenêtre principale

La fenêtre principale de l'interface graphique est composées de 7 fenêtres filles (cf. figure C.1) dont les fonctionnalités sont les suivantes :.

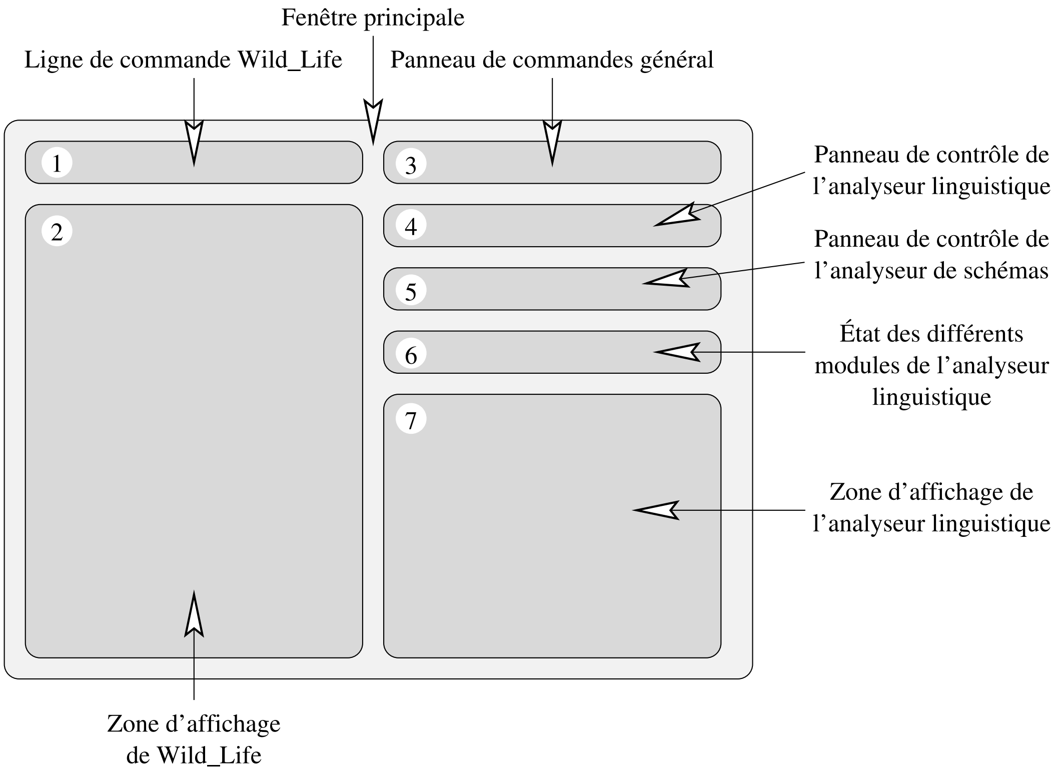


FIG. C.1 - *La fenêtre principale*

Fenêtre 1 : permet de saisir des commandes Life qui sont ensuite directement envoyées au sous-système chargé d'effectuer la construction de la scène. Cette fenêtre permet, entre autres, d'interroger ce sous-système sur la scène construite.

Fenêtre 2 : zone où sont affichées les commandes exécutées par le sous-système de construction de la scène (cf. figure C.22). Cette zone sert aussi à afficher le résultat des interrogations faites auprès de ce module.

Fenêtre 3 : zone contenant un ensemble de boutons permettant de contrôler l'analyse (cf. figure C.2 - chargement du texte et du schéma à analyser, démarrage ou arrêt de l'analyse ...).

Fenêtre 4 : zone contenant un ensemble de boutons permettant d'intervenir sur l'analyse linguistique (cf. figure C.3 - configuration de l'analyse, visualisation du résultat, sauvegarde du résultat ...).

Fenêtre 5 : zone contenant un ensemble de boutons permettant d'intervenir sur l'analyse du schéma (cf. figure C.4 - configuration de l'analyse, visualisation du résultat, sauvegarde du résultat ...).

Fenêtre 6 : zone où figurent les noms des différents modules composant l'analyseur linguistique (cf. figure C.5). Cette zone permet d'une part de visualiser l'état des différents modules (actif, désactivé ou inactif - dans le cas de la figure C.5, le module AMS est désactivé, le module RA est actif et les autres sont inactifs) et d'autre part, d'accéder en cliquant sur le nom d'un module aux données détenues par ce module.

Fenêtre 7 : zone où apparaissent des messages indiquant les diverses tâches entreprises par les modules de l'analyseur linguistique (cf. figure C.9).

Les fenêtres 4 et 5 contiennent toutes les deux un bouton “Configure”, celui-ci permet à l'utilisateur d'appeler la fenêtre de configuration correspondant à chacun des sous-systèmes (cf. figures C.6 et C.7).

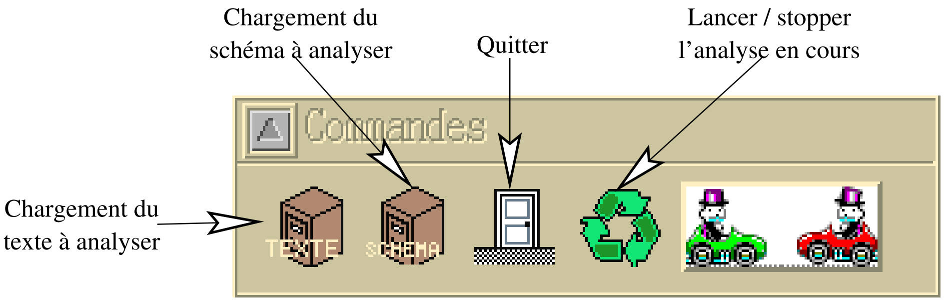


FIG. C.2 - *Panneau de commandes général*

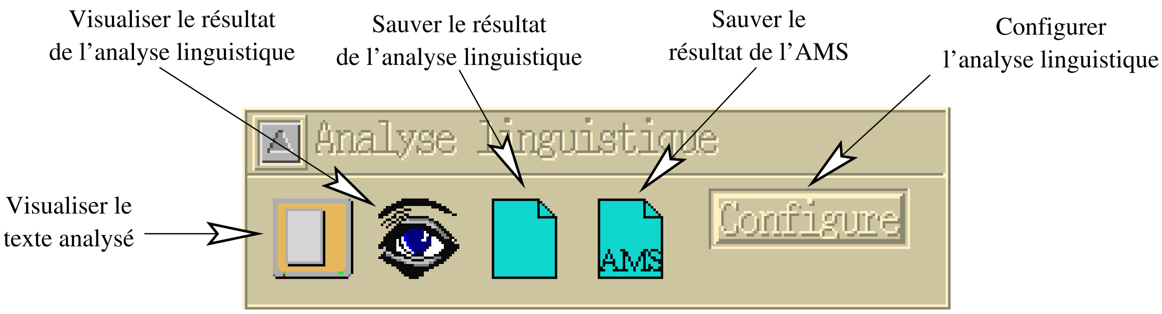


FIG. C.3 - *Panneau de commandes de l'analyseur linguistique*

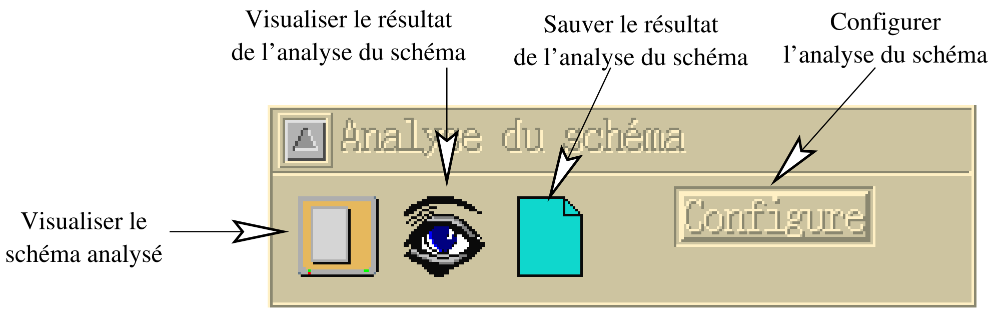


FIG. C.4 - *Panneau de commandes de l'analyseur de schémas*

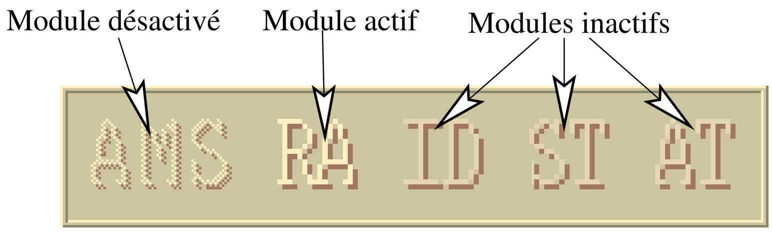


FIG. C.5 - *Les modules de l'analyseur linguistique*

FIG. C.6 - *Configuration de l'analyseur linguistique*

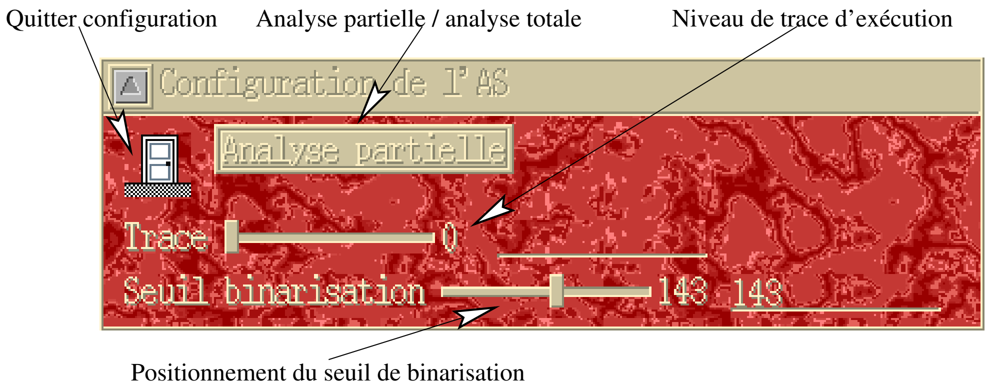


FIG. C.7 - *Configuration de l'analyseur de schémas*

C.2 Fenêtres liées à l'analyseur linguistique

Dans le cas de l'analyseur linguistique, chacun des modules peut être configuré. Pour chacun de ces modules, il est possible de modifier plusieurs éléments (cf. figure C.8) : les ressources utilisées par ce module (on peut alors modifier le noms des fichiers de données du module - cf. figures C.10) et C.19, le niveau de visualisation (on peut alors supprimer l'affichage des résultats et messages de ce module), le niveau d'activité du module (on peut le rendre totalement inactif et dans ce cas, l'analyse se fera sans ce module) et le niveau d'exécution (on peut demander une exécution en pas à pas d'un module de façon à pouvoir examiner des résultats intermédiaires - cf. figure C.11, il est possible de stopper l'analyse morpho-syntagmatique après chacune de ses étapes). La visualisation des différentes tâches effectuées par l'analyseur linguistique est effectuée par l'intermédiaire de la fenêtre de la figure C.9.

FIG. C.8 - *Configuration d'un module de l'analyseur linguistique*

```
Déroulement de l'analyse linguistique

Début analyse
RA: Demande Modèle des Attentes
RA: Réception Modèle des Attentes
Début texte
AMS: début analyse
AMS: analyse terminée
AMS: Transfert résultat analyse (phrase 1 )
AMS: fin transfert
RA: Début phrase
RA: Début chemin
RA: Début proposition
RA: Recherche des entrées lexicales
RA: Recherche des constructions possibles
RA: Analyse d'un choix de construction
RA: Appel au ID pour identification des entités des schémas
ID: Identification de 'v' tel qu'on ait relation12(v, je)
ID: entite v0 existante
ID: Identification de 'v' tel qu'on ait relation11(je, v)
ID: entite v0 existante
RA: Recherche des schémas possibles en fonction des réponses du ID
RA: Choix de schéma
RA: Vérification des liens et suppression des liens inutiles
RA: Vérification des liens ok
RA: Déterminer les verbes principaux
RA: Verbe principal de je roulais : roulais
RA: appels à ID, AT et ST pour compléter l'analyse de la proposition
RA: proces 1_0, groupe verbal 'je roulais'
RA: identification des entités (appel au ID)
ID: identification de 'je' => s0
ID: identification de 'sur la chaussée' => r0
RA: relations aspectuo-temporelles (appel au AT)
RA: relations spatio-temporelles (appel au ST)
AT: relations aspectuo-temporelles de je roulais => ante(im1_0, ie0)
RA: execution des instructions contenues dans le schéma (appel au modu
lié à l'instruction)
ID: instruction identification liée au schéma de 'roulais'
ID: instruction identification liée au schéma de 'roulais'
RA: Fin du procès 1_0
RA: Fin du choix de schéma courant
RA: Fin choix de construction
RA: Fin proposition
RA: Fin chemin
RA: Fin phrase
RA: Transmission des procès à WL et attente d'une validateion
RA: Réception du nouveau Modèle des Attentes
 => validation de l'analyse de la phrase
On passe à la phrase suivante
AMS: Transfert résultat analyse (phrase 2 )
AMS: pas de phrase
RA: pas de phrase suivante => fin du texte
RA: demande à WL de valider l'analyse du texte
RA: analyse du texte validée
 => fin de l'analyse du texte
```

FIG. C.9 - *Visualisation des différentes tâches effectuées par l'analyseur linguistique dans le cas de l'analyse de : "Je roulais sur la chaussée"*

C.2.1 Fenêtres liées à l'analyseur morpho-syntagmatique

Ces fenêtres sont au nombre de 9 :

Fenêtre 1 : La fenêtre de configuration de l'analyseur morpho-syntagmatique qui permet d'indiquer, entre autres, les fichiers contenant le dictionnaire des formes fléchies et ceux contant les grammaires (cf. figure C.10).

Fenêtre 2 : Une fenêtre permettant de fixer les points d'arrêt de l'analyse (cf. figure C.11).

Fenêtre 3 : La fenêtre de visualisation du déroulement de l'analyse (cf. figure C.12) qui permet, non seulement de visualiser les graphes résultant de l'analyse, mais aussi d'accèder, par l'intermédiaire de son panneau de commande, aux diverses informations portées par les arcs.

Fenêtre 4 : La fenêtre de visualisation des arcs (cf. figure C.13) qui peut être appelée grâce au panneau de commande de la fenêtre 3 permet, sur saisie d'un numéro d'arc, de visualiser les données portées par ce dernier.

Fenêtre 5 : La fenêtre de visualisation des nœuds (cf. figure C.14) qui peut être appelée grâce au panneau de commande de la fenêtre 3 permet, sur saisie d'un numéro de nœud, de visualiser les données portées par ce dernier.

Fenêtre 6 : La fenêtre de visualisation des propositions (cf. figure C.15) qui peut être appelée grâce au panneau de commande de la fenêtre 3 permet, sur saisie d'un numéro de proposition, de visualiser les données liées à cette dernière.

Fenêtre 7 : La fenêtre de visualisation du dictionnaire (cf. figure C.16) qui peut être appelée grâce au panneau de commande de la fenêtre 3 permet, sur saisie d'une chaîne de caractères, de visualiser les données liées à l'unité morphologique correspondante.

Fenêtre 8 : La fenêtre de visualisation des catégories (cf. figure C.17) qui peut être appelée grâce au panneau de commande de la fenêtre 3 permet de visualiser les catégories morphologiques avec leur liste de traits possibles.

Fenêtre 9 : La fenêtre de visualisation des traits morphologiques (cf. figure C.18) qui peut être appelée grâce au panneau de commande de la fenêtre 3 permet de visualiser les valeurs possibles pour chacun des traits morphologiques.

FIG. C.10 - *Configuration des ressources de l'analyseur morpho-syntagmatique*

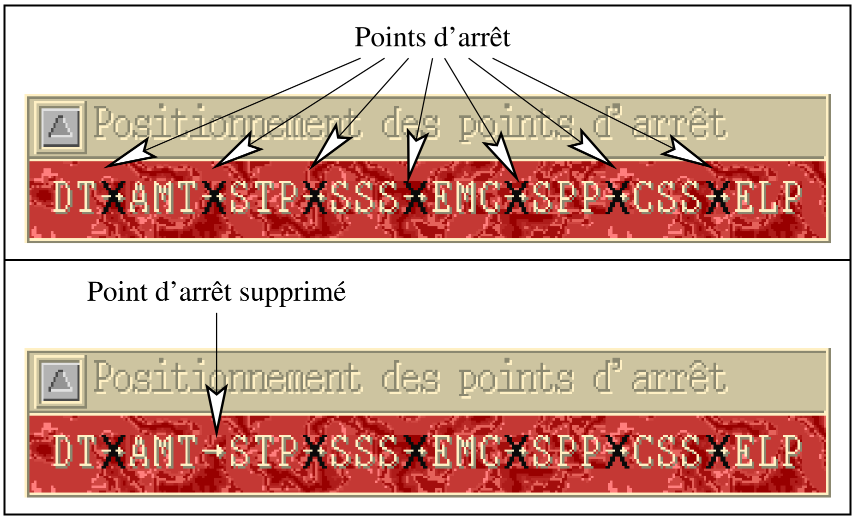


FIG. C.11 - *Points d'arrrêt de l'analyseur morpho-syntagmatique*

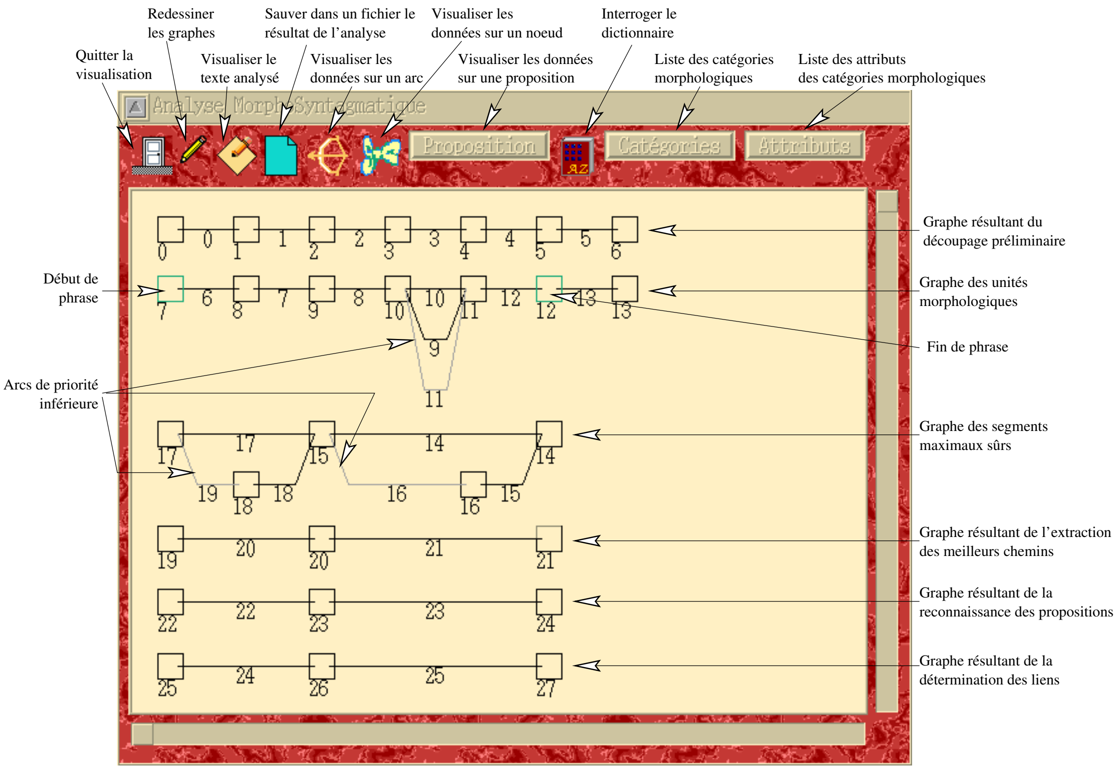


FIG. C.12 - *Visualisation de l'analyse effectuée par l'analyseur morpho-syntagmatique pour la phrase "Je roulais sur la chaussée"*

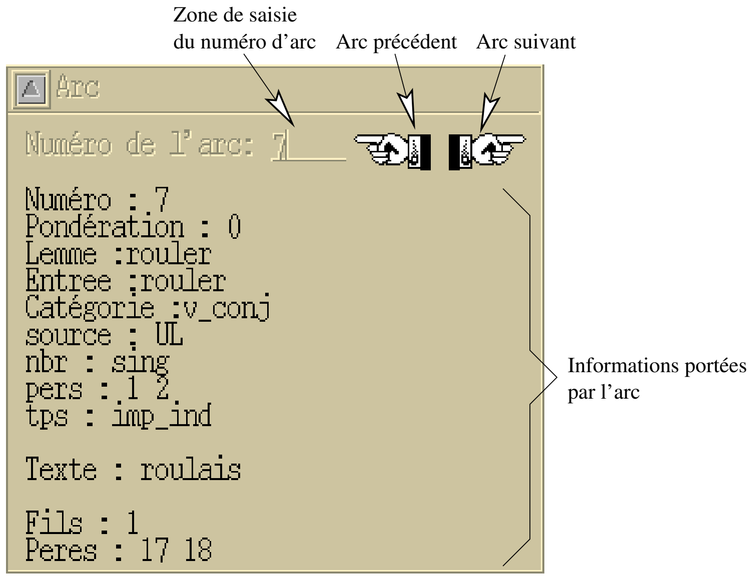


FIG. C.13 - *Visualisation des données portées par un arc*

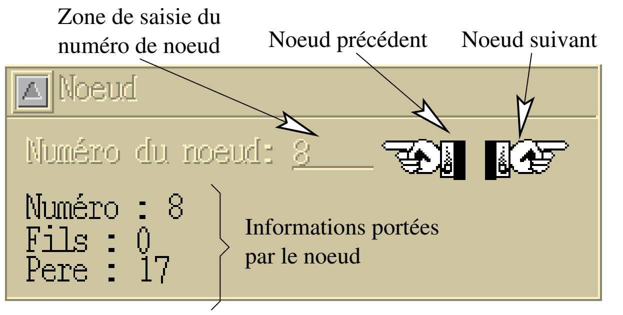


FIG. C.14 - *Visualisation des données portées par un nœud*

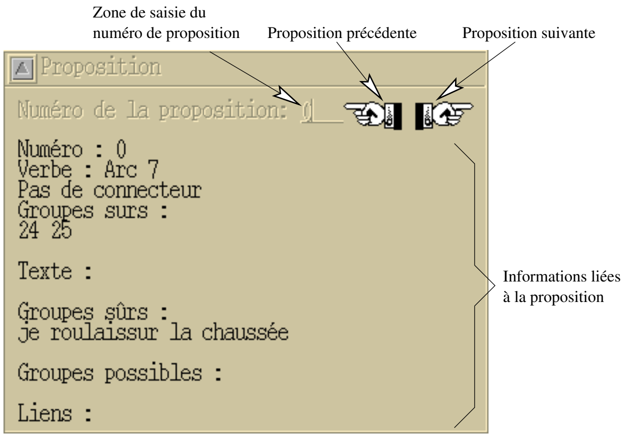


FIG. C.15 - *Visualisation des données liées à une proposition*

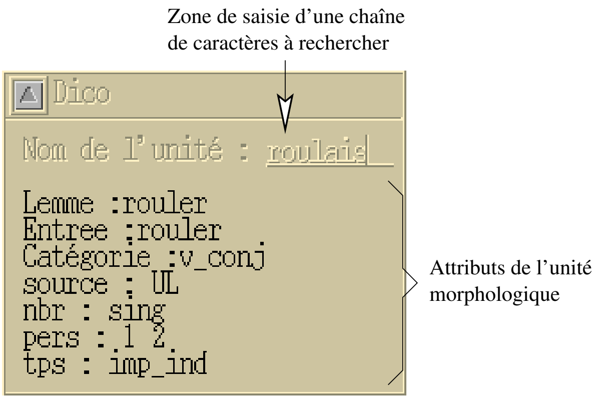


FIG. C.16 - *Recherche d'une unité morphologique dans le dictionnaire*

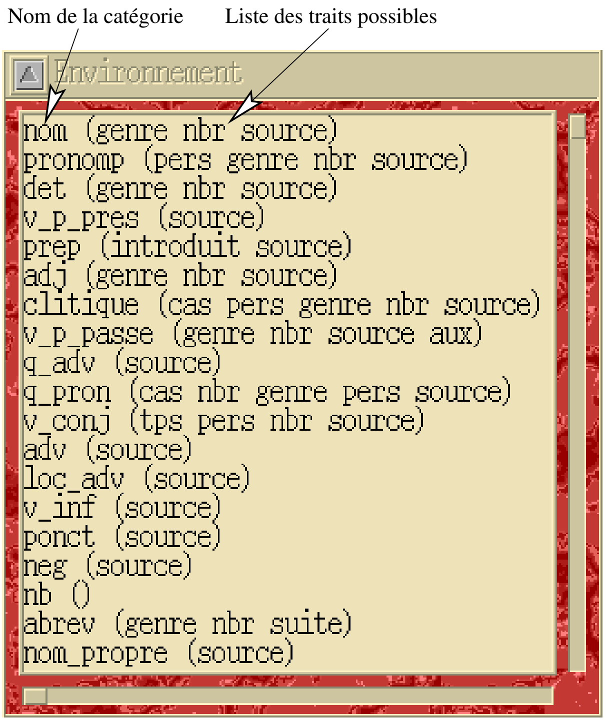


FIG. C.17 - *Visualisation des catégories morphologiques*

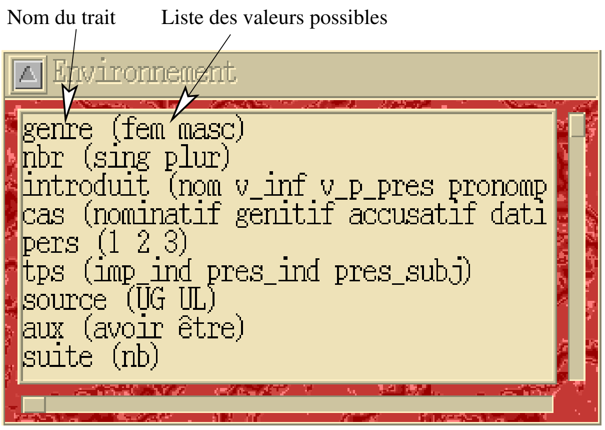


FIG. C.18 - *Visualisation des traits morphologiques*

C.2.2 Fenêtres liées au module d'analyse des relations actancielles

Ces fenêtres sont au nombre de 2 :

Fenêtre 1 : La fenêtre de configuration de l'analyseur morpho-syntagmatique qui permet d'indiquer, entre autres, les fichiers contenant le dictionnaire des formes fléchies et ceux contant les grammaires (cf. figure C.19).

Fenêtre 2 : La fenêtre de visualisation du résultat de l'analyseur linguistique (cf. figure C.20) qui permet de visualiser les descriptions de procès générées en donnant des indications sur les causes de la création des entités et relations.

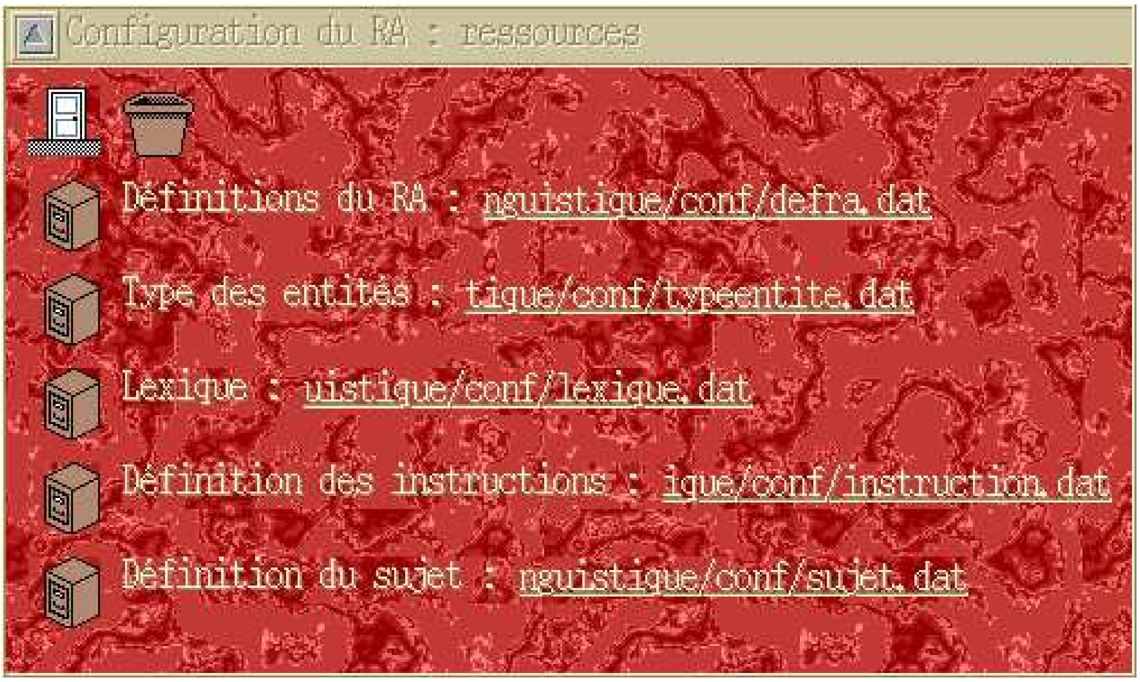


FIG. C.19 - *Configuration des ressources du module d'analyse des relations actancielles*

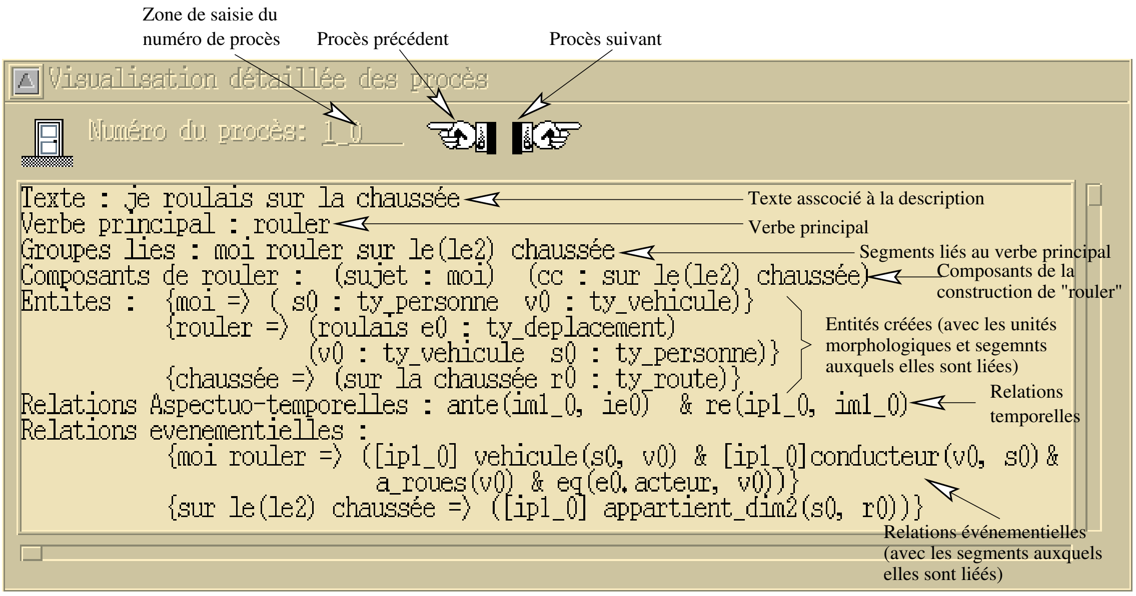


FIG. C.20 - *Visualisation du résultat de l'analyseur linguistique*

C.2.3 Fenêtres liées au module d'identification des entités

La seule fenêtre qui est pour l'instant associée à ce module est celle qui permet de visualiser les entités créées avec une partie des relations qu'elles entretiennent (celles qui sont issues du modèle initial des attentes du lecteur - cf. figure C.21).

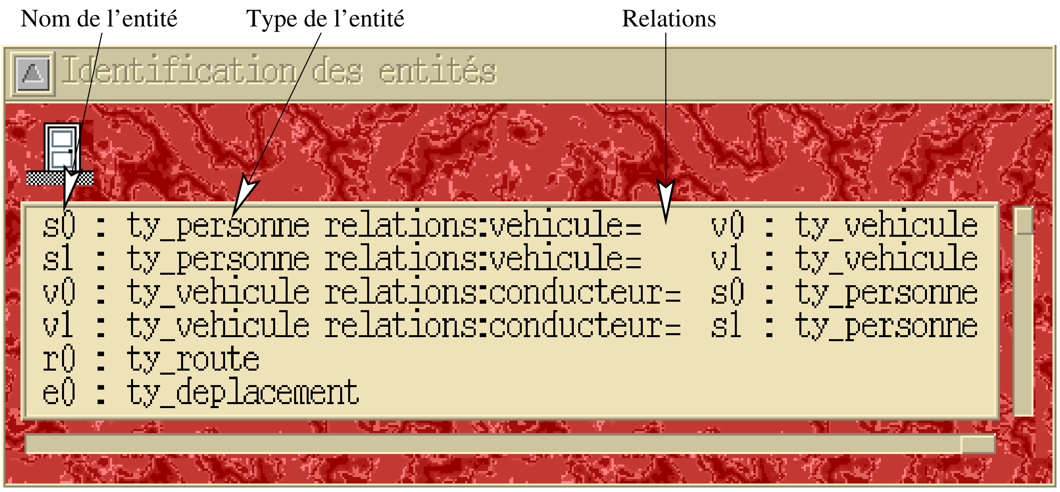


FIG. C.21 - *Visualisation des entités*

C.3 Fenêtres liées à la construction de la scène

Les fenêtres liées au sous-système de construction de la scène sont au nombre de 2 : une fenêtre qui permet de suivre le déroulement des tâches effectuées par ce sous-système (cf. figure C.22) et une qui permet de visualiser le déroulement de la scène (cf. figure C.23).

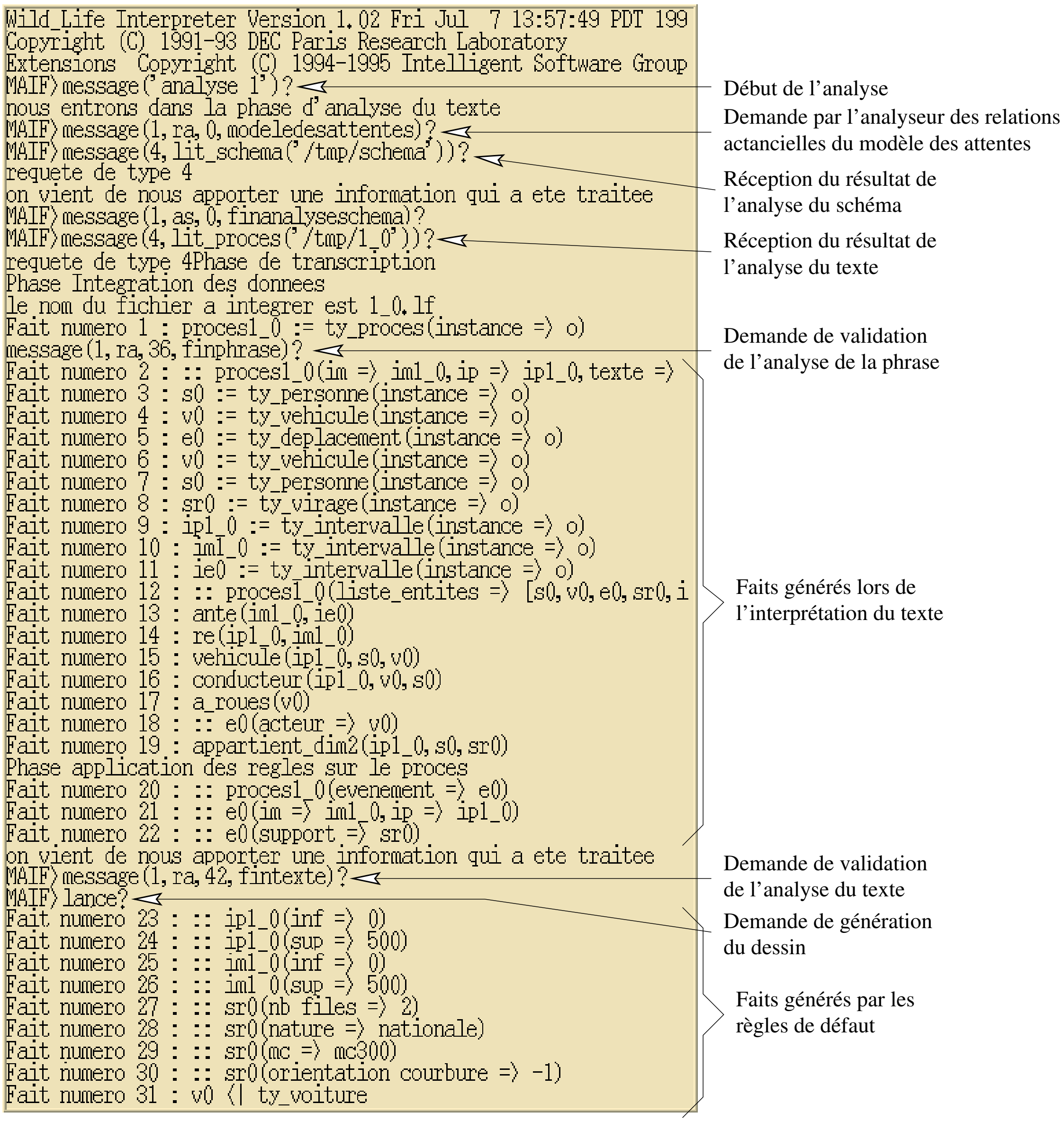


FIG. C.22 - *Visualisation des tâches affectuées par le sous-système de construction de la scène*

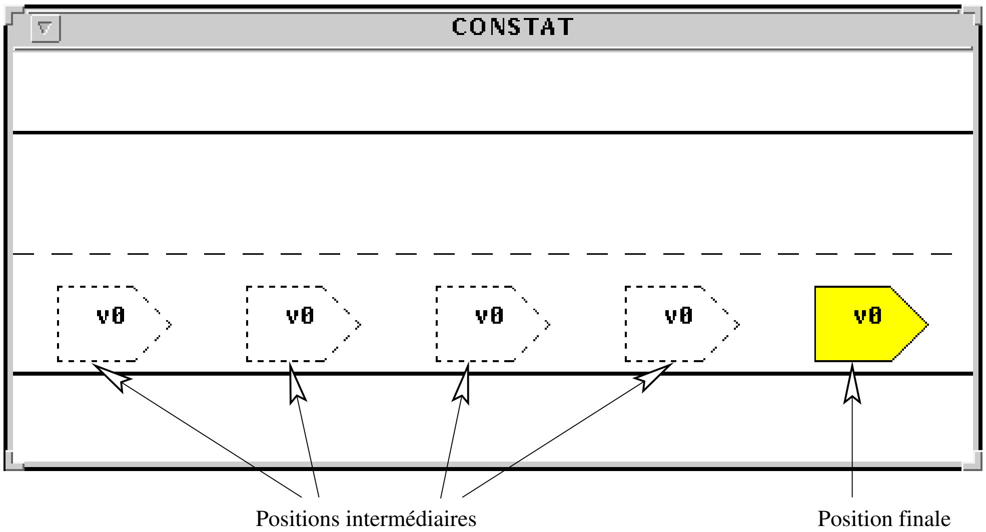


FIG. C.23 - *Visualisation de la scène construite pour l'analyse de la phrase "Je roulais sur la chaussée"*

Table des figures

L'ensemble du projet

L'analyseur de schémas

L'analyseur linguistique

Mise en place du système global

Liste des tableaux

L'ensemble du projet

L'analyseur de schémas

L'analyseur linguistique

Mise en place du système global

Table des matières

Résumé

Nous proposons une architecture capable de fournir une représentation de la scène décrite par un document composé à la fois d'un texte et d'un schéma (document composite). Le corpus utilisé pour mettre au point cette architecture est un corpus de constats amiables d'accidents automobiles (constats remplis par les conducteurs).

La caractéristique principale de cette architecture réside dans le fait qu'elle réparti le travail d'interprétation du document en trois sous-systèmes bien distincts. Le premier (analyseur de schémas) a pour but d'effectuer une analyse géométrique du schéma. Le second (analyseur linguistique) a pour but d'extraire les entités et relations présentes dans le texte en n'utilisant que des connaissances linguistiques. Le dernier a pour but de reconstruire la scène en utilisant les données fournies par les deux analyseurs ainsi qu'un ensemble de connaissances sur le monde et les conditions d'énonciation.

Mots clés

Intelligence artificielle, Images, traitement des, Linguistique informatique, Traitement réparti, Analyse documentaire, Analyse du discours.

Christèle Poirier

Décembre 1996

Université de Caen

Groupe de Recherche en Informatique, Image et Instrumentation de Caen

CNRS URA 1526