

François-Régis Chaumartin

Master de Recherche en Linguistique et Informatique

Conception et réalisation d'une interface syntaxe / sémantique utilisant des ressources de large couverture en langue anglaise



2004 – 2005

Conception et réalisation d'une interface syntaxe / sémantique utilisant des ressources de large couverture en langue anglaise

Abstract

En Traitement Automatique du Langage, disposer d'une représentation sémantique (ou conceptuelle) d'un texte est un préalable pour en « comprendre » le sens. Nous décrivons ici comment passer d'une forme syntaxique (issue d'une analyse grammaticale) à une forme sémantique (sous forme de prédicats et de relations entre ces prédicats).

Notre approche s'appuie sur l'interopérabilité de plusieurs ressources couvrant des aspects d'ordre syntaxique (*Link Grammar Parser*), lexical (*WordNet*) et syntaxico-sémantique (*VerbNet*) de la langue anglaise. L'utilisation conjointe de ces ressources de large couverture permet d'obtenir des résultats encourageants, en termes de désambiguïsation syntaxique et lexicale. Elle permet aussi d'attribuer à chaque interprétation d'une phrase un indice de vraisemblance sémantique.

Mots clés

TAL, compréhension de texte, analyse syntaxique, désambiguïsation, interface syntaxe-sémantique, identification de schémas, logique des prédicats, unification, grammaire, rôles thématiques, contraintes de sélections, VerbNet, WordNet, Link Grammar Parser, satisfaction de contraintes.

Abstract

In Natural Language Processing, we must first compute a semantic representation of a text prior to “understanding” it. We describe here how to pass from a syntactic structure (generated by a syntactic parser of English) to a semantic form (in the form of predicates and relations between these predicates).

*Our approach is based on the interoperability between several resources, covering syntactical (*Link Grammar Parser*), lexical (*WordNet*) and semantic (*VerbNet*) aspects of English. The joint use of these broad-coverage resources leads to encouraging results on lexical and syntactical disambiguation. That also makes it possible to assign a “semantic probability” to each interpretation of a sentence.*

Keywords

NLP, text understanding, parsing, disambiguation, syntax/semantic interface, pattern recognition, predicate calculus, unification, grammar, thematic roles, selectional restrictions, VerbNet, WordNet, Link Grammar Parser, constraints solver.

Remerciements

Il est d'usage de commencer un document de ce type par des remerciements, adressés à ceux et celles qui en ont facilité la naissance et permis le bon aboutissement.

Mes remerciements vont donc (dans l'ordre chronologique) à :

Mes parents, qui m'ont immergé dans un environnement littéraire, et poussé vers des études scientifiques,

Les créateurs d'ELIZA, de HAL et d'autres programmes spectaculaires, qui donnent du rêve en réalisant les leurs,

Bruno Petazzoni, Professeur de mathématiques et d'informatique, pour ses enseignements dispensés pendant mes jeunes années,

Christian Jacquelinet, Docteur en médecine et en linguistique informatique, qui m'a convaincu de reprendre la voie des études et de la recherche, douze ans après avoir obtenu mon diplôme d'ingénieur,

Laurence Danlos, Professeur de linguistique informatique à l'Université Paris 7, qui a accepté un étudiant atypique dans le cursus du Master de Recherche qu'elle dirige,

Sylvain Kahane, Professeur à l'Université Paris 10, qui jongle avec les constituants de la langue avec la dextérité du linguiste et la rigueur du mathématicien,

Alexis Nasr, Professeur à l'Université Paris 7, dont je partage la conviction qu'en TAL, une idée est d'autant plus intéressante quand elle est implémentée d'une façon efficiente,

Ceux et celles qui ont eu la gentillesse de relire ce document et de formuler commentaires, remarques et critiques constructives,

Last but not least, Carole, Cerise et Emilie, à qui j'ai volé un peu de temps cette année, qui me soutiennent par leur compréhension et leurs encouragements.

Table des matières

Cadre général	1
Introduction	1
Objectifs	1
Analyse syntaxique et lexicale d'une phrase d'exemple	2
Tableau d'ensemble	5
L'interface syntaxe / sémantique	7
Introduction	7
Classes de verbes	7
Rôles thématiques	7
Contraintes de sélection	10
Analyse sémantique de la phrase d'exemple	11
Présentation des ressources utilisées	13
Introduction	13
Link Grammar Parser	13
WordNet	14
eXtended WordNet	17
SUMO	18
VerbNet	19
Intégration de ces ressources	20
Relations entre ces ressources	23
Introduction	23
Interface entre VerbNet et WordNet	23
Correspondance entre rôles thématiques et sommet de la hiérarchie des noms	24
Enrichissement du Link Grammar Parser grâce à VerbNet et WordNet	26
De la syntaxe à la sémantique par identification de schémas	27
Introduction	27
Quelques rappels sur PROLOG	27
Structures syntaxique et sémantique	29
Mécanisme d'identification de schémas	33
Mise en œuvre	37
Introduction	37
Utilisation de VerbNet pour identifier des constructions typiques de verbes	37
Traduction en PROLOG d'une description de classe de verbes	39
Automatisation de la traduction des descriptions de VerbNet en PROLOG	47
Désambiguïsation des noms dans un groupe nominal	50
Prise en compte simultanée des contraintes possibles	53
Indice de vraisemblance d'une interprétation de phrase	55
Conclusion	59
Résultats obtenus	59
Limitations actuelles	59
Améliorations possibles	59
Travaux analogues	60
Synthèse et perspectives	60
Bibliographie	61

Annexe : de VerbNet à PROLOG**63**

Exemple de fichier VerbNet : murder.xml.....	63
Le programme PROLOG correspondant : murder.pp.....	65

Cadre général

Introduction

La compréhension de textes par la machine est une suite d'opérations complexes. Elle peut être décomposée en plusieurs niveaux. A chaque niveau peut correspondre une structure de représentation dédiée. Il n'existe pas de consensus absolu sur le nombre de structures, d'étapes de traitement, et leur nature exacte ; toutefois, les structures suivantes reviennent fréquemment, certaines étant très consensuelles, d'autres moins :

- Structure lexicale (pour reconnaître les mots en incluant les formes fléchies),
- Structure syntaxique, pour identifier les relations entre mots,
- Structure syntaxique profonde¹,
- Structure identifiant les coréférences (par résolution de chaînes anaphoriques),
- Structure identifiant les entités nommées (personnes, lieux, organisations...),
- Structure syntaxique profonde avec désambiguïsation (au moins partielle) des mots,
- Structure sémantique avec un étiquetage des rôles sémantiques...

Disposer d'une structure syntaxique ou sémantique correcte est un préalable à des traitements plus complexes sur un texte :

- Extraction d'information,
- Questions / réponses,
- Résumé,
- Traduction,
- « Compréhension » en profondeur (voir par exemple [Mueller, 2003]).

Objectifs

Notre but est la conception et la réalisation d'une boîte à outils modulaire d'analyse de textes anglais, avec différents niveaux clairement identifiés. Le choix de travailler sur la langue anglaise est basé sur la présence de nombreuses ressources déjà disponibles dans cette langue. Toutefois, les résultats obtenus pourront être généralisés au Français ou à d'autres langues.

Le travail décrit ici porte principalement sur l'interface entre le niveau syntaxique et le niveau sémantique. Nos objectifs sont multiples :

- Définir une forme de représentation syntaxique,
- Définir une forme de représentation sémantique,
- Trouver des règles de passage du niveau syntaxique au niveau sémantique,
- Effectuer une désambiguïsation au moins partielle, en conservant les ambiguïtés si elles ne peuvent pas être levées,
- Affecter à chaque interprétation syntaxico-sémantique un indice de vraisemblance.

¹ Utilisée par exemple dans la Théorie Sens Texte : « il + a + mangé » → « LUI (masc, sing) + MANGER(passé composé) »

La désambiguïsation que nous souhaitons effectuer se fait à un double niveau :

- Au niveau syntaxique, pour identifier les meilleures interprétations syntaxiques,
- Au niveau lexical, pour trouver le(s) sens acceptable(s) de chaque mot, dans le contexte d'une construction syntaxique donnée.

L'originalité et la spécificité de notre approche résident en une démarche d'intégration de ressources de large couverture, issues de plusieurs projets de recherche¹. Autour de ces ressources, nous construisons un analyseur modulaire qui effectue trois niveaux d'opérations :

- Une **analyse syntaxique**, pour construire les différentes interprétations syntaxiques d'une phrase,
- Une **analyse lexicale**, capable de donner les différents sens d'un mot, et pour chaque sens d'en indiquer le domaine (par exemple : *Humain, Animal, Machine, Idée...*) et les relations (synonymie, généralisation, spécialisation...) avec d'autres sens de mots,
- Une **analyse sémantique**, en mesure de créer des relations sémantiques, notamment d'extraire les différents rôles thématiques autour d'un prédicat verbal (ou autre).

Nous allons étudier :

- Quelle interopérabilité est possible entre ces ressources ?
- Comment ces ressources peuvent s'enrichir mutuellement ?
- Comment leur emploi conjoint contribue à la réalisation de l'interface syntaxe/sémantique, et permet de diminuer l'explosion combinatoire ?

Analyse syntaxique et lexicale d'une phrase d'exemple

Nous allons dérouler un exemple pour illustrer quelques-uns des problèmes qui se posent d'une façon concrète. Ceci permettra de préciser l'idée générale de notre approche. Nous allons décrire brièvement l'analyse de la phrase : "*the cultivator eliminated the beetles with pesticide*" (« le cultivateur a éliminé les scarabées avec du pesticide »).

Analyse syntaxique

Un analyseur syntaxique basé sur une grammaire de dépendances produit deux interprétations de cette phrase². Nous avons donc ici une première ambiguïté, de nature syntaxique.

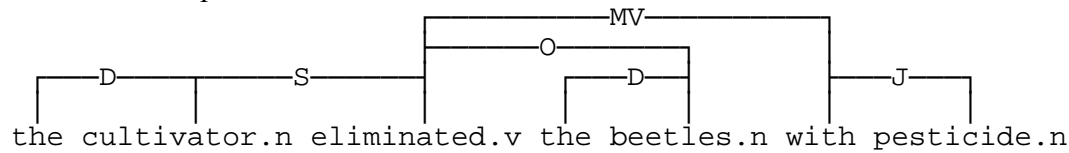
Notons que dans cet exemple simple, nous avons de la chance d'avoir *seulement* deux interprétations. On parle souvent des « forêts d'arbre » en sortie d'un analyseur syntaxique. Une phrase plus longue et plus complexe aurait, par explosion combinatoire, des centaines ou des milliers d'interprétations.

¹ Ces différentes ressources sont décrites dans le chapitre suivant. Pour en dire juste un mot ici :

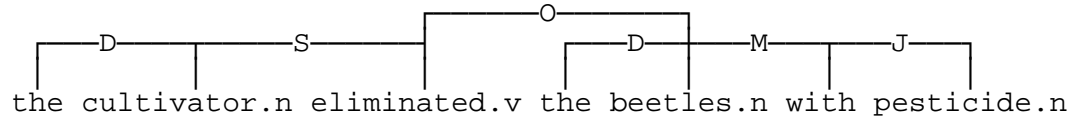
- L'analyseur syntaxique est essentiellement une encapsulation du **Link Grammar Parser**,
- La partie lexicale regroupe **WordNet**, **eXtended WordNet** et l'ontologie **SUMO**,
- L'analyseur sémantique est une construction originale, basée sur un ensemble de règles d'identifications de schémas, et utilisant **VerbNet**.

² Chaque interprétation est donnée sous forme d'un graphe composé de nœuds (les mots) reliés par des arcs étiquetés (les fonctions grammaticales). L'étiquette d'un lien vaut S (*subject*) entre sujet et verbe, D (*determiner*) entre déterminant et nom, O (*object*) entre verbe et complément d'objet direct, MV (*modifier verb*), etc.

La première interprétation est la plus compréhensible : « l'agriculteur a éliminé les scarabées en utilisant du pesticide ».



La seconde interprétation considère que le complément d'objet direct est un groupe nominal composé : « l'agriculteur a éliminé les-scarabées-qui-ont-du-pesticide ».



Notre enjeu consiste donc à affecter à la première interprétation un indice de vraisemblance sémantique plus grand qu'à la seconde.

La sémantique ne peut pas se déduire de la syntaxe seule

Un texte courant, même simple, contient souvent une part d'implicite. Dans beaucoup de cas, la connaissance du monde (c'est-à-dire l'utilisation permanente d'informations pragmatiques) est nécessaire pour désambiguïser les mots du texte.

La seconde interprétation syntaxique de la phrase d'exemple heurte le sens commun. Remarquons néanmoins qu'elle est tout aussi correcte (syntaxiquement) que la première. Elle deviendrait tout à fait compréhensible si on remplaçait « *pesticide* » par « *wings* » (ailes) : “*the cultivator eliminated the beetles with wings*” (« l'agriculteur a éliminé les-scarabées-qui-ont-des-ailes »). La différence d'interprétation vient du fait que :

- Du pesticide est un instrument pouvant servir à éliminer des insectes,
- Une aile est une partie du corps d'un insecte, et un scarabée est un insecte : « *beetles with wings* » forme un groupe nominal cohérent.

La sémantique ne peut donc pas se déduire uniquement de la syntaxe ; on a aussi besoin d'informations pragmatiques pour « comprendre » un texte.

Analyse lexicale

L'analyseur syntaxique associe à chaque mot une catégorie grammaticale :

- *cultivator*, *beetles* et *pesticide* sont reconnus en tant que formes de nom (extension **.n**),
- *eliminated* est reconnu en tant que forme de verbe (extension **.v**).

A ce stade, la partie du discours de chaque mot est reconnue (noms, verbes, adjectifs, adverbe...) mais leur sens précis, en cas de polysémie, n'est pas connu. Le nombre de sens d'un mot varie considérablement selon la précision du lexique utilisé.

Celui que nous utilisons indique que :

- Le mot anglais *cultivator* possède deux sens en tant que nom :
 - 1) Quelqu'un qui cultive le sol (un agriculteur ou cultivateur au sens usuel),
 - 2) Un instrument de ferme utilisé pour casser la surface du sol.

- Le verbe anglais *to eliminate* a six sens¹ :
 - 1) Finir / terminer (un cours par exemple)
 - 2) Tuer / éradiquer,
 - 3) Eliminer / rejeter (une possibilité),
 - 4) Eliminer / rejeter (du corps),
 - 5) Eliminer (d'un concours ou d'une course),
 - 6) Eliminer (une variable d'une équation).

- En fonction du contexte grammatical, *beetle* peut être un verbe (trois sens), un adjectif (un sens), ou un nom. Dans ce dernier cas, *beetle* veut dire :
 - 1) L'insecte scarabée,
 - 2) Un marteau à tête large.

- Le nom *pesticide* a, comme en français, un seul sens :
 - 1) Un produit chimique servant à tuer des animaux nuisibles.

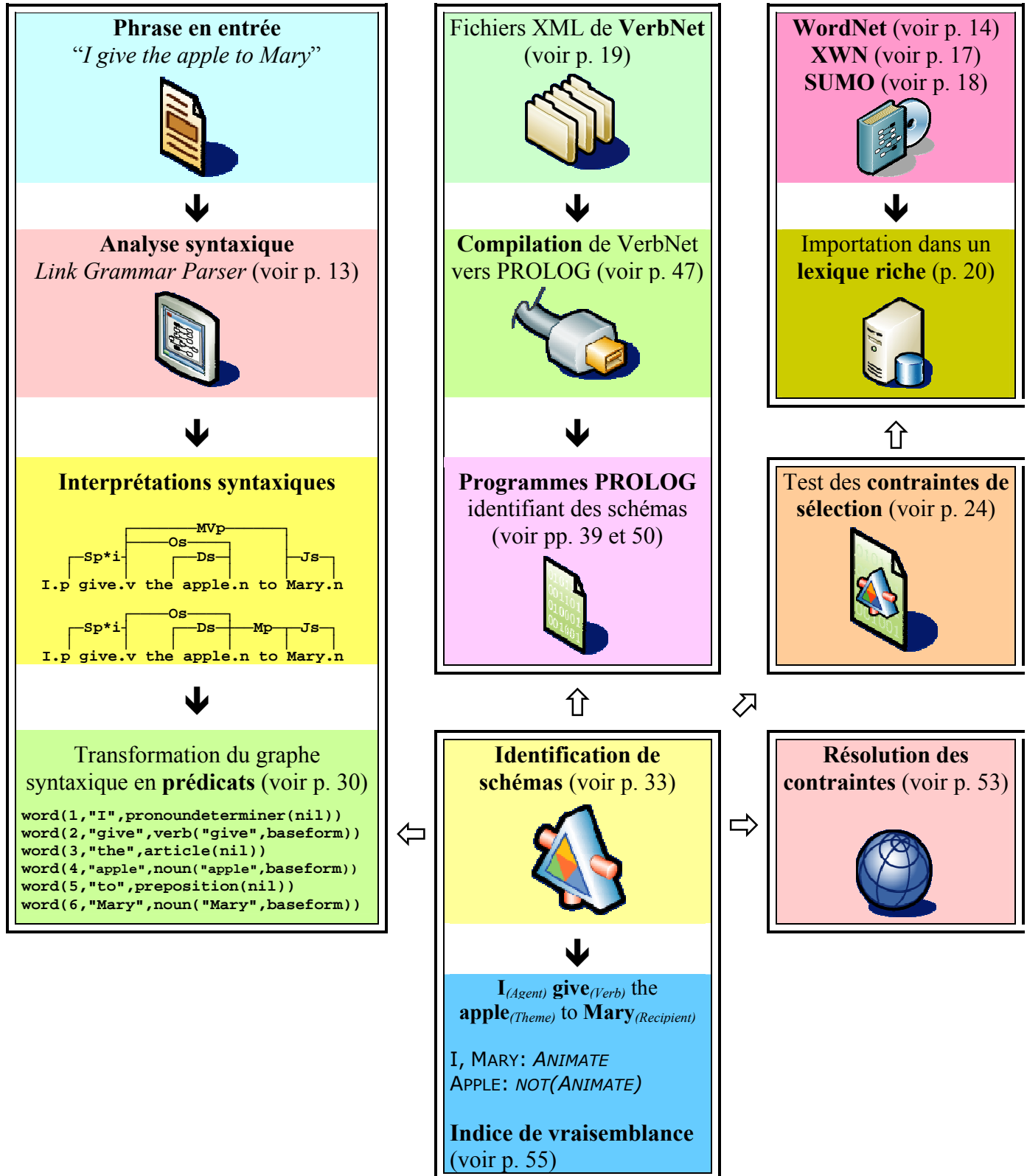
Le nombre d'interprétations possibles d'un texte est le nombre d'arbres syntaxiques multiplié par le produit du nombre de sens de chaque mot. Toutefois, certains sens de certains mots peuvent être incompatibles avec une structure syntaxique donnée, ce qui permet d'éliminer ces sens.

Avec notre phrase d'exemple, chaque interprétation syntaxique ayant 24 sens, nous avons 48 interprétations possibles. C'est à ce stade que l'analyseur sémantique rentre en jeu, pour réduire le nombre d'interprétations possibles, et calculer leur indice de vraisemblance.

¹ Rappelons que c'est le nombre de sens *donné par le lexique que nous utilisons*.

Tableau d'ensemble

Ce schéma décrit succinctement l'ensemble de notre démarche, et montre les interactions entre ressources utilisées ainsi que le processus d'analyse sémantique.



L'interface syntaxe / sémantique

Introduction

L'analyseur sémantique que nous avons conçu utilise un mécanisme très général, l'identification de schémas¹, décrit en détail page 33. Ce mécanisme se trouve notamment mis en œuvre pour détecter des constructions typiques de verbes et de noms.

Avant de continuer l'analyse de notre phrase d'exemple sur la partie sémantique, nous allons commencer par définir trois notions fondamentales pour la bonne compréhension de la suite : les classes de verbes, les rôles thématiques et les contraintes de sélection.

Classes de verbes

On appelle (dans VerbNet) **classe de verbes** un regroupement de verbes qui partagent un même comportement syntaxique et sémantique et qui, de ce fait, connaissent les mêmes constructions typiques.

Par exemple, la classe de verbes "**murder**" regroupe plusieurs verbes tels que : *to assassinate* (« assassiner »), *to eliminate* (« éliminer » au sens 2), *to execute* (« exécuter »), *to immolate* (« immoler »), *to liquidate* (« liquider »), *to massacre* (« massacrer »), etc.

Les verbes membres ne sont pas forcément tous synonymes entre eux. Dans la classe "**give**" par exemple, on retrouve « donner », « louer », « prêter », « rendre », « restituer »... qui partagent les mêmes constructions, mais avec des spécificités de sens.

Rôles thématiques

Les rôles thématiques font référence à la relation sémantique sous-jacente entre un prédicat et ses arguments. Ils ont été introduits à la fin des années 60 ([Gruber, 1965], [Fillmore, 1968], [Jackendoff, 1972]) de façon à créer un ensemble fini de types de participants en tant qu'arguments de prédicat. Ces rôles sont utilisés pour décrire les comportements lexicaux et syntaxiques des verbes.

Ces rôles sont indépendants de la construction syntaxique. Par exemple, dans les deux phrases suivantes, « Marie » a le rôle thématique *Patient* de l'action de frapper, et « Jean » a le rôle *Agent* :

- Jean frappe Marie,
- Marie est frappée par Jean,

Chaque argument du verbe (chaque actant) joue un rôle thématique. Il peut être, par exemple, *Agent*, *Patient*, *Thème*, *Instrument*, *Source*... de l'action ou de l'événement décrit par le verbe. Par exemple, la classe de verbes "**murder**" a trois constructions typiques :

- *Agent* élimine *Patient* (« Brutus tua Jules César »),
- *Agent* élimine *Patient* avec *Instrument* (« Brutus tua César avec un poignard »),
- *Instrument* élimine *Patient* (« le pesticide tua les insectes »).

¹ *Pattern recognition* en anglais.

Chaque argument d'un verbe est assigné à un unique rôle thématique au sein d'une classe de verbe. L'une des exceptions à cette règle concerne les classes contenant des verbes avec des arguments symétriques (comme dans « Jean et Marie discutent » ou « La France et l'Italie se touchent ») qui ont alors deux rôles tels qu'*Acteur1* et *Acteur2*. VerbNet définit une trentaine de rôles thématiques. Nous les détaillons ici, en donnant un exemple à chaque fois.

Acteur

Acteur est utilisé dans des classes de communication (“**chitchat**”, “**marry**”, “**meet**”) quand les deux arguments peuvent être considérés comme symétriques, comme dans « Pierre et Marie se fiancent ».

Agent

Agent est un instigateur actif d'une action ou d'un événement. *Agent* est généralement un humain ou un sujet animé ; il peut aussi être utilisé pour désigner un sujet ayant une volonté propre, comme une force ou une machine.

Pour identifier un rôle *Agent*, on peut utiliser :

- Le test de volonté (« Tom cassa volontairement la tasse » vs. « *Tom se sentit malade volontairement »),
- Le test de promesse (« Tom promet de casser la tasse » vs. « *Tom promet de se sentir malade »).

Attribut

Attribut de *Patient* ou de *Thème* fait référence à une caractéristique de quelque chose qui est en train de changer, comme dans « le prix du pétrole augmente ». *Attribut* a une contrainte de sélection *Scalaire* (définie par une quantité, une masse, une longueur, une heure, une température, etc.).

Bénéficiaire

Bénéficiaire désigne l'entité bénéficiant d'une action, généralement introduite par une proposition commençant par « pour », comme dans « Marie a créé un jouet pour le bébé ».

Cause

Cause est surtout utilisé par les classes de verbes psychologiques ou relatifs au corps, comme dans « les touristes ont admiré les tableaux » ou « cela compte pour moi ».

Destination

Destination est le point final ou la direction d'un déplacement (introduit par « vers » ou « sur »...). Il est utilisé dans des classes telles que “**banish**”, “**send**”, “**carry**”, comme dans « le roi a exilé le capitaine sur l'île ».

Emplacement

Emplacement est un participant qui exprime une destination, une origine, un endroit, généralement introduit par un complément circonstanciel de lieu, comme dans « le bateau apparaît à l'horizon ».

Etendue

Le rôle *Etendue* est utilisé pour spécifier l'intervalle ou le degré de changement, comme dans « le prix du pétrole augmente de 10% ».

Expérimentateur

Expérimentateur est un participant caractérisé par le fait d'avoir conscience de quelque chose ou d'expérimenter quelque chose, comme dans « Pierre souffre ». Plusieurs verbes psychologiques ou d'émotion ont un *Expérimentateur* pour sujet (aimer, admirer...) ou pour objet (amuser, perturber...).

Heure

Heure est un rôle spécifique à la classe “**begin**” pour exprimer un horaire, comme dans « la réunion commence à 16 heures ».

Instrument

Instrument est un objet ou une force physique qui provoque un changement dans quelque chose, généralement par contact direct, comme dans « il toucha la balle avec la raquette ».

Matériau

Matériau est le point de départ d'une transformation, utilisé par exemple dans “**build**”, comme dans « Marie a sculpté une jolie statuette avec le bout de bois ».

Montant

Montant est utilisé pour représenter une valeur, une somme d'argent ou l'équivalent (par exemple dans les classes “**build**”, “**get**”, “**obtain**”), comme dans « ils lui ont facturé 10 € ».

Objectif

Objectif est le participant vers lequel le mouvement a lieu, comme par exemple dans « les martiens rentrent à la maison ».

Patient

Patient est un participant soumis à un processus ou affecté par une action. L'emphase est mise sur le changement d'état. Le Patient peut être sujet (« la glace a fondu ») ou objet du verbe (« il chauffa l'eau »). *Patient1* et *Patient2* sont aussi utilisés en cas de rôles symétriques (« la crème et l'œuf se mélangèrent »).

Pour déterminer un rôle *Patient*, un test possible est « qu'est-ce qui est arrivé à X ? ».

Prédicat

Prédicat est la partie de l'énoncé qui exprime ce qui est dit à propos du *Thème*, comme dans « il se vante d'être l'homme le plus fort du monde ».

Produit

Produit est le résultat final d'une transformation, comme dans « David a construit une maison ».

Réceptient

Réceptient est un participant qui est la destination du transfert d'une entité concrète ou abstraite, comme dans « Jean a passé le sel à Marie ». Ce rôle autorise toujours une contrainte de sélection de type *Animé* et parfois *Organisation*.

Source

Source est le point de départ du mouvement, généralement introduit par une préposition (« les martiens viennent d'une autre planète »).

Stimulus

Stimulus est l'événement ou l'objet qui provoque une réaction chez un *Expérimentateur*, comme dans « l'orage effraya les enfants ». Ce rôle n'impose généralement pas de contrainte de sélection.

Thème

Thème est un participant qui est localisé dans un endroit ou qui se déplace d'un endroit à l'autre. L'emphase est mise sur la localisation ou la possession. (« Jean donne un ballon », « Marie marche »...)

Thème1 et Thème2 peuvent être utilisés en cas de rôles symétriques, comme dans « Jean échange le livre pour une revue ».

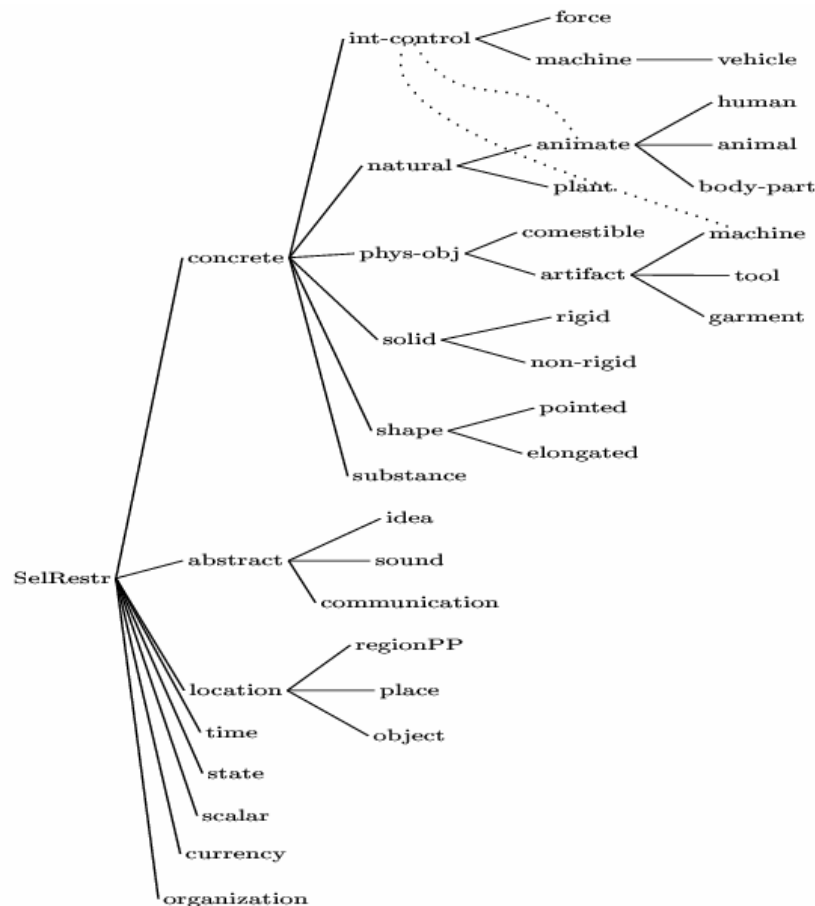
Topique

Topique est le rôle utilisé par les verbes de communication pour exprimer le thème ou le sujet d'une conversation ou d'un transfert de message, comme dans « Pierre a mis en garde Marie contre les effets de la colère ».

Contraintes de sélection

Les rôles thématiques peuvent avoir des contraintes de sélection, qui en restreignent les sens possibles. Par exemple, dans “**murder**”, *Agent* a une contrainte de sélection *Humain* ou *Animé*. VerbNet en propose une quarantaine, organisées selon un graphe d'héritage, comme le montre le schéma ci-après.

L'un des enjeux de notre analyseur sémantique, pour désambiguïser le sens des mots, sera d'établir une correspondance entre les mots du lexique et la hiérarchie des contraintes de sélection. (Ce point est approfondi page 24.)



Analyse sémantique de la phrase d'exemple

Le cœur de notre analyseur sémantique cherche à identifier des constructions qui ont un sens. Le module principal identifie les différentes constructions typiques d'un verbe, avec les arguments associés. Pour revenir à notre exemple, *to eliminate* est membre de deux classes de verbes :

- Le deuxième sens de *to eliminate* appartient à la classe de verbes “**murder**”, dont la signature¹ en tant que prédicat est : *Verbe(Agent, Patient, Instrument)*,
- Les autres sens de *to eliminate* sont membres de la classe de verbes “**remove**”, dont la signature en tant que prédicat est : *Verbe(Agent, Thème, Source)*.

Si on recherche dans notre phrase d'exemple les constructions typiques, et les rôles thématiques associés, on trouve quatre interprétations possibles :

- Pour la première construction syntaxique :

“**murder#1**” the **cultivator**_(Agent) **eliminated**_(Verb) the **beetles**_(Patient) with **pesticide**_(Instrument)
 (Contraintes de sélection satisfaites : *cultivator*_(animate + concrete) *beetles*_(animate))

“**remove#1**” the **cultivator**_(Agent) **eliminated**_(Verb) the **beetles**_(Theme) with pesticide
 (Contraintes de sélection satisfaites : *cultivator*_(internal_control))

¹ Ou, dit autrement, comme liste d'arguments.

- Pour la seconde construction syntaxique :

“**murder#2**” the **cultivator**_(Agent) **eliminated**_(Verb) the **beetles**_(Patient) with pesticide
(Contraintes de sélection satisfaites : cultivator_(animate + concrete) beetles_(animate))

“**remove#2**” the **cultivator**_(Agent) **eliminated**_(Verb) the **beetles**_(Theme) with pesticide
(Contraintes de sélection satisfaites : cultivator_(internal_control))

Notre hypothèse est que l’indice de vraisemblance sémantique d’une interprétation de phrase est proportionnel au nombre de rôles thématiques qu’on peut identifier. L’indice le plus élevé est donc associé, pour la phrase d’exemple, à l’interprétation “**murder#1**”.

Cette interprétation a donc la probabilité la plus élevée d’être correcte. L’unique sens compatible du verbe *to eliminate* est le deuxième sens. Des contraintes de sélection s’imposent alors :

- Le nom *cultivator* est *Animé* et *Concret* ; il ne peut donc s’agir que du premier sens,
- Le nom *beetles* est *Animé*, ce qui en élimine le second sens.

Nous avons donc identifié une interprétation de la phrase qui semble plus correcte que les autres, et où le sens de chaque mot a été correctement reconnu.

Présentation des ressources utilisées

Introduction

Plusieurs ressources ont été utilisées conjointement dans le cadre du mémoire. Elles sont présentées ici brièvement, de façon à donner une idée d'ensemble. Le contenu de chaque ressource est ensuite détaillé dans un sous-chapitre dédié, qui décrit son utilisation effective dans le cadre de notre interface syntaxe / sémantique.

Analyseur syntaxique

Le *Link Grammar Parser* est l'implémentation d'un analyseur syntaxique basé sur une grammaire de dépendances.

Analyseur lexical

Trois ressources lexicales ont été fusionnées au sein d'une unique base de données, pour être facilement utilisées ensemble :

- **WordNet** est un dictionnaire riche, qui détaille les relations entre mots. Les noms et les verbes y sont hiérarchisés avec une structure de généralisation / spécialisation,
- **Extended WordNet** est un prolongement de WordNet qui précise les relations entre mots par analyse de la définition textuelle de chaque entrée,
- **SUMO** (*Standard Upper Merged Ontology*) est une proposition d'ontologie pour modéliser formellement les concepts de haut niveau de la connaissance humaine.

Analyseur sémantique

VerbNet décrit les classes de verbes anglais en explicitant leurs constructions syntaxiques et sémantiques.

Link Grammar Parser

Présentation

Après une étude de différents analyseurs syntaxique, notre choix s'est porté sur le *Link Grammar Parser*. Nous rejoignons en cela les conclusions de l'étude comparative effectuée dans [Aubin, 2003].

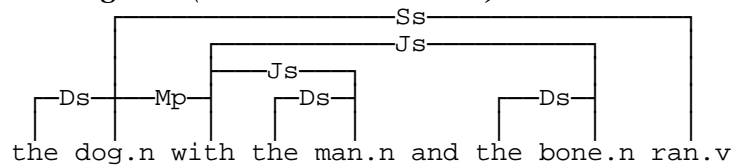
Le *Link Grammar Parser* est un analyseur syntaxique de la langue anglaise, basé sur la théorie des grammaires de dépendance. Cet outil est dû à Davy Temperley, Daniel Sleator et John Lafferty, de l'Université de Carnegie Mellon. Il est téléchargeable sur <http://bobo.link.cs.cmu.edu/link>. Son code source (en langage C¹) est utilisable librement. Sa description complète peut être trouvée dans [Sleator & Temperley, 1991].

Partant d'une phrase, cet analyseur en détermine la structure syntaxique, qui consiste en un ensemble de liens typés reliant des paires de mots. La grammaire de dépendances utilisée distingue 107 types de liens. L'analyse peut donner une forêt d'arbres, chaque arbre étant pondéré par un « coût syntaxique ».

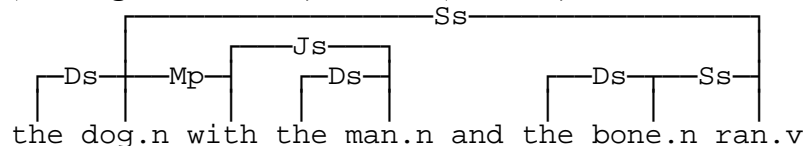
¹ C est un langage de bas niveau, parfois piègeur (possibilité de manipulation directe de la mémoire), mais offrant des performances optimales et une bonne portabilité d'une plate-forme à l'autre.

Par exemple, la phrase “*the dog with the man and the bone ran*” donne deux analyses, correspondant à la distribution possible autour de la conjonction de coordination *and* :

The dog with (the man and the bone) ran



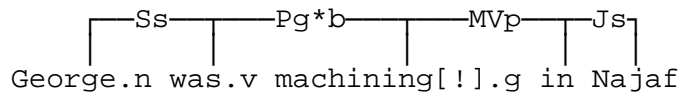
(The dog with the man) ran and (the bone) ran



Caractéristiques

Robustesse et rapidité de l'analyseur

L'analyseur offre de bonnes performances pour un analyseur en profondeur, ainsi qu'une excellente robustesse. Il traite avec succès des phrases complexes, et est tolérant à la présence de mots inconnus. Quand il en rencontre, il essaie de déterminer sa partie du discours en fonction du contexte. Par exemple, dans la phrase suivante, *machining* n'est pas présent dans le lexique de référence (comme l'indique l'annotation [!]), mais est supposé être le gérondif d'un verbe (extension .g).



Stockage externe de la grammaire

La grammaire décrit d'une façon relativement complète la langue anglaise courante. Elle est stockée indépendamment du code, dans des fichiers textes. Ces fichiers textes contiennent également le lexique de référence (approximativement 60 000 mots). La grammaire et le lexique sont donc étroitement imbriqués.

Ce stockage externe permet de modifier ou d'enrichir la grammaire ou le lexique. En revanche, le formalisme de stockage est propriétaire, et plutôt complexe à maîtriser.

Utilisation dans le cadre de nos travaux

Nous utilisons le *Link Grammar Parser* pour effectuer l'analyse syntaxique d'une phrase avant de passer à l'étape d'analyse sémantique.

WordNet

Un traitement automatique portant sur des textes courants (corpus de *news* par exemple) ne peut s'appuyer sur un lexique restreint de quelques centaines de mots. Il doit s'appuyer sur un lexique riche, structuré d'une façon qui le rende exploitable par la machine.

De nombreuses ressources électroniques de ce type, de large couverture et gratuites, sont disponibles en langue anglaise. WordNet est probablement la plus utilisée d'entre elles.

Présentation

WordNet est un projet mené à l'université de Princeton par une équipe de psycholinguistes, de linguistes et d'informaticiens. Ce projet a été amorcé en 1985, sous l'impulsion de G. Miller, sur la base du corpus Brown¹. Sa description complète se trouve dans [Miller, 1995].

Ce projet est le fruit d'un travail manuel de plusieurs années-homme. WordNet est probablement la ressource la plus largement utilisée aujourd'hui dans les projets TAL en langue anglaise², par exemple pour effectuer de l'extraction d'informations, ou des questions/réponses.

WordNet offre un réseau sémantique complet de la langue anglaise. Les nœuds sont constitués par des ensembles de synonymes (ou *synsets*). Chaque ensemble de synonymes correspond au sens d'un (ou plusieurs) **lemme** (mot ou unité polylexicale) ; un *synset* est défini d'une façon différentielle par les relations qu'il entretient avec les sens voisins. Une définition du *synset* en langue naturelle vient compléter ces relations. Une recherche à partir d'un mot fournit la liste des noms, verbes, adverbes et adjectifs associés³.

WordNet est téléchargeable sur <http://wordnet.princeton.edu>.

Exemples de relations présentes

WordNet définit également un ensemble riche de relations, en distinguant les relations entre *synsets* des relations entre lemmes. Un concept de nom est associé à :

- Des hyperonymes (ancêtres) et hyponymes (descendants),
- Des holonymes (noms incluant) et méronymes (noms inclus),
- Des antonymes (contraires),
- Etc.

Hyperonymes et hyponymes

Par exemple, partant du sens le plus général du mot *cat* (« chat »), on obtient une liste ordonnée d'ancêtres et de descendants :

¹ Plusieurs versions se sont succédées (version 1.0 en juin 1991, version 1.7 en 2001, version 2.0 en août 2003). La version 2.1, la plus récente, disponible depuis juin 2005 apporte une notion d'instance en plus de la notion d'héritage (« *George Washington* » est une instance de « *Président américain* »). Les prochaines versions devraient apporter des enrichissements dans trois domaines : morphologie dérivationnelle, désambiguïsation des termes des définitions (ce que fait déjà eXtended WordNet), *clustering* par domaine.

² Consulter, par exemple, <http://wordnet.princeton.edu/links>.

³ Par exemple, à partir du mot *glasses*, WordNet produit :

- Un sens pour le nom *glasses*,
- Sept sens pour le nom *glass*,
- Cinq sens pour le verbe *to glass*.

Proxem Browser

Search Word GO Options

Hyperonym

- ⓘ feline, felid -- (any of various lithe - bodied round - headed fissiped mammals many with retractile claws)
- ⓘ carnivore -- (terrestrial or aquatic flesh-eating mammal ; terrestrial carnivores have four or five clawed digits on each limb)
 - ⓘ placental, placental mammal, eutherian, eutherian mammal -- (mammals having a placenta ; all mammals except monotremes and marsupials)
 - ⓘ mammal -- (any vertebrate having the skin more or less covered with hair ; young are born except for the small subclass of monotremes and nourished with milk)
 - ⓘ vertebrate, craniate -- (animals having a bony or cartilaginous skeleton with a segmented spinal column and a large brain enclosed in a skull or cranium)
 - ⓘ chordate -- (any animal of the phylum chordata having a notochord or spinal column)
 - ⓘ animal, animate being, beast, brute, creature, fauna -- (a living organism characterized by voluntary movement)
 - ⓘ organism, being -- (a living thing that has (or can develop) the ability to act or function independently)
 - ⓘ living thing, animate thing -- (a living (or once living) entity)
 - ⓘ object, physical object -- (a tangible and visible entity ; an entity that can cast a shadow ; "it was full of rackets, balls and other objects")
 - ⓘ entity -- (that which is perceived or known or inferred to have its own physical existence)

Hyponym

- ⓘ domestic cat, house cat, Felis domesticus, Felis catus -- (any domesticated member of the genus felis)
- ⓘ wildcat -- (any small or medium - sized cat resembling the domestic cat and living in the wild)

Cache size : Synsets 190 Relations 1255 Lemmas 360 Relations 83

Holonymes et méronymes

Avec ces relations, on peut déterminer qu'un chat a des pattes, un pelage, une queue...

Proxem Browser

Search Word GO Options

HasPart

(Inherited from ⓘ feline, felid)

- ⓘ paw -- (a clawed foot of an animal especially a quadruped)
 - ⓘ pad -- (the foot or fleshy cushion - like underside of the toes of an animal)

(Inherited from ⓘ mammal)

- ⓘ coat, pelage -- (growth of hair or wool or fur covering the body of an animal)
- ⓘ hair, pilus -- (any of the cylindrical filaments characteristically growing from the epidermis of a mammal ; "there is a hair in my soup")

(Inherited from ⓘ vertebrate, craniate)

- ⓘ belly -- (the underpart of the body of certain vertebrates such as snakes or fish)
- ⓘ caudal appendage -- (tail especially of a mammal posterior to and above the anus)
- ⓘ digit, dactyl -- (a finger or toe in human beings or corresponding part in other vertebrates)
 - ⓘ nail -- (horny plate covering and protecting part of the dorsal surface of the digits)
 - ⓘ half-moon, lunula, lunule -- (the crescent-shaped area at the base of the human fingernail)
 - ⓘ matrix -- (the formative tissue at the base of a nail)
 - ⓘ phalanx -- (any of the bones (or phalanges) of the fingers or toes)
- ⓘ rib, costa -- (any of the 12 pairs of curved arches of bone extending from the spine to or toward the sternum in humans (and similar bones in most vertebrates))
 - ⓘ costal cartilage -- (the cartilages that connect the sternum and the ends of the ribs ; its elasticity allows the chest to move in respiration)
- ⓘ tail -- (the posterior part of the body of a vertebrate especially when elongated and extending beyond the trunk or main part of the body)

Cache size : Synsets 190 Relations 1255 Lemmas 360 Relations 83

Caractéristiques

Couverture large et en profondeur de la langue anglaise

WordNet version 2.0 compte 203 000 lemmes définissant 115 000 *synsets*, 278 000 relations conceptuelles (entre *synsets*) et 70 000 relations lexicales (entre lemmes).

Profusion de sens pour un mot donné

La contrepartie de cette importante couverture est que WordNet est très (trop ?) précis dans le sens des définitions. On a ici une granularité très fine (les anglo-saxons parlent de *fine-grained definitions*) par opposition à une granularité grossière (*coarse-grained definition*). Par exemple, le verbe *to give* (« donner ») n'a pas moins de 44 sens. Une telle profusion de sens ne facilite pas forcément une tâche de désambiguïsation au niveau lexical.

Absence de relations pragmatiques

L'une des limites de WordNet est de ne pas matérialiser d'une façon formelle tout le sens contenu dans les définitions des termes. Par exemple, l'information qu'un chat ne rugit pas ne se retrouve explicitement dans aucune relation. De même, des relations pragmatiques telles que *savon / bain* sont actuellement absentes de WordNet.

Utilisation dans le cadre de nos travaux

WordNet est notre point de départ pour alimenter un lexique utilisable par la machine. Ce lexique est utilisé :

- D'une part pour déterminer les différents sens d'un mot donné,
- D'autre part pour rechercher quels sens d'un nom vérifient des contraintes de sélection (par exemple, quels sens du nom *chat* correspondent à un *Animal* ?).

eXtended WordNet

Présentation

XWN (eXtended WordNet) est un projet de l'université de Dallas, qui enrichit les relations du dictionnaire. Ce projet utilise WordNet, plus précisément les définitions de chaque *synset*. XWN effectue une analyse syntaxique de chaque définition (avec désambiguïsation lexicale), et la transforme en forme logique.

Par exemple, le terme *cousin* a la définition suivante "*the child of your aunt or uncle*" (« l'enfant de votre tante ou de votre oncle »), avec pour analyse syntaxique :

```
(TOP (S (NP (NN cousin) )
(VP (VBZ is)
(NP (NP (DT the) (NN child) )
(PP (IN of)
(NP (PRP$ your) (NN aunt) (CC or) (NN uncle) ) ) ) )
(. . ) ) )
```

Ainsi que la forme logique suivante (dont l'utilisation est détaillée page 26) :

```
cousin:NN(x1) -> child:NN(x1) of:IN(x1, x4) aunt:NN(x2) or:CC(x4, x2, x3) uncle:NN(x3)
```

Caractéristiques

Les informations présentes dans XWN sont de qualité *gold* (validé humainement), *silver* (accord entre deux programmes d'analyse automatique) ou *normal*. Du fait de la complexité de la tâche, et de l'absence de contrôle par un œil humain de l'ensemble des informations, il est sage de penser que seule l'information de qualité *gold* est valide, les autres contenant souvent des contresens.

eXtended WordNet est téléchargeable sur <http://xwn.hlt.utdallas.edu>.

Utilisation dans le cadre de nos travaux

eXtended WordNet nous sert, dans le cadre de l'analyse sémantique, à déterminer si un nom a un attribut particulier (*rigide*, *allongé*, *pointu*, etc.). Pour ce faire, nous recherchons dans les mots définissant le nom (ou ses ancêtres) un adjectif de ce type. Ce point est décrit page 25.

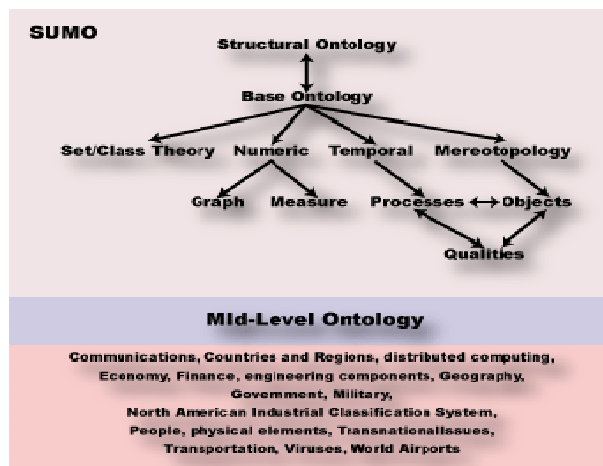
SUMO

Présentation

Les ontologies sont des artefacts construits en fonction d'une tâche précise. Force est de constater qu'une ontologie donnée ne semble pas pouvoir être facilement réutilisée pour une tâche autre que celle qui a motivé sa construction originelle.

Il découle de ce constat de nombreuses recherches sur la réutilisabilité du « haut » des ontologies, avec pour argumentaire : puisqu'il est difficile, voire impossible, de réutiliser directement des ontologies, trop proches de vues détaillées qu'on peut avoir sur un domaine, intéressons-nous au « haut » de l'ontologie. Cette *Upper Ontology* répertorie et organise de grandes catégories de la pensée ou de la société humaine qui devraient pouvoir être réutilisables dans de très nombreuses applications et être alors « génériques ».

L'objectif du groupe SUO17 (*Standard Upper Ontology*) est de réfléchir, puis soumettre à la normalisation, la constitution d'un haut d'ontologie qui se voudrait universel pour les grandes catégories d'objets et de pensées. Le résultat est SUMO (*Suggested Upper Merged Ontology*), qui vise à s'imposer en tant que standard, et commence à être utilisée notamment pour le Web sémantique. MILO (*Mid-Level Ontologies*) est un ensemble d'ontologies multi domaines, de niveau intermédiaire, créées en se basant sur SUMO.



SUMO est écrit en langage SUO-KIF, dérivé simplifié de KIF (*Knowledge Interchange Format*), qui est un langage équivalent à la logique du premier ordre.

SUMO, un mappage de SUMO vers WordNet, et MILO (les ontologies de niveau intermédiaire) sont téléchargeables sur <http://www.ontologyportal.org>.

Utilisation dans le cadre de nos travaux

Nous avons utilisé SUMO pour facilement identifier, dans WordNet, les sens d'un nom relatifs à un domaine donné. Par exemple, on obtient deux significations de *cat* (« chat ») en tant que « félin ». Cette possibilité de regrouper les sens de noms par domaine permet de se servir de WordNet aussi bien avec une approche grossière (*coarse-grained definition*) que fine (*fine-grained definitions*), ce qui simplifie la désambiguïsation (voir page 25).

VerbNet

Présentation

VerbNet est un lexique des classes de verbes anglais. C'est un projet de l'Université de Pennsylvanie, mené sous l'impulsion de Martha Palmer. Son objectif est de regrouper par classe les verbes partageant les mêmes comportements syntaxiques et sémantiques. En cela, c'est un prolongement des travaux de [Levin, 1993]. VerbNet est téléchargeable sur <http://www.cis.upenn.edu/group/verbnet>, sous forme de 192 descriptions stockées en XML¹. On peut en trouver une description détaillée dans [Kipper-Schuler, 2003].

Caractéristiques

Comme nous l'avons déjà vu, une classe de verbes regroupe plusieurs verbes, et identifie des rôles thématiques avec d'éventuelles contraintes de sélection. Elle décrit plusieurs constructions typiques (des « *frames* ») des verbes membres. La sémantique de l'action ou de l'événement est également précisée. Par exemple, pour « **murder** », au démarrage de l'événement, *Patient* est vivant (*alive(start(E), Patient)*), à la fin de l'événement, *Patient* n'est plus vivant (! *alive(result(E), Patient)*).

Des sous-classes permettent de décrire d'éventuelles spécialisations d'une classe.

La version actuelle de VerbNet (1.5) distingue 192 classes, qui regroupent 3880 sens de verbes. Il existe une traçabilité de VerbNet vers WordNet ainsi que vers XTAG².

Utilisation dans le cadre de nos travaux

Nous utilisons VerbNet en deux temps, pour la phase d'analyse sémantique :

- Dans une étape préparatoire, l'analyseur sémantique traduit les descriptions en XML de classes de verbes en programmes en langage PROLOG³.

¹ *eXtensible Markup Language* : un langage à balises normalisé, héritier de SGML. XML permet de représenter dans un document textuel tout type de données structurées complexes. C'est l'espéranto actuel pour stocker de l'information. Chaque plate-forme de développement informatique offre de nombreux outils pour lire et écrire de tels fichiers, valider leur contenu, transformer leur contenu, etc. Ces frameworks sont eux aussi standardisés. Pour plus d'informations, consulter <http://www.w3.org/XML/>.

² XTAG est une grammaire TAG (*Tree Adjoining Grammar*) lexicalisée développée à l'Université de Pennsylvanie. Pour plus d'informations, consulter <http://www.cis.upenn.edu/~xtag/>.

³ PROLOG (**PRO**grammation **LOG**ique) est un langage déclaratif utilisant le mécanisme d'unification avec retour arrière. Il est souvent utilisé dans des applications d'Intelligence Artificielle. Un programme PROLOG se constitue de prédicats décrivant des faits ou des règles. On utilise un tel programme pour chercher si un but donné est satisfait ou non. PROLOG est donc parfaitement adapté pour identifier des schémas. Le lecteur pourra trouver des rappels sur PROLOG page 27.

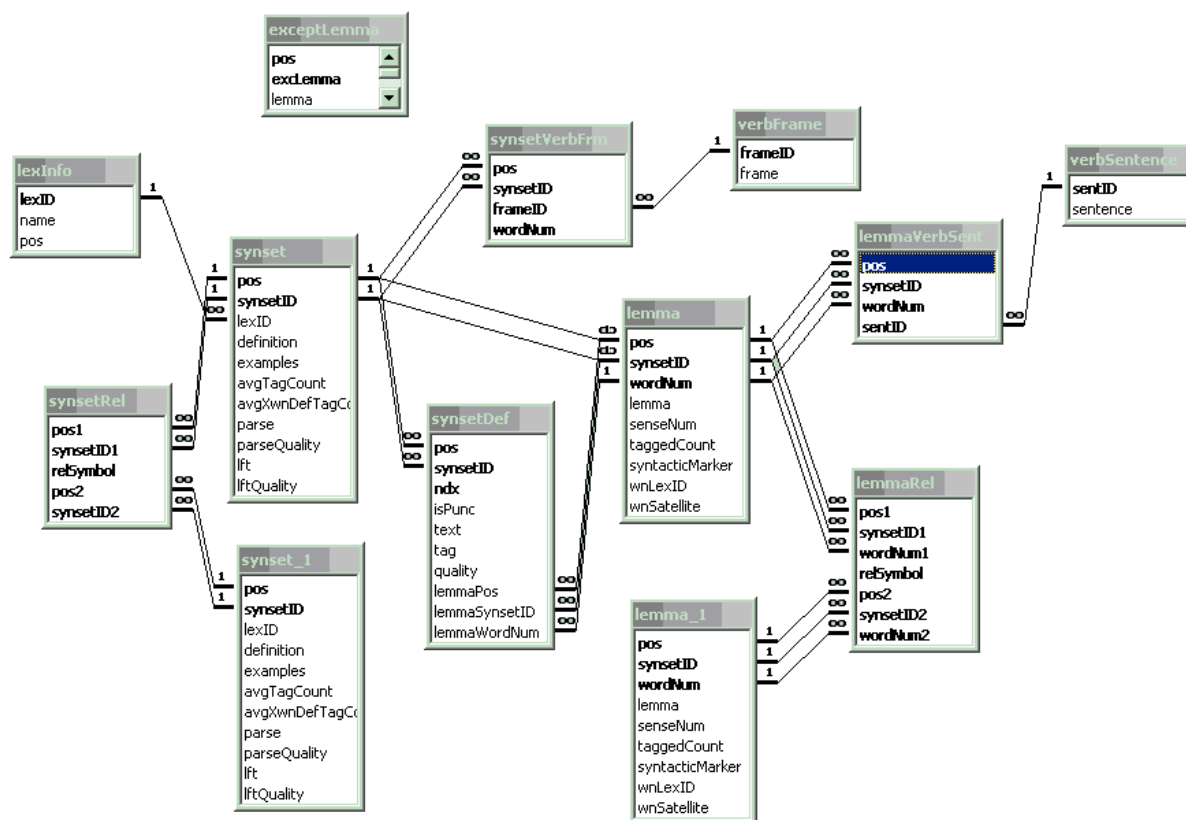
- Lors de l'analyse effective d'un texte, ces programmes utilisent le mécanisme d'unification pour identifier des schémas, de façon à reconnaître les constructions typiques de verbes dans une phrase fournie en entrée.

Intégration de ces ressources

L'intégration de ces différentes ressources a nécessité un important travail d'ingénierie préalable.

Intégration dans une base de données lexicale unique

Les trois ressources lexicales ont été importées dans une base de données relationnelle, dont la structure est indiquée sur le schéma ci-dessous¹. La structure de cette base est proche du modèle de données de WordNet.



Le langage SQL² permet de faire des requêtes complexes. Par exemple, on peut extraire les mots les plus polysémiques, ou encore compter les verbes d'utilisation très rare...

¹ Quelques remarques sur la notation (très classique) utilisée : les rectangles matérialisent des **tables** qui stockeront des informations. Les lignes en gras sont les colonnes clés, ou l'**identifiant** de la table. Les traits entre tables sont des **relations**, qui garantissent l'application de **contraintes d'intégrité** (par exemple, on ne peut pas créer un *lemma* sans le rattacher à un *synset*). Le nombre 1, ou le symbole infini, à chaque bout d'une relation précise sa **cardinalité** (ou multiplicité) : par exemple, un *synset* a plusieurs *lemmas*, et un *lemma* est rattaché à un unique *synset*, etc.

² *Structured Query Language* : le langage ensembliste standard d'interrogation et de manipulation des bases de données relationnelles.

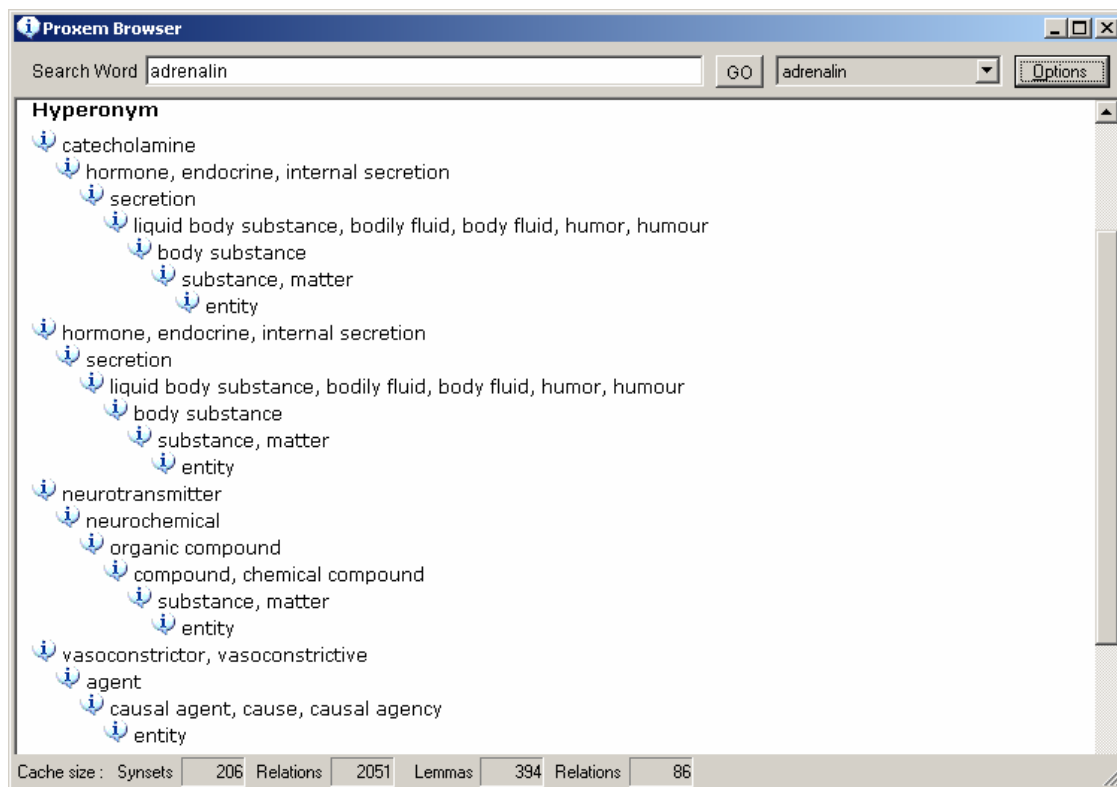
Création d'une couche d'objets d'accès aux données

Une couche d'objets a été écrite pour accéder au lexique. Elle présente deux avantages :

- Sa manipulation ne nécessite pas de connaissance du SQL.
- Elle cache les objets du lexique en mémoire pour permettre des manipulations de masse avec de bonnes performances.

Prise en compte de l'héritage multiple

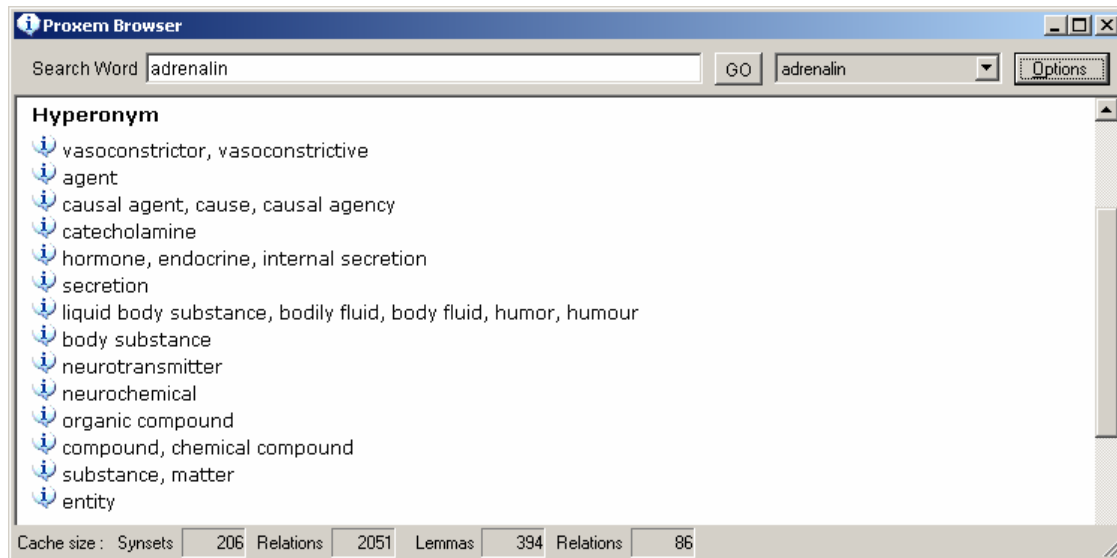
Dans WordNet, chaque nom ou verbe peut avoir plusieurs hyperonymes. Par exemple, le nom *adrenalin* a pour hyperonymes : *catecholamine*, *hormone*, *neurotransmitter*, *vasoconstrictor*¹.



Cette possibilité est utilisée pour peu de termes², néanmoins il est indispensable de la prendre en compte. Nous avons donc développé une « projection à plat » des hyperonymes, en transformant le DAG d'héritage en structure linéaire respectant un ordre partiel.

¹ On remarquera une redondance qui est une erreur de conception, *catecholamine* étant une spécialisation de *hormone*.

² Dans WordNet version 2.0, sur un total de 92 634 relations d'héritages entre *synsets* (noms et verbes), seulement 2 117 présentent un héritage multiple (et seulement 70 ont 3 hyperonymes directs ou plus).



Relations entre ces ressources

Introduction

Pour faire coopérer plusieurs ressources, il faut définir des interfaces entre chaque couple de ressources. L'information centrale pour laquelle nous avons besoin d'une correspondance entre les différentes ressources est le concept (*synset*) de WordNet¹. Des informations de traçabilité entre *synsets* existent déjà entre les trois ressources lexicales que nous utilisons (WordNet, eXtended WordNet et SUMO²), et également entre VerbNet et WordNet. Les autres interfaces restent à créer explicitement.

Par ailleurs, nous pouvons dans certains cas utiliser les informations d'une ressource pour en enrichir une autre. Notamment, le *Link Grammar Parser* peut bénéficier :

- Des cadres de description de VerbNet, qui peut être considéré comme une grammaire de référence pour les verbes. Les règles de dépendances du *LinkGrammar Parser* peuvent être affinées en utilisant ces cadres de description,
- Du lexique de WordNet, qui permet d'ajouter de nouveaux mots.

Cette partie décrit ces différentes interfaces, ainsi que les possibilités d'enrichissement entre ressources.

Interface entre VerbNet et WordNet

Correspondance des verbes entre VerbNet et WordNet

Dans VerbNet version 1.5, les 192 classes de verbes regroupent 3 880 membres, qui sont décrits dans des fichiers XML. Dans 82% des cas, un verbe membre d'une classe est accompagné d'un *synset*, qui permet d'identifier dans WordNet le bon sens du verbe.

Un examen détaillé montre que cette correspondance de VerbNet vers WordNet n'est cependant pas établie à 100% :

- Compte-tenu du caractère très (trop ?) précis des définitions de WordNet, un doute est parfois indiqué dans VerbNet sur la pertinence de l'association d'un verbe membre avec un *synset* donné³,
- Certains membres ne précisent pas de *synset*, alors que le verbe existe dans WordNet,
- Quelques membres ne précisent pas de *synset*, et le verbe n'existe pas dans WordNet.

Toutefois, en dépit de cette remarque, nous pouvons considérer qu'une grande majorité des verbes décrits dans VerbNet a une correspondance directe vers WordNet.

¹ Par exemple, dans WordNet version 2.0, le concept « chat » (le félin domestique au sens usuel) est le nom qui a pour identifiant le nombre 2037721. C'est l'une des deux façons d'identifier un *synset* d'une façon unique.

² Une interface entre SUMO et WordNet a été définie par <http://www.teknowledge.com>. Certaines des correspondances qu'elle propose sont discutables, mais elle a le mérite d'exister. On pourra consulter à ce sujet [Niles & Pease, 2003].

³ Présence d'un marqueur « ? ».

Correspondance des verbes entre WordNet et VerbNet

WordNet version 2.0 décrit approximativement 13 500 *synsets* et 24 600 lemmes de verbes. Un membre de VerbNet peut être associé à plusieurs *synsets* de verbes ; un peu plus de 6 000 *synsets* de WordNet sont ainsi référencés dans VerbNet (soit seulement 44%). Toutefois, quand une relation directe n'existe pas entre WordNet et VerbNet, on peut utiliser la relation d'héritage de WordNet pour rechercher si l'hyperonyme d'un verbe a une relation directe dans VerbNet.

Correspondance entre rôles thématiques et sommet de la hiérarchie des noms

Nous avons vu qu'il existe dans WordNet une hiérarchie d'héritage pour les noms. A titre indicatif, les deux premiers niveaux se constituent des concepts abstraits suivants :

- **Abstraction**: attribute, measure/quantity/amount, relation, set, space, time...
- **Human action**: activity, communication, distribution, inactivity, judgment, leaning, legitimation, motivation, proclamation, production, speech act, waste...
- **Entity**: anticipation, causal agent, enclosure, expanse, location, physical object, sky, substance, thing...
- **Event**: group action, natural event, might-have-been, migration, miracle, nonevent, social event...
- **Group, grouping**: association, biological group, people, collection, aggregation, community, ethnic group, kingdom, multitude, population, race, rare-earth element...
- **Phenomenon**: effect/result, levitation, fortune/chance, rebirth, natural phenomenon, process, pulsation...
- **Possession**: assets, circumstances, property/material possession, transferred property, treasure...
- **Psychological feature**: cognition/knowledge, feeling, motivation/need...
- **State**: action/activity, existence, state of mind, condition, conflict, damnation, death, degree, dependency, disorder, employment, end, freedom, antagonism, immaturity, imminence, imperfection, integrity, maturity, omnipotence, perfection, physiological state, relationship, state of affairs, status, temporary state, natural state...

L'un des points essentiels dans la réalisation de notre interface syntaxe / sémantique est l'établissement d'une correspondance entre les rôles thématiques de VerbNet et le haut de la hiérarchie des noms dans WordNet. Pour ce faire, nous définissons pour chaque contrainte de sélection une ou plusieurs règles, encodée(s) en XML, lui associant des concepts de WordNet. Ces règles sont essentiellement de deux types¹. Une contrainte de sélection s'applique :

- a) Si un concept hérite d'un concept ancêtre particulier défini plus haut dans la hiérarchie des noms de WordNet,

ou
- b) Si la définition textuelle d'un concept (ou de l'un de ses ancêtres) est qualifiée par un attribut particulier.

¹ D'autres règles, plus spécifiques, ne sont pas détaillées ici ; elles traitent par exemple le cas des littéraux ou des pronoms personnels réflexifs (*myself, yourself, etc.*).

En tout, nous avons défini une centaine de règles de correspondance. Ces règles sont stockées dans un fichier XML pour être facilement modifiables.

Précisons ces règles à travers un exemple. La classe de verbes “**poke**” (« enfoncer ») précise que le rôle *Instrument* a une contrainte de sélection *Pointu*. Nous allons voir comment définir une telle contrainte avec les deux types de règles.

Héritage d'un concept particulier

Nous exprimons cette contrainte avec des règles du type a) en considérant comme *Pointu* tout héritier des noms « aiguille » (“*needle*”) ou « clou » (“*nail*”) au sens usuel. Il faut toutefois lever des ambiguïtés lexicales ; en effet, “*nail*” a trois sens dans notre lexique, et peut vouloir dire :

- Un ongle¹,
- Un **clou** (c’est le sens qui nous intéresse),
- Une ancienne unité de mesure.

C’est à ce stade que nous utilisons les informations venant de SUMO, pour lever l’ambiguïté. Notre lexique associe un domaine de l’ontologie SUMO à chacun de ces trois sens de “*nail*” :

- BodyPart,
- **EngineeringComponent**,
- UnitOfMeasure.

Nous déclarerons donc notre contrainte de sélection *Pointu* comme tout héritier de “*nail*” au sens “*Engineering Component*” :

```
<Mapping selRestr="pointy" type="noun" value="nail/EngineeringComponent" />
```

De même, nous pouvons ajouter une seconde règle de type a) pour prendre en compte les héritiers de « aiguille » (“*needle*”) au sens « instrument » (“*device*”).

```
<Mapping selRestr="pointy" type="noun" value="needle/Device" />
```

Présence d'un attribut dans un concept ancêtre

La contrainte *Pointu* s’exprime aussi avec des règles du type b). Nous vérifions alors que la définition du concept (ou de l’un de ses ancêtres) contient un adjectif tel que « pointu » (“*pointed*”) ou « aiguisé » (“*sharp*”) avec les règles :

```
<Mapping selRestr="pointy" type="attribute" value="pointed" />
<Mapping selRestr="pointy" type="attribute" value="sharp" />
```

De cette façon, “*claw*” (« griffe » d’animal), qui a pour définition “*a sharp curved horny process on the toe of a bird or some mammals or reptiles*”, sera automatiquement reconnu comme compatible avec la contrainte de sélection *Pointu*.

¹ Certes, un ongle peut être considéré comme *Pointu* (« elle m’a enfoncé ses ongles dans la chair ») en plus d’être *Tranchant*, mais nous négligerons ce point ici.

Utilisation de la forme logique d'eXtended WordNet

Nous utilisons la forme logique fournie par eXtended WordNet pour vérifier les règles du type b). Par exemple, la forme logique du nom “*claw*” est l’expression suivante :

```
claw:NN(x1) → sharp:JJ(x1) curved:JJ(x1) horny:JJ(x1) process:NN(x1)
on:IN(x1, x2) toe:NN(x2) of:IN(x2, x6) bird:NN(x3) or:CC(x6, x3, x7)
some:JJ(x7) mammal:NN(x4) or:CC(x7, x4, x5) reptile:NN(x5)
```

Pour obtenir cette forme, la définition textuelle a été traitée par un analyseur morpho syntaxique qui a identifié les parties du discours de chaque mot de la définition (JJ pour un adjectif, NN pour un nom, etc.). On remarque que dans ce formalisme le mot sur lequel porte la définition est toujours identifié par x1. Pour déterminer si le nom est *Pointu*, on cherche donc tout simplement si la forme logique contient la chaîne de caractères « sharp:JJ(x1) ».

Enrichissement du Link Grammar Parser grâce à VerbNet et WordNet

Enrichissement du lexique grâce à WordNet

Le lexique de référence standard du *Link Grammar Parser* compte approximativement 60 000 mots. Il est intéressant de le compléter à partir de WordNet, qui est plus riche.

Enrichir un lexique à partir d’un autre est une opération complexe, du fait d’importantes différences de structures dans ces lexiques. Par exemple, WordNet n’a pas d’information indiquant si un concept est un nom dénombrable ou un nom de masse ; or, cette information est essentielle pour connaître les traitements grammaticaux possibles sur le concept.

Nous n’avons pas cherché à développer une telle interface. Nous nous sommes contenté d’importer la ressource existante *LinkGrammar-WN*¹. Cette ressource enrichit le lexique du *Link Grammar Parser* de 14 000 nouveaux noms provenant de WordNet. Elle a été créée en se basant sur la démarche de fusion de lexiques présentée dans [Szolovits, 2004].

Enrichissement de la grammaire grâce à VerbNet

Les cadres de description de la syntaxe des verbes permettent d’enrichir les règles lexico syntaxiques utilisées par le *Link Grammar Parser*. Par exemple, VerbNet indique que le verbe “*to like*” accepte une complétive ; cette information n’est pas présente dans la grammaire d’origine de l’analyseur syntaxique. Cette amélioration, et quelques autres, ont été ajoutées manuellement².

Il serait possible d’automatiser une campagne d’enrichissement de ce type, qui permettrait d’améliorer la qualité de l’analyseur syntaxique, et de lui faire reconnaître davantage de constructions licites.

¹ Disponible sur <http://www.eturner.net/linkgrammar-wn>.

² Par ajout du connecteur TH+ dans la description <vc-like> (on remarque ici que le formalisme utilisé par le *Link Grammar Parser* n’est pas toujours d’une lecture aisée) : ({@MV+} & (TO+ or TH+ or Pg+)) or ((O+ or B- or OX+) & {@MV+} & {TOo+}) or ([[@MV+ & O*n+]]).

De la syntaxe à la sémantique par identification de schémas

Introduction

Nous allons présenter ici le mécanisme principal utilisé dans notre démarche, en détaillant :

- Les structures syntaxique et sémantique, et le processus de passage de l'une à l'autre,
- Le mécanisme d'identification de schémas,
- L'application de ce mécanisme à la reconnaissance de constructions typiques,
- Des rappels sur PROLOG, le langage que nous avons choisi d'utiliser pour implémenter ce mécanisme.

Quelques rappels sur PROLOG

PROLOG (PROgrammation LOGique) est un langage très utilisé dans le monde de l'Intelligence Artificielle, facile à utiliser. Il suffit d'énoncer des faits et des règles pour avoir un programme fonctionnel.

Un petit peu d'histoire

En 1970, Alain Colmerauer commence à utiliser la logique des prédicats pour faire des déductions sur des textes. En se basant sur les travaux de Jacques Herbrand sur les prédicats (datant des années 30) et sur ceux, contemporains, de Robert Kowalski sur les méthodes de résolution, il pose les bases du premier modèle théorique de PROLOG. Son objectif est de pouvoir décrire, en français, un univers à l'ordinateur afin que celui-ci puisse répondre à des questions sur cet univers.

Le premier interprète PROLOG est implémenté en FORTRAN en 1972 à l'Université de Marseille. Le premier compilateur PROLOG apparaît en 1977 à l'Université d'Edinburgh.

Principes

PROLOG est un langage de programmation logique basé sur deux grands mécanismes, le chaînage arrière et l'unification :

- **Le chaînage arrière** est le fait de partir du but recherché, de rechercher les règles dont le but est la conclusion, puis, en prenant les conditions de ces règles comme nouveaux sous-buts, recommencer la recherche récursivement. Ensuite, il ne reste plus qu'à unifier les faits trouvés avec le but recherché.
- **L'unification** est le fait d'essayer de rendre deux assertions identiques (un fait et une règle en général) en donnant des valeurs aux variables qu'elles contiennent.

PROLOG est un langage plutôt clair et lisible. En dépit de son âge canonique (à l'échelle informatique) et de la concurrence de langages logiques plus récents et plus puissants¹, PROLOG reste largement utilisé pour l'écriture de systèmes experts et le traitement du langage naturel. Nous allons survoler maintenant les principaux éléments du langage, c'est-à-dire les prédicats, les clauses, les faits, les règles et les buts.

¹ PowerLoom par exemple (voir <http://www.isi.edu/isd/LOOM/PowerLoom>).

Prédicats

Les prédicats expriment une relation entre leurs arguments, ou une propriété de l'un d'eux. Par exemple, le prédicat `est_pere_de(pierre, marie)` indique qu'une relation existe entre Pierre et Marie, en l'occurrence que Pierre est le père de Marie.

Clauses

Les clauses sont un ensemble de faits et de règles. Cet ensemble de clauses forme la base de connaissances que va utiliser le programme pour raisonner. Lors de l'appel d'un prédicat, PROLOG essaye de le démontrer en regardant les règles et les faits associés à ce prédicat. Au fur et à mesure de ses démonstrations, le programme fait remonter les résultats de toutes ses démonstrations et donne ainsi comme réponse toutes les solutions à la question posée au départ.

En termes de syntaxe, chaque clause se termine par un point. Les balises `/*` et `*/` servent à encadrer les commentaires, indiquant à l'interpréteur de ne pas tenir compte du contenu de ces balises.

Faits

Un fait est une relation, toujours vraie, entre des objets. Par exemple :

```
/* ce fait affirme que Pierre est le père de Marie */  
est_pere_de(pierre, marie).
```

Règles

Une règle est une relation conditionnelle, qui prend la forme suivante :

```
conclusion SI condition_1a ET condition_2a (...) ET condition_Ia .  
  
/* OU (implicite) */  
  
conclusion SI condition_1b ET condition_2b (...) ET condition_Ib .
```

La conclusion n'est prouvée que si les conditions sont vraies. Dans la plupart des versions de PROLOG :

- La condition SI s'écrit :-
- Le ET se traduit par une virgule.

Une règle a donc la forme suivante :

```
conclusion :-  
    condition_1a,  
    condition_2a,  
    (...),  
    condition_Ia.  
/* OU */  
conclusion :-  
    condition_1b,  
    condition_2b,  
    (...),  
    condition_Ib.
```

Voyons un exemple concret. Si le prédicat `est_pere_de` est défini, on peut créer la règle :

```
est_grandpere_de(X, Y) :- est_pere_de(X, Z), est_pere_de(Z, Y).  
/* X est le grand-père de Y si X est le père de Z et Z est le père de Y */
```

On remarquera l'utilisation de symboles commençant par une majuscule (X, Y et Z). En PROLOG, tout symbole commençant par une majuscule est considéré comme une variable.

Si on ajoute les prédicats `est_mere_de`, `a_pour_frere` et `a_pour_soeur`, on peut de même créer la règle `est_oncle_de` en exprimant que le neveu (ou la nièce) de quelqu'un est l'enfant de son frère ou de sa sœur :

```
est_oncle_de(X, Y) :- a_pour_frere(X, F), est_pere_de(F, Y).  
/* OU */  
est_oncle_de(X, Y) :- a_pour_soeur(X, S), est_mere_de(S, Y).
```

Un point remarquable est qu'une règle peut se définir d'une définition récursive. Par exemple :

```
est_ancetre_de(X, Y) :- est_pere_de(X, Z), est_ancetre_de(Z, Y).
```

Buts

Un but permet d'interroger un programme. PROLOG résout ce but, et affiche toutes les solutions possibles. Voici des exemples de buts :

```
est_pere_de(jean, marie)  
/* demande si Jean est le père de Marie */  
  
est_pere_de(pierre, X)  
/* demande si Pierre a des enfants et si oui affiche leur nom */
```

Structures syntaxique et sémantique

Hypothèses de travail

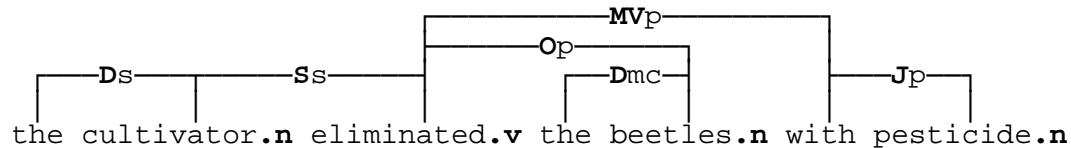
Notre donnée de départ est une phrase en anglais. Nous faisons une triple hypothèse :

- La phrase est grammaticalement correcte,
- Elle est analysée avec succès¹ par le *Link Grammar Parser* (aucun mot n'est ignoré),
- Tous les mots constituant la phrase sont connus du lexique.

Structure syntaxique de départ

Le *Link Grammar Parser* produit, à partir de la phrase fournie en entrée, un graphe orienté acyclique de dépendances. Par exemple, l'une des interprétations syntaxiques de la phrase "*the cultivator eliminated the beetles with pesticide*" est :

¹ Par exemple, une phrase elliptique comme "*John eats the apple and Mary the pear*", pourtant constituée de peu de mots, n'est pas correctement reconnue.



Dans ce graphe, on remarque que :

- Les **nœuds** sont les mots de la phrase ; certains d’entre eux ont un suffixe qui indique la partie du discours (nom, verbe, adjectif, adverbe, préposition, etc.). Par exemple, “*eliminated.v*” est reconnu en tant que forme de verbe (suffixe *.v*), et “*beetles.n*” en tant que forme de nom (suffixe *.n*),
- Des **arcs étiquetés** relient les nœuds du graphe ; chaque étiquette précise un rôle grammatical (sujet-verbe, déterminant-nom, etc.¹). Par exemple, entre “*cultivator.n*” et “*eliminated.v*”, le libellé du lien est « Ss » où :
 - La première lettre (S_ majuscule) désigne une fonction sujet (*Subject*),
 - La seconde lettre (_s minuscule) précise qu’il est singulier (*singular*).

Le graphe issu de l’analyse effectuée par le *Link Grammar Parser* est notre structure syntaxique de départ.

Structure sémantique

Nous cherchons à transformer le graphe syntaxique en une structure sémantique. Nous retenons comme formalisme, pour cette dernière, un ensemble de prédicats. Nous détaillons ici les différentes étapes de sa construction².

Structure intermédiaire avec identification des parties du discours

Une représentation équivalente du graphe syntaxique précédent est l’ensemble de clauses (des faits au sens PROLOG) suivant, avec des prédicats *word* (mot) et *link* (lien). Chaque prédicat mot et lien débute avec un numéro identifiant unique au sein de la phrase. Dans les prédicats de type *word*, la partie du discours de chaque mot est explicitée :

```

word(1, "the", article).
word(2, "cultivator", noun).
word(3, "eliminated", verb).
word(4, "the", article).
word(5, "beetles", noun).
word(6, "with", preposition).
word(7, "pesticide", noun).

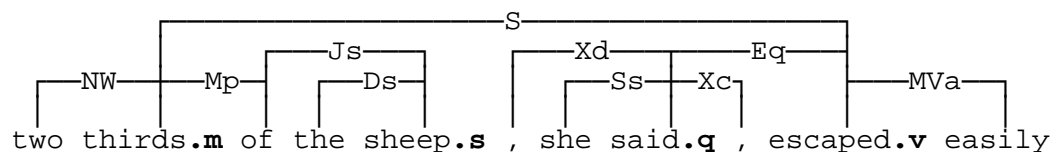
link(2, d, 1, 2). /* the - cultivator */
link(3, s, 2, 3). /* cultivator - eliminated */
link(4, mv, 3, 6). /* eliminated - with */
link(5, o, 3, 5). /* eliminated - beetles */
link(6, d, 4, 5). /* the - beetles */
link(7, j, 6, 7). /* with - pesticide */
  
```

¹ On dénombre, dans le *Link Grammar Parser*, 107 types de lien différents. Leur documentation peut être consultée sur <http://bobo.link.cs.cmu.edu/link/dict>.

² Nous avons conscience de l’intérêt qu’aurait une structure intermédiaire supplémentaire pour représenter la syntaxe profonde. Toutefois, elle n’est pas indispensable dans le cadre de nos travaux.

Passer du graphe syntaxique à cette représentation semble trivial. Toutefois, cette transformation s'avère étonnamment complexe en termes d'ingénierie. En effet, dans le graphe syntaxique, certains mots ("the", "with") n'ont *aucun* suffixe. Pire, on trouve *généralement* un suffixe pour indiquer une partie du discours (.v pour les verbes, .n pour les noms, .a pour les adjectifs, .p pour les prépositions...) mais cette convention n'a pas de caractère impératif ou régulier.

Illustrons cette difficulté sur un autre exemple. La phrase "two thirds of the sheep, she said, escaped easily" (« les deux tiers des moutons s'échappèrent facilement, dit-elle ») donne le graphe suivant. On remarque que le nom "thirds" est suffixé en .m, "sheep" en .s, le verbe "said" en .q (car ils correspondent à des constructions particulières) et que les autres mots, sauf "escaped.v", n'ont pas de suffixe.



En fait, le *Link Grammar Parser* impose seulement que si un mot appartient à plusieurs parties du discours (ou apparaît dans des constructions syntaxiques différentes), il doit apparaître autant de fois dans le lexique, avec des suffixes différents permettant de distinguer les constructions. Par exemple, dans l'exemple précédent, on trouve bien dans le lexique un mot "thirds.n" (avec une extension en n) compatible avec les règles usuelles d'appariement des noms, et "thirds.m" qui ne peut se trouver qu'à droite d'un lien étiqueté en « NW » (comme *Number-Word*) qui représente des fractions.

Pour identifier la partie du discours de tous les mots d'une phrase, nous avons dû définir plusieurs dizaines de règles, tenant compte des « codages en dur » du lexique fourni par le *Link Grammar Parser* et de constructions autour de liens d'un type donné.

Par exemple, on considère comme un adverbe tout mot à droite d'un lien étiqueté en « MVa », comme "easily" dans l'exemple ci-dessus. Autre exemple : on peut considérer comme verbe tout mot à droite d'un lien de type « S », « SF », « SX » ou à gauche d'un lien de type « SI », « SFI », « SXI » (ces trois derniers liens marquant un sujet inversé).

Structure intermédiaire avec informations lexicales

A ce stade, la partie du discours de chaque mot est connue. On peut alors invoquer l'analyseur lexical pour trouver le mot de base de chaque forme fléchie¹.

```

word(1, "the", article).
word(2, "cultivator", noun("cultivator", baseform)).
word(3, "eliminated", verb("eliminate", pasttense)).
word(4, "the", article).
word(5, "beetles", noun("beetle", plural)).
word(6, "with", preposition).
word(7, "pesticide", noun("pesticide", baseform)).

```

(Idem que précédemment pour les liens.)

¹ Pour un nom, "baseform" indique simplement le singulier.

A ce stade, notre analyseur sait prendre en compte des ambiguïtés. Par exemple, la phrase “*I found the city*” est traitée avec ses deux variantes lexico-syntaxiques possibles :

« Je fonde la cité »	« J’ai trouvé la cité »
<pre>word(1, "I", pronoundeterminer(nil)). word(2, "found", verb("found", baseform)). word(3, "the", article(nil)). word(4, "city", noun("city", baseform)). link(2, s, 1, 2). /* I - found */ link(3, o, 2, 4). /* found - city */ link(4, d, 3, 4). /* the - city */</pre>	<pre>word(1, "I", pronoundeterminer(nil)). word(2, "found", verb("find", pasttense)). word(3, "the", article(nil)). word(4, "city", noun("city", baseform)). link(2, s, 1, 2). /* I - found */ link(3, o, 2, 4). /* found - city */ link(4, d, 3, 4). /* the - city */</pre>

De même, dans “*I see the sheep*”, on identifie une construction au singulier et une au pluriel :

« Je vois un mouton »	« Je vois des moutons »
<pre>word(1, "I", pronoundeterminer(nil)). word(2, "see", verb("see", baseform)). word(3, "the", article(nil)). word(4, "sheep", noun("sheep", baseform)). link(2, s, 1, 2). /* I - see */ link(3, o, 2, 4). /* see - sheep */ link(4, d, 3, 4). /* the - sheep */</pre>	<pre>word(1, "I", pronoundeterminer(nil)). word(2, "see", verb("see", baseform)). word(3, "the", article(nil)). word(4, "sheep", noun("sheep", plural)). link(2, s, 1, 2). /* I - see */ link(3, o, 2, 4). /* see - sheep */ link(4, d, 3, 4). /* the - sheep */</pre>

Structure intermédiaire avec identification des constructions de verbes

A présent, nous pouvons utiliser les informations fournies par VerbNet pour identifier les constructions typiques de verbe (le procédé exact mis en œuvre est détaillé page 37). Nous obtenons à présent pour notre phrase d’exemple :

```
word(1, "the", article).
word(2, "cultivator", noun("cultivator", baseform)).
word(3, "eliminated", verb("eliminate", pasttense)).
word(4, "the", article).
word(5, "beetles", noun("beetle", plural)).
word(6, "with", preposition).
word(7, "pesticide", noun("pesticide", baseform)).

link(2, d, 1, 2). /* the - cultivator */
link(3, s, 2, 3). /* cultivator - eliminated */
link(4, mv, 3, 6). /* eliminated - with */
link(5, o, 3, 5). /* eliminated - beetles */
link(6, d, 4, 5). /* the - beetles */
link(7, j, 6, 7). /* with - pesticide */

frame_murder(3 /* Verb = eliminated */,
  2 /* Agent = cultivator */,
  5 /* Patient = beetles */,
  7 /* Instrument = pesticide */).

has_constraint(2 /* cultivator */, animate).
has_constraint(5 /* beetle */, animate).
```

Structure sémantique avec désambiguïstation lexicale

Il ne nous reste plus qu’à invoquer à nouveau l’analyseur lexical, pour trouver la liste des sens des mots compatibles avec les contraintes de sélections fixées à l’étape précédente.

Notre forme sémantique se trouve enrichie d’un prédicat qui indique les sens possibles pour un mot :


```

word(1, "I", pronoundeterminer(nil)).
word(2, "have", verb("have", baseform)).
word(3, "closed", verb("close", pastparticiple)).
word(4, "the", article(nil)).
word(5, "door", noun("door", baseform)).

link(2, s, 1, 2). /* he - has */
link(3, pp, 2, 3). /* has - closed */
link(4, o, 3, 5). /* closed - door */
link(5, d, 4, 5). /* the - door */

```

Nous avons indiqué en gras, dans les prédicats et dans le graphe syntaxique, les éléments caractéristiques d'une construction au participe passé. Ce sont :

- Un lien de type PP qui relie :
 - Le verbe auxiliaire "have"
 - A un verbe au participe passé.

L'écriture d'un programme PROLOG qui extrait des constructions de ce type est facile. Il suffit d'indiquer qu'on cherche à reconnaître ces éléments, et qu'ils sont reliés entre eux. Pour améliorer la lisibilité de ce programme, nous avons surligné dans la même couleur les variables de même noms, qui correspondent aux éléments qui doivent être unifiés :

```

recognize_past_participle(AuxiliaryTense, VerbBaseForm, NLink) :-
word(NAuxiliary, _, verb("have", AuxiliaryTense)),
link(NLink, pp, NAuxiliary, NVerb),
word(NVerb, _, verb(VerbBaseForm, pastparticiple)).

```

En exécutant le programme, on obtient le temps de l'auxiliaire, la forme de base du verbe et le numéro du lien les reliant :

```

AuxiliaryTense = baseform
VerbBaseForm = "close"
NLink = 3

```

Cette réponse correspond à la reconnaissance des éléments surlignés dans la phrase suivante :

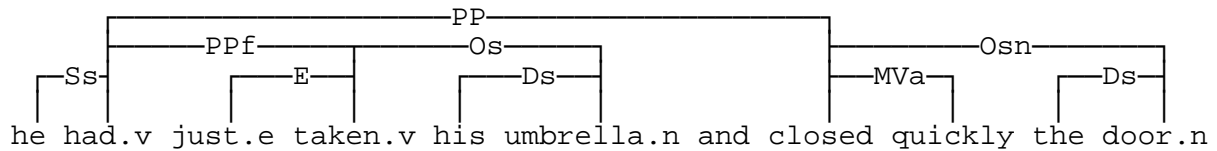
```

word(1, "I", pronoundeterminer(nil)).
word(2, "have", verb("have", baseform)).
word(3, "closed", verb("close", pastparticiple)).
word(4, "the", article(nil)).
word(5, "door", noun("door", baseform)).

link(2, s, 1, 2). /* he - has */
link(3, pp, 2, 3). /* has - closed */
link(4, o, 3, 5). /* closed - door */
link(5, d, 4, 5). /* the - door */
link(5, d, 4, 5). /* the - door */

```

Cette approche a l'intérêt de tolérer les modifications de surface. Par exemple, si on complique la phrase (ajout d'adverbes, apparition d'un second groupe verbal), cela ne perturbe pas la reconnaissance :



L'exécution du même programme PROLOG trouve maintenant deux réponses :

```
AuxiliaryTense = pasttense
VerbBaseForm = "take"
NLink = 3

AuxiliaryTense = pasttense
VerbBaseForm = "close"
NLink = 7
```

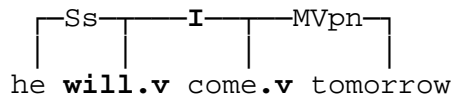
Autre exemples

Nous avons vu, à travers un exemple, le principe général d'identification de schémas par un programme PROLOG. Nous allons maintenant présenter quelques autres exemples de constructions grammaticales usuelles, de façon à illustrer davantage notre propos. Pour chaque construction, nous mettons en gras dans le graphe syntaxique les éléments caractéristiques que nous cherchons à identifier.

Futur

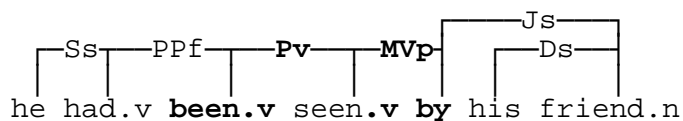
“*He will come tomorrow*” (« Il viendra demain »).

On cherche l'auxiliaire “*will*” reliant un verbe à l'infinitif via un lien étiqueté en « I ».



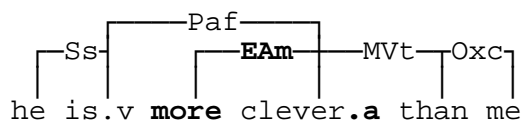
Forme passive

“*He had been seen by his friend*” (« Il a été vu par son ami »).



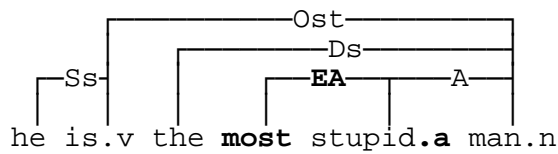
Comparatif

“*He is more clever than me*” (« Il est plus malin que moi »).



Superlatif

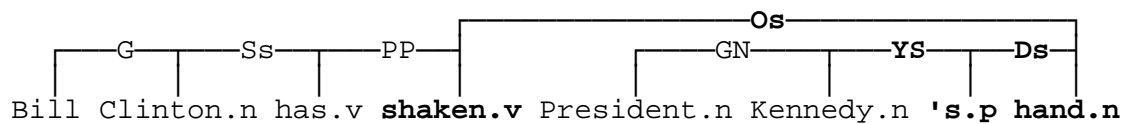
“*He is the most stupid man*” (« C’est l’homme le plus stupide »).



Collocation

De même, cette approche permet la détection d’une collocation¹ complexe, comme par exemple le verbe “*to shake one’s hand*” (« serrer la main de quelqu’un »).

Bill Clinton has shaken President Kennedy's hand (« Bill Clinton a serré la main du president Kennedy »).



¹ Nous entendons ici par collocation une cooccurrence fréquente, au sens anglo-saxon, comme “*whiskey on the rocks*” (« whiskey sec ») ou “*dark coffee*” (« café noir »).

Mise en œuvre

Introduction

Notre approche consiste à travailler sur la langue anglaise, en s'appuyant sur des ressources lexicales, syntaxiques et sémantiques de large couverture, issues de projets de recherche, et en faisant coopérer ces ressources. Partant d'une phrase en langage naturel, nous cherchons à :

- Désambiguïser la phrase, sur le plan syntaxique et lexical,
- Attribuer un « indice de vraisemblance » à chaque interprétation de la phrase.

Nous avons vu dans le chapitre précédent le mécanisme d'identification de schémas. Nous allons maintenant présenter, dans la première partie de ce chapitre, la façon dont nous l'appliquons pour atteindre notre objectif, en détaillant :

- L'application de ce mécanisme à la reconnaissance de constructions verbales,
- Une démarche d'automatisation visant à générer, à partir de VerbNet, des programmes PROLOG de reconnaissance de constructions typiques de verbes,
- Des pistes de désambiguïstation des noms dans le groupe nominal.

Les traitements effectués pendant ces différentes étapes permettent d'identifier plusieurs schémas, avec des contraintes de diverses natures qui portent sur le sens :

- D'un verbe polysémique (contrainte imposée par une classe de verbe),
- D'un nom considéré individuellement¹,
- De plusieurs noms pris simultanément².

Ces contraintes sont, de plus, éventuellement incompatibles entre elles. Nous verrons, dans la seconde partie de ce chapitre :

- Comment prendre en compte simultanément toutes les contraintes possibles ?
- Comment calculer un indice de vraisemblance pour chaque interprétation de phrase ?

Utilisation de VerbNet pour identifier des constructions typiques de verbes

Objectif

VerbNet décrit la syntaxe des classes de verbes³. L'une des idées majeures de notre mémoire est d'utiliser cette information pour identifier des constructions verbales typiques dans une phrase. Pour ce faire, nous allons traduire la description de la syntaxe d'une classe de verbes en un programme PROLOG, capable d'extraire du graphe syntaxique créé par le *Link Grammar Parser* chaque verbe prédicat avec ses arguments (ses rôles thématiques).

¹ Par exemple, une contrainte de sélection peut imposer que l'*Agent* d'un verbe soit *Animé*.

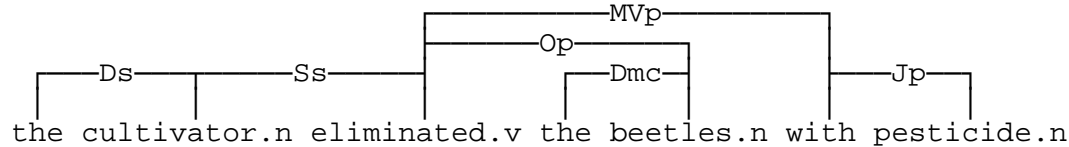
² Par exemple, les noms « veste », « poche » et « kangourou » ont chacun plusieurs sens ; mais dans des combinaisons telles que « la poche de la veste » ou « la poche du kangourou », l'ambiguïté lexicale est levée.

³ Ainsi que leur sémantique, mais à ce stade, nous n'utiliserons que la description de la syntaxe.

Rappels sur la phrase d'exemple

Comme nous l'avons déjà vu, notre phrase d'exemple donne deux constructions syntaxiques, regroupant un total de quatre constructions verbales typiques.

Première construction syntaxique



La structure équivalente, après identification des parties du discours et des formes de base lexicales, est :

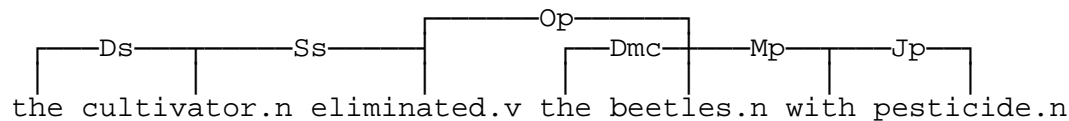
```

word(1, "the", article(nil)).
word(2, "cultivator", noun("cultivator", baseform)).
word(3, "eliminated", verb("eliminate", pasttense)).
word(4, "the", article(nil)).
word(5, "beetles", noun("beetle", plural)).
word(6, "with", preposition(nil)).
word(7, "pesticide", noun("pesticide", baseform)).
link(2, d, 1, 2). /* the - cultivator */
link(3, s, 2, 3). /* cultivator - eliminated */
link(4, mv, 3, 6). /* eliminated - with */
link(5, o, 3, 5). /* eliminated - beetles */
link(6, d, 4, 5). /* the - beetles */
link(7, j, 6, 7). /* with - pesticide */
  
```

Dans ce cas, notre programme propose deux constructions verbales typiques :

- Classe de verbes “**murder**” – construction 2
“the cultivator_(Agent) eliminated_(Verb) the beetles_(Patient) with pesticide_(Instrument)”
 Contraintes de sélection : cultivator_(animate && concrete) beetles_(animate)
- Classe de verbes “**remove**” – construction 1
“the cultivator_(Agent) eliminated_(Verb) the beetles_(Theme) with pesticide”
 Contraintes de sélection : cultivator_(int_control) with_(lsrc)

Seconde construction syntaxique



La structure équivalente, après identification des parties du discours et des formes de base lexicales, est :

```

word(1, "the", article(nil)).
word(2, "cultivator", noun("cultivator", baseform)).
word(3, "eliminated", verb("eliminate", pasttense)).
word(4, "the", article(nil)).
word(5, "beetles", noun("beetle", plural)).
word(6, "with", preposition(nil)).
word(7, "pesticide", noun("pesticide", baseform)).
link(2, d, 1, 2). /* the - cultivator */
link(3, s, 2, 3). /* cultivator - eliminated */
  
```

```
link(4, o, 3, 5). /* eliminated - beetles */
link(5, d, 4, 5). /* the - beetles */
link(6, m, 5, 6). /* beetles - with */
link(7, j, 6, 7). /* with - pesticide */
```

Dans ce second cas, notre programme propose deux autres constructions verbales typiques :

- Classe de verbes “**murder**” – construction 1
“the cultivator_(Agent) eliminated_(Verb) the beetles_(Patient) with pesticide”
 Contraintes de sélection : cultivator_(animate && concrete) beetles_(animate)
- Classe de verbes “**remove**” – construction 1
“the cultivator_(Agent) eliminated_(Verb) the beetles_(Theme) with pesticide”
 Contraintes de sélection : cultivator_(int_control)

Traduction en PROLOG d’une description de classe de verbes

Nous allons rentrer ici dans le détail du processus de traduction d’une classe de verbes vers le programme PROLOG équivalent. Ce processus, assez technique, est l’un des points essentiels de notre mémoire. Nous allons commencer par présenter d’une façon macroscopique les différentes sections d’un fichier de description provenant de VerbNet. Puis nous verrons la correspondance des concepts vers du code PROLOG, et les difficultés à prendre en compte pour effectuer cette traduction sans négliger aucune information. La classe de verbes “**murder**” sera notre fil conducteur.

Structure d’un fichier XML de VerbNet

Chaque fichier de VerbNet est représenté en XML, et découpé en sections balisées (indiquées ci-après entre <CHEVRONS>) selon la structure arborescente suivante :

- <MEMBERS> décrit les verbes membres qui appartiennent à la classe, en précisant l’identifiant de leur sens dans le lexique WordNet,
- <THEMROLES> indique les rôles thématiques de la classe :
 - <SELRESTRS> précise leurs éventuelles contraintes de sélections,
- <FRAMES> indique chacune des constructions typiques, en donnant à chaque fois :
 - <SYNTAX> sa syntaxe,
 - <SEMANTICS> sa sémantique,
 - <EXAMPLES> un ou plusieurs exemples,
- <SUBCLASSES> regroupe éventuellement en sous-classes :
 - <VNSUBCLASS> les cas particulier d’une classe de verbes.

La classe de verbe “murder”

Le fichier *murder.xml* contient 170 lignes¹. On y trouve trois constructions typiques, avec une phrase d’exemple pour chaque construction :

- *Agent* élimine *Patient* (« Brutus tua Jules César »),
- *Agent* élimine *Patient* avec *Instrument* (« Brutus tua César avec un poignard »),
- *Instrument* élimine *Patient* (« le pesticide tua les insectes »).

¹ Le fichier *murder.xml* est reproduit en annexe (page 63) sous une forme légèrement simplifiée.

Correspondance des concepts entre VerbNet et du code PROLOG

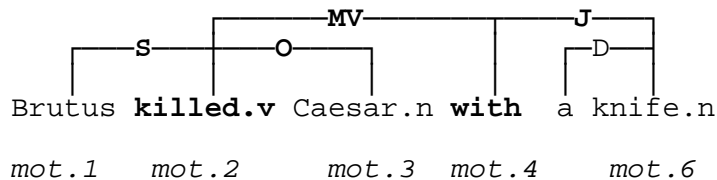
VerbNet associe un ou plusieurs exemples à chaque *frame*. La description de la syntaxe de chaque exemple donne le cadre de sous-catégorisation permettant d'identifier une construction typique. Notre idée maîtresse consiste à utiliser cet exemple, en établissant une correspondance entre :

- La déclaration de syntaxe de la *frame* fournie par VerbNet,
- La syntaxe de la phrase d'exemple de la *frame* analysée par le *Link Grammar Parser*.

Voyons cela sur un cas concret. La déclaration de la deuxième *frame* de la classe de verbe “murder” se compose des éléments suivants¹ :

- `<SYNTAX>`
`<NP value="Agent" />`
`<VERB />`
`<NP value="Patient" />`
`<PREP value="with" />`
`<NP value="Instrument" />`
`</SYNTAX>`
- `<EXAMPLES>`
`<EXAMPLE>"Brutus killed Caesar with a knife"2</EXAMPLE>`
`</EXAMPLES>`

Notre approche consiste à utiliser le *Link Grammar Parser* pour analyser la (ou les) phrase(s) d'exemple. En l'occurrence, nous obtenons :



Nous commençons par identifier dans l'exemple le verbe (l'unique nom suffixé en .v). Nous conservons ensuite les liens, à gauche ou à droite, du verbe et des autres éléments littéraux explicitement cités dans la déclaration de syntaxe (ici la préposition “with”), ce qui donne³ :

```
link(2, s, 1, 2). /* Brutus - killed */
link(4, o, 2, 3). /* killed - Caesar */
link(3, mv, 2, 4). /* killed - with */
link(5, j, 4, 6). /* with - knife */
(Remarque : ici, le lien d entre "a" et "knife" ne nous sert donc pas.)
```

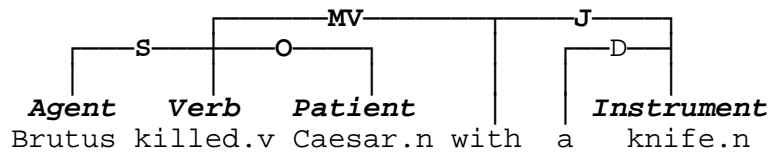
¹ En toute rigueur, les balises `<NP />` et `<PREP />` incluent d'autres balises `<SYNRESTRS />` et `<SELRESTRS />`. Ici, ces dernières sont vides, ce qui indique simplement que ces cinq fragments syntaxiques n'ont pas de contraintes de sélection particulières, en plus de celles déjà déclarées d'une façon globale pour les rôles thématiques. C'est pourquoi nous ne les avons pas reproduites dans cet exemple.

² « Brutus tua César avec un poignard ».

³ Rappelons que pour représenter les liens syntaxiques, nous utilisons un prédicat `link()` à quatre arguments : l'identifiant du lien, l'étiquette du lien, l'indice du mot à gauche du lien, l'indice du mot à droite du lien.

Synchronisation des deux structures syntaxiques

Il ne nous reste plus qu'à « synchroniser » ces différentes structures :



<NP value="Agent" />	/* Brutus */	
	link(2, s, 1, 2).	link(L2, s, Agent, Verb).
<VERB />	/* killed */	
	link(4, o, 2, 3).	link(L4, o, Verb, Patient).
<NP value="Patient" />	/* Caesar */	
	link(3, mv, 2, 4).	link(L3, mv, Verb, With).
<PREP value="with" />	/* with */	
	link(5, j, 4, 6).	link(L5, j, With, Instrument).
<NP value="Instrument" />	/* knife */	

Codage du prédicat d'identification de la *frame 2* de “murder”

En disposant de cette table de correspondance, l'écriture du début du prédicat PROLOG qui reconnaît la *frame* est tout simple. On applique la méthode vue page 33 pour identifier un schéma avec les quatre liens correctement reliés entre eux et la préposition “with”.

```

frame_murder_2(L2:L4:L3:L5:nil, NAgent, NVerb, NPatient, NInstrument) :-
link(L2, s, NAgent, NVerb),
link(L4, o, NVerb, NPatient),
link(L3, mv, NVerb, NWith),
word(NWith, "with", preposition(nil)),
link(L5, j, NWith, NInstrument),
...

```

Ce prédicat a comme signature :

- La liste des liens consommés (L2:L4:L3:L5:nil),
- L'indice du verbe (NVerb),
- L'indice des mots correspondant aux trois rôles thématiques (NAgent, NPatient, NInstrument).

Il nous reste toutefois à vérifier des contraintes :

- Dans cette construction, le *Patient* apparaît forcément avant l'*Instrument*¹.
- Les rôles thématiques *Agent*, *Instrument* et *Patient* ont des contraintes de sélection,
- La forme de base du verbe doit être effectivement membre de la classe de verbe.

¹ Il n'est pas absurde de supposer que “Brutus killed with a knife Caesar” corresponde aussi à une construction valide (avec *Instrument* avant *Patient*), mais il faudrait dans ce cas déclarer explicitement dans le fichier *murder.xml* une *frame* correspondant à l'ordre *Agent, Verbe, Instrument, Patient*.

Fin du codage du prédicat d'identification de la *frame 2* de “murder”

Pour prendre en compte ces différentes contraintes, notre prédicat PROLOG est finalisé de la façon suivante :

```
frame_murder_2(WN, L2:L4:L3:L5:nil, NAgent, NVerb, NPatient, NInstrument):-
link(L2, s, NAgent, NVerb),
link(L4, o, NVerb, NPatient),
link(L3, mv, NVerb, NWith),
word(NWith, "with", preposition(nil)),
link(L5, j, NWith, NInstrument),

/* Patient doit apparaître avant Instrument */
NPatient < NInstrument,

/* Agent, Instrument, Patient ont des contraintes de sélection */
isAgent_murder_42_1(NAgent),
isPatient_murder_42_1(NPatient),
isInstrument_murder_42_1(NInstrument),

/* La forme de base du verbe doit être membre de la classe de verbe */
word(NVerb, _VerbDerivedForm, verb(Verb, VerbForm)),
isMember_murder_42_1(Verb, WN).
```

Remarquons au passage qu'on ajoute en tête de la signature du prédicat un premier argument `WN` qui correspond à la liste des sens possibles du verbe, en accord avec la codification de WordNet. Notons aussi que les prédicats utilisés (`isAgent`, `isPatient`, `isInstrument`, `isMember`) sont suffixés par `murder_42_1` ; nous allons maintenant expliquer pourquoi.

Prise en compte de l'héritage entre classes

La balise `<SUBCLASSES>` déclare les éventuelles sous-classes qui spécialisent une classe de verbe donnée. Une sous-classe permet :

- De raffiner les contraintes de sélection portant sur les rôles thématiques,
- De déclarer de nouveaux rôles thématiques,
- D'associer des verbes à la sous-classe,
- De créer de nouvelles *frames*.

L'identifiant d'une sous-classe étend l'identifiant de sa classe mère. Par exemple, la classe principale de “murder” a pour identifiant `ID="murder-42.1"` ; sa sous-classe unique a pour identifiant `ID="murder-42.1-1"`. Notre code doit donc être architecturé pour tenir compte de cette notion, ainsi que de la relation d'héritage¹ entre une sous-classe et sa classe mère. Pour ce faire, nous suffixons par cet identifiant le nom des prédicats qui testent :

- Les contraintes de sélection sur les rôles thématiques,
- L'appartenance d'un verbe en tant que membre de la classe.

C'est pourquoi notre prédicat `frame_murder_2()` appelle `isMember_murder_42_1()`, `isAgent_murder_42_1()`, `isPatient_murder_42_1()` et `isInstrument_murder_42_1()`.

¹ Au sens classique du paradigme orienté objet, c'est-à-dire que la sous-classe hérite de tous les comportements de sa classe mère.

Codage de l'appartenance d'un membre à la classe de verbes

Le fichier *murder.xml* déclare une dizaine de verbes membres, dont “kill” (« tuer ») qui est introduit au niveau de la sous-classe :

```
<VNCLASS ID="murder-42.1">
  <MEMBERS>
    <MEMBER name="assassinate" wn="%2:41:00"/>
    <MEMBER name="butcher" wn="%2:35:00"/>
    <MEMBER name="eliminate" wn="%2:30:00"/>
    ...
  </MEMBERS>
  <SUBCLASSES>
    <VNSUBCLASS ID="murder-42.1-1">
      <MEMBERS>
        <MEMBER name="kill" wn="%2:35:00 %2:35:01 %2:42:00"/>
      </MEMBERS>
    ...
  </SUBCLASSES>
</VNCLASS>
```

Nous reprenons cette information dans notre code PROLOG. Pour implémenter le mécanisme d'héritage entre classes, nous ajoutons explicitement (ligne en gras) que tous les verbes membres de la sous-classe "murder-42.1-1" doivent être également pris en compte à ce niveau. De cette façon, le verbe “kill” pourra être utilisé dans les constructions de la classe "murder-42.1" même s'il n'est défini que dans la sous-classe "murder-42.1-1".

```
/* VNCLASS ID="murder_42_1" */
isMember_murder_42_1("assassinate", 2408713:nil).
isMember_murder_42_1("butcher", 1283599:nil).
isMember_murder_42_1("eliminate", 457540:nil).
...
isMember_murder_42_1(Verb, WN) :- isMember_murder_42_1_1(Verb, WN).

/* VNCLASS ID="murder_42_1_1" */
isMember_murder_42_1_1("kill", 1284688:1286230:2667731:nil).
```

Codage des contraintes de sélection sur les rôles thématiques

Le fichier *murder.xml* déclare plusieurs contraintes de sélection :

```
<VNCLASS ID="murder-42.1">
  <THEMROLE type="Agent">
    <SELRESTRS><SELRESTR Value="+" type="animate"/></SELRESTRS>
  </THEMROLE>
  <THEMROLE type="Patient">
    <SELRESTRS><SELRESTR Value="+" type="animate"/></SELRESTRS>
  </THEMROLE>
  <THEMROLE type="Instrument">
    <SELRESTRS/> <!-- à ce stade, pas de contrainte de sélection -->
  </THEMROLE>
  ...
  <SUBCLASSES>
    <VNSUBCLASS ID="murder-42.1-1">
      <THEMROLE type="Instrument"> <!-- raffinement local -->
        <SELRESTRS><SELRESTR Value="+" type="concrete"/></SELRESTRS>
      </THEMROLE>
    ...
  </SUBCLASSES>
</VNCLASS>
```

On voit que *Patient* et *Agent* ont toujours une contrainte de sélection *Animé*, et qu'*Instrument* doit être *Concret* mais cette dernière contrainte de sélection est déclarée uniquement à partir de la sous-classe "murder-42.1-1".

De même que précédemment, nous reprenons cette information dans notre code PROLOG en prenant en compte l'héritage entre classes.

```
/* VNCLASS ID="murder_42_1" */

isAgent_murder_42_1(NAgent) :-
testSelRestr(20 /* animate */, NAgent), !.

isPatient_murder_42_1(NPatient) :-
testSelRestr(20 /* animate */, NPatient), !.

isInstrument_murder_42_1(NInstrument) :- .

/* VNCLASS ID="murder_42_1_1" */

/* comportement identique à celui de la classe-mère */
isAgent_murder_42_1_1(NAgent) :- isAgent_murder_42_1(NAgent), !.

/* comportement identique à celui de la classe-mère */
isPatient_murder_42_1_1(NPatient) :- isPatient_murder_42_1(NPatient), !.

/* surcharge locale par rapport à la classe-mère */
isInstrument_murder_42_1_1(NInstrument) :-
isInstrument_murder_42_1(NInstrument),
testSelRestr(1 /* concrete */, NInstrument), !.
```

Le prédicat `testSelRestr()` sert à tester individuellement une contrainte de sélection (premier argument, codé sous forme d'un entier) sur un rôle thématique (second argument, identifié par la position du mot directeur associé). L'implémentation de ce prédicat utilise la technique présentée page 24.

Point d'entrée du programme

A ce stade, nous avons vu comment traduire en PROLOG la détection de chaque *frame* avec la prise en compte du mécanisme d'héritage entre classes de verbes. Il nous reste à rendre homogène l'appel aux différentes *frames* d'une classe de verbes. En effet, il ne serait pas pratique de devoir tester systématiquement la présence de chaque *frame* en tenant compte de sa signature particulière :

```
/* Agent tue Patient (Brutus tua Jules César) */
frame_murder_1(WN, Links, Agent, Verb, Patient)

/* Agent tue Patient avec Instrument (Brutus tua César avec un poignard) */
frame_murder_2(WN, Links, Agent, Verb, Patient, Instrument)

/* Instrument tue Patient (le pesticide tua les insectes) */
frame_murder_3(WN, Links, Verb, Patient, Instrument)
```

Nous allons donc encapsuler l'appel aux différentes *frames* avec un prédicat de haut niveau à sept arguments, `frame_murder()`, qui devient le point d'entrée de notre programme. Nous pourrions le coder comme suit :

```

frame_murder(1, WN, Links, Verb, Agent, Patient, Instrument):-
frame_murder_1(WN, Links, Agent, Verb, Patient).

frame_murder(2, WN, Links, Verb, Agent, Patient, Instrument):-
frame_murder_2(WN, Links, Agent, Verb, Patient, Instrument).

frame_murder(3, WN, Links, Verb, Agent, Patient, Instrument):-
frame_murder_3(WN, Links, Verb, Patient, Instrument).

```

Toutefois, cette implémentation souffre d'un petit défaut. Si on l'applique à la phrase "*Brutus killed Caesar with a knife*", on obtient... trois réponses ! En effet, elle reconnaît la construction la plus longue (*Agent, Verbe, Patient, Instrument*) grâce à `frame_murder_2`, mais aussi ses deux sous-ensembles possibles :

- (*Agent, Verbe, Patient*) grâce à `frame_murder_1`,
- (*Patient, Verbe, Instrument*) grâce à `frame_murder_3`.

Ce comportement n'est aucunement une erreur. Simplement, nous allons le modifier pour tenir compte d'une hypothèse linguistique que nous formulons ainsi : « *quand on cherche à identifier une construction verbale, la construction la plus couvrante possible peut être privilégiée* ».

Un ordre partiel existe entre les *frames*. On peut considérer qu'une *frame1* est plus petite qu'une *frame2* si tous les rôles thématiques de *frame1* se retrouvent dans les rôles thématiques de *frame2*, mais que le contraire est faux. En l'occurrence, pour "**murder**" :

- *frame*(*Agent, Verbe, Patient*) < *frame*(*Agent, Verbe, Patient, Instrument*),
- *frame*(*Patient, Verbe, Instrument*) < *frame*(***Agent, Verbe, Patient, Instrument***).

Notre choix est donc de ne garder une *frame* comme résultat que si elle ne se trouve pas elle-même incluse dans une autre *frame* plus large. Nous modifions en conséquence l'implémentation de notre prédicat de haut niveau pour tenir compte de cette contrainte :

```

frame_murder(1, WN, Links, Verb, Agent, Patient, Instrument):-
frame_murder_1(WN, Links, Agent, Verb, Patient),
not(frame_murder_2(_, _, Agent, Verb, Patient, Instrument)).

frame_murder(2, WN, Links, Verb, Agent, Patient, Instrument):-
frame_murder_2(WN, Links, Agent, Verb, Patient, Instrument).

frame_murder(3, WN, Links, Verb, Agent, Patient, Instrument):-
frame_murder_3(WN, Links, Verb, Patient, Instrument),
not(frame_murder_2(_, _, Agent, Verb, Patient, Instrument)).

```

Conclusion

Nous avons vu comment traduire manuellement un fichier de VerbNet en un programme PROLOG équivalent. Cette traduction tient compte de l'héritage entre classes de verbes.

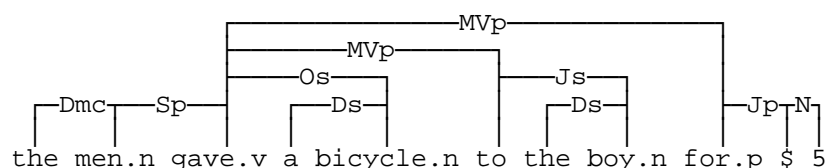
Prise en compte de constructions plus complexes

Limite du système actuel

A ce stade, nous ne savons reconnaître que des constructions simples où sujet, verbe et compléments se suivent d'une façon linéaire. Force est de constater que, dans un corpus réel, les phrases ne sont pas toujours aussi triviales. Nous avons donc exploré des techniques qui permettent de reconnaître des constructions grammaticales plus complexes.

Objectif

A partir d'une phrase comme “*the men gave a bicycle to the boy for \$5*” (« les hommes donnèrent une bicyclette au garçon pour 5 dollars »), notre système sait déjà extraire les rôles thématiques : *the men*_(Agent) *gave*_(Verb) *a bicycle*_(Theme) *to the boy*_(Recipient) *for \$*_(Asset) *5*.



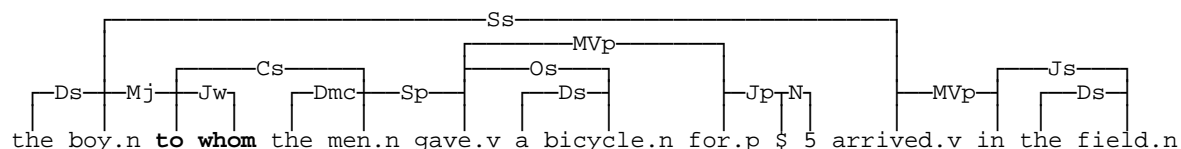
Notre objectif est de reconnaître plusieurs variantes dérivées de la construction de base (sujet, verbe, compléments). Les relatives nous ont semblé constituer un bon chantier d'expérimentation. Nous avons étendu nos programmes d'identification de schémas, pour généraliser la reconnaissance des liens :

- Entre sujet et verbe,
- Entre verbe et complément d'objet direct,
- Entre verbe et complément d'objet indirect introduit par une préposition.

De cette façon, nous sommes en mesure de reconnaître plusieurs *frames* simultanément dans une même phrase, et d'enrichir d'autant notre structure sémantique.

Un premier exemple avec extraction du complément

A partir de la phrase “*the boy to whom the men gave a bicycle for \$5 arrived in the field*” ou “*the boy that the men gave a bicycle to for \$5 arrived in the field*” (« le garçon à qui les hommes donnèrent une bicyclette pour 5 dollars arriva dans le champ »), nous reconnaissons deux constructions, l'une pour “**give**”, l'autre pour “**escape**”.



Classe de verbes “**give**” :

the boy_(Recipient) **to whom** **the men**_(Agent) **gave**_(Verb) **a bicycle**_(Theme) **for \$5**_(Asset) arrived in the field

Classe de verbes “**escape**” :

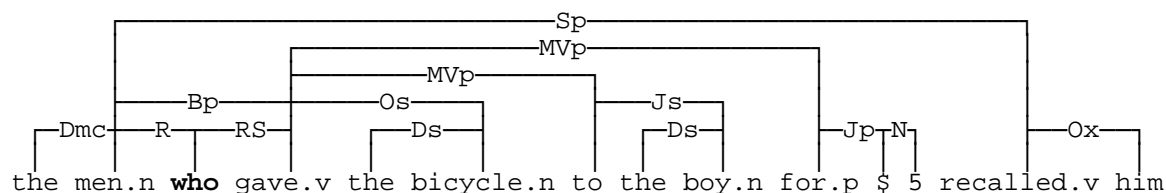
the boy_(Theme) **to whom** the men gave a bicycle for \$5 **arrived**_(Verb) **in the field**_(Oblique)

On remarquera qu'en fonction du contexte de la classe de verbes, un même mot (“*boy*”) peut jouer plusieurs rôles thématiques (*Recipient* ou *Thème*).

Dans cette construction, on constate que le complément extrait n'a pas de lien syntaxique direct avec le verbe. En revanche, il est relié au sujet, qui est lui-même relié au verbe, et c'est cette séquence caractéristique que nous recherchons.

Un second exemple avec une proposition relative enchâssée

Pour finir, donnons un second exemple de construction complexe, voisine de la précédente. A partir de la phrase “*the men who gave the bicycle to the boy for \$5 recalled him*” (« les hommes **qui** donnèrent une bicyclette au garçon pour 5 dollars le rappelèrent »), nous reconnaissons deux constructions, l'une pour “**give**”, l'autre pour “**banish**”.



Classe de verbes “**give**” :

the men_(Agent) **who gave**_(Verb) **the bicycle**_(Theme) **to the boy**_(Recipient) **for \$**_(Asset) **5** recalled him

Classe de verbes “**banish**” :

the men_(Agent) who gave the bicycle to the boy for \$ **recalled**_(Verb) **him**_(Theme)

Conclusion

Avec ce type de démarche, nous étendons les capacités d'analyse sémantique de notre système, et le rendons capable de comprendre davantage des phrases complexes, tirées de la « vie réelle ».

Automatisation de la traduction des descriptions de VerbNet en PROLOG

Motivation

Dans l'exemple décrit précédemment, *murder.xml* (170 lignes¹) a été traduit manuellement en *murder.pp* (165 lignes). Le travail nécessaire pour effectuer cette transformation a nécessité plus d'une journée, car il a d'abord fallu trouver la correspondance des concepts, et identifier tous les cas de figure à prendre en compte. Admettons qu'en régime de croisière, une fois les bons réflexes acquis, ce travail ne nécessite que deux heures, tests inclus². L'ensemble des 192 fichiers XML de VerbNet pèse 1,96 Mo ; *murder.xml* y contribue à hauteur de 6,65 Ko (soit 0,34%). Avec l'hypothèse de productivité précédente, une règle de trois montre qu'une traduction manuelle de l'ensemble prendrait 589 heures, soit 73 jours.

Nous concevons et mettons en œuvre, depuis plus de dix ans, des techniques et outils de génération de code à partir de modèles. Nous avons donc pris le pari d'industrialiser le

¹ Seulement 104 lignes sans la description de la sémantique (que nous n'avons pas utilisée pour l'instant).

² A titre indicatif, la productivité moyenne d'un développeur n'excède pas 100 lignes *par jour*, tests inclus, avec un langage de haut niveau.

processus décrit dans la partie précédente. L'objectif fixé est d'écrire un compilateur¹ de descriptions de VerbNet qui, à partir des fichiers décrivant des classes de verbes, sache générer automatiquement les programmes identifiant tous les schémas typiques de constructions de verbes. La motivation de cette approche repose sur :

- L'espoir qu'un tel compilateur permette un **gain de temps** : c'est-à-dire que son écriture nécessite moins de temps que les 73 jours prévus pour une tâche manuelle,
- Un objectif de **qualité du code** : contrairement à un être humain soumis à une tâche répétitive, un générateur automatique ne commet jamais d'erreur d'inattention,
- Un souci de **réutilisation** : si les descriptions de VerbNet sont modifiées ou enrichies, le compilateur régénère les programmes PROLOG en quelques minutes.

Difficultés rencontrées

Notre approche consiste, rappelons-le :

- D'abord à analyser syntaxiquement la (ou les) phrase(s) d'exemple d'une *frame* avec le *Link Grammar Parser*,
- Puis à synchroniser le résultat de cette analyse avec la syntaxe déclarée pour la *frame*, en établissant une correspondance entre les deux structures.

Chacun de ces points soulève des difficultés spécifiques, présentées ici.

Carences dans la grammaire du *Link Grammar Parser*

Mener à bien la première étape suppose que le *Link Grammar Parser* reconnaisse correctement la phrase d'exemple. Nous avons eu la (mauvaise) surprise de constater que, au départ, seulement la moitié des phrases d'exemples étaient correctement analysées.

Certaines des phrases d'exemples ("*Clouds cleared from the sky*", "*There appeared a ship on the horizon*"...) peuvent sembler tirées par les cheveux² pour un lecteur dont l'anglais n'est pas la langue natale. Quoiqu'il en soit, même si elles ne sont pas d'une construction courante, ces phrases sont syntaxiquement valides.

Plus gênant, des constructions beaucoup plus fréquentes sont également analysées avec des erreurs. Par exemple, dans "*the phone company billed me \$5*" (« l'entreprise de téléphonie m'a facturé 5 dollars ») le mot "*me*" n'est pas reconnu, alors que "*the phone company charged me \$5*" (qui veut dire la même chose) est correctement analysée en totalité.

Pour contourner cette première difficulté, nous avons deux possibilités :

- Mettre à jour la grammaire du *Link Grammar Parser* en précisant les règles d'appariement des verbes fautifs³,
- Simplifier les phrases d'exemple de VerbNet, typiquement en remplaçant un verbe fautif par un synonyme analysé correctement par le *Link Grammar Parser*.

¹ Nous revendiquons ce terme, un compilateur étant un traducteur d'un langage de haut niveau vers un autre langage de plus bas niveau (pas forcément de l'assembleur ou du code assembleur).

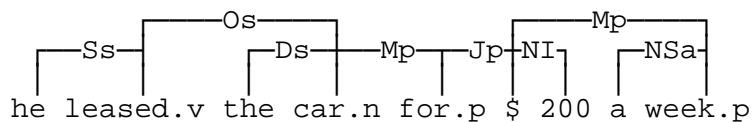
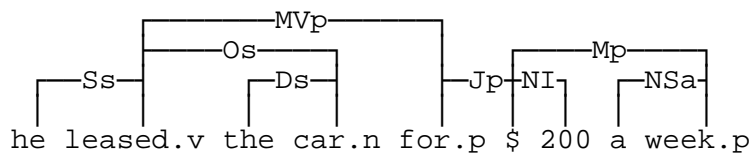
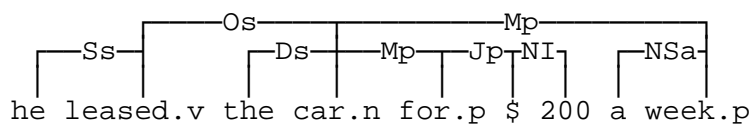
² Ou tout simplement poétiques ? (« Les nuages s'effacèrent du ciel », « Là apparut un navire à l'horizon »...)

³ Typiquement, VerbNet considère un verbe comme « aboyer » comment intransitif. Pourtant, on peut dire « aboyer un avertissement », même si cette construction en tant que verbe transitif est peu fréquente.

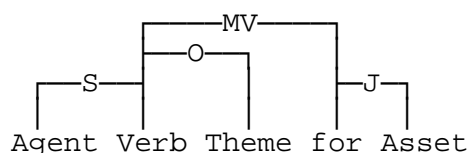
Pour des raisons de simplicité, nous avons opté pour la seconde solution. Nous avons donc consacré trois à quatre jours à simplifier systématiquement des centaines de phrases d'exemple. De cette façon, nous sommes passés de 50% à 90% de reconnaissance. Les 10% restants correspondent à des constructions peu fréquentes, qui pourront être traitées ultérieurement.

Synchronisation des deux structures syntaxiques

Synchroniser automatiquement les deux structures nécessite d'identifier, dans le résultat de l'analyse du *Link Grammar Parser*, une et une seule interprétation correspondant à la syntaxe déclarée dans VerbNet. Par exemple, l'analyse de la phrase "He leased the car for \$200 a week" (« il loua la voiture pour 200 dollars par semaine ») donne trois interprétations :



Notre compilateur doit déterminer quelle interprétation est compatible avec la syntaxe NP(Agent) VERB NP(Theme) PREP(for) NP(Asset). Ici, il s'agit de la deuxième interprétation :



Nous avons dû prendre en compte plusieurs dizaines de règles, et leur interaction, pour couvrir les différents cas de figure possible. Dans 90% des cas, nous trouvons ainsi l'interprétation syntaxique qui permet de synchroniser les deux structures¹.

Bilan sur l'automatisation

Au final, l'écriture du compilateur s'est révélé être un travail d'ingénierie plutôt complexe. Cette partie a pris une quinzaine de jours, au lieu des 73 jours nécessaires pour une traduction manuelle. L'investissement s'est donc révélé rentable.

A partir des 192 fichiers XML de VerbNet version 1.5, le compilateur génère automatiquement 31 600 lignes de code PROLOG (1,44 Mo).

¹ L'un des points délicats restant à couvrir concerne les verbes contenant des arguments symétriques qui ont alors deux rôles tels qu'*Acteur1* et *Acteur2*.

Désambiguïsation des noms dans un groupe nominal

Principe

Jusqu'ici, notre travail a essentiellement tourné autour du groupe verbal. Nous avons également défriché des méthodes utilisables pour désambiguïser des noms dans un groupe nominal.

Nous avons traité certaines constructions possessives telles que “*the hand of the man*” / “*the man's hand*” (« la main de l'homme ») ou “*the man with a hand*” (« l'homme avec une main »). Pour ce faire, nous essayons de trouver une éventuelle relation de méronymie¹ ou d'holonymie entre les deux noms reliés par une construction de ce type, en utilisant WordNet (Cf. l'illustration page 16).

Soient un nom A possédant plusieurs sens $A\#1, A\#2, A\#3, \text{etc.}$ et un autre nom B possédant également plusieurs sens $B\#1, B\#2, B\#3, \text{etc.}$ L'existence d'une telle relation de partie à tout permet d'exhiber un ou plusieurs couples $(A\#i, B\#j)$ de sens reliés deux par deux avec une affinité forte. La relation peut être plus ou moins directe :

- $A\#i$ est relié à $B\#j$ directement par une relation de méronymie,
- Un hyperonyme de $A\#i$ est relié à un hyperonyme de $B\#j$ par une relation de méronymie,
- $A\#i$ est relié à C par une relation de méronymie, et C est relié à $B\#j$ par une autre relation de méronymie, etc.

L'hypothèse linguistique que nous formulons² est que plus la distance entre $A\#i$ et $B\#j$ est faible, meilleure est la probabilité que ces deux sens soient effectivement liés. Cette technique permet donc de désambiguïser les noms au sein d'un groupe nominal.

Exemples

“The man's hand”

Examinons ce que cette approche donne sur un groupe nominal constitué de deux mots très polysémiques : “*the man's hand*”.

Le nom “*hand*” a 14 sens³ :

1. [BodyPart] *hand* (216), *manus*, *mitt*, *paw*: the extremity of the superior limb.
2. [OccupationalRole] *hired hand*, *hand* (5), *hired man*: a hired laborer on a farm.
3. [Text] *handwriting*, *hand* (4), *script*: something written by hand.
4. [Capability] *hand* (3): ability.

¹ La méronymie est une relation sémantique entre mots d'une même langue. C'est une relation partitive hiérarchisée, de partie à tout. Un méronyme A d'un mot B est un mot dont le signifié désigne une sous partie du signifié de B. Par exemple, *bras* est un méronyme de *corps*, de même que *toit* est un méronyme de *maison*. La relation inverse s'appelle l'holonymie.

² Hypothèse qui nous semble conforme à l'intuition, mais nous avons conscience que cela ne constitue en rien une preuve.

³ Dans les sens qui suivent, l'information entre [crochets] en début de ligne est le domaine de l'ontologie SUMO ; le chiffre entre parenthèses, décroissant de lignes en lignes, est la fréquence d'utilisation du mot.

5. [Region] *hand* (2): a position given by its location to the side of an object.
6. [Collection] *hand* (1), *deal*: the cards held in a card game by a given player.
7. [Attribute] *hand* (1): one of two sides of an issue.
8. [EngineeringComponent] *hand* (1): a rotating pointer on the face of a timepiece.
9. [UnitOfMeasure] *hand*: a unit of length equal to 4 inches; used in measuring horses.
10. [member] *hand*: a member of the crew of a ship.
11. [Human] *bridge player*, *hand*: a card player in a game of bridge.
12. [Expressing] *hand*: a round of applause to signify approval.
13. [BodyPart] *hand*: terminal part of the forelimb in vertebrates (such as kangaroos).
14. [Cooperation] *hand*, *helping hand*: physical assistance.

Le nom “*man*” a 11 sens :

1. [FullyFormed] *man* (1437), *adult male*: an adult male person as opposed to a woman.
2. [Government] *serviceman*, *military man*, *man* (432): someone who serves in the armed forces; a member of a military force.
3. [Human] *man* (275): the generic use of the word to refer to any human being.
4. [Human] *world*, *human race*, *humanity*, *humankind*, *human beings*, *humans*, *mankind*, *man* (205): all of the inhabitants of the earth.
5. [Hominid] *homo*, *man* (88), *human being*, *human*: any living or extinct member of the family hominidae.
6. [Male] *man* (36): a male subordinate.
7. [FullyFormed] *man* (6): an adult male person who has a manly character, virile.
8. [Male] *man* (2): a male person who plays a significant role (husband or lover or boyfriend) in the life of a particular woman.
9. [OccupationalRole] *valet de chambre*, *gentleman*, *gentleman's gentleman*, *man* (1): a manservant who acts as a personal attendant to his employer.
10. [Island] *Man*, *Isle of Man*: one of the British isles in the Irish Sea.
11. [Artifact] *man*, *piece*: game equipment consisting of an object used in playing certain board games.

Nous avons donc en tout 154 couples possibles. En revanche, seuls 16 de ces couples (10%) ont leurs deux éléments reliés par une relation de méronymie¹ :

(hand#1, man#5) Distance=1 [hand#1 ←*IsPart*→ man#5]

(hand#1, man#3) Distance=6 [hand#1 →*Hyperonym*→ extremity →*Hyperonym*→ external body part →*Hyperonym*→ body part ←*IsPart*→ organism ←*Hyponym*← person ←*Hyponym*← man#3]

(hand#1, man#1) Distance=7 [hand#1 →*Hyperonym*→ extremity →*Hyperonym*→ external body part →*Hyperonym*→ body part ←*IsPart*→ organism ←*Hyponym*← person ←*Hyponym*← male ←*Hyponym*← man#1]

(hand#1, man#7) Distance=7

(hand#1, man#2) Distance=8

(hand#1, man#6) Distance=9

(hand#1, man#8) Distance=9

(hand#1, man#9) Distance=10

(hand#13, man#3) Distance=10

(hand#13, man#5) Distance=10

¹ Nous en précisons le chemin complet à titre informatif pour les trois premiers couples. On remarquera que seuls les sens #1 et #13 de “*hand*” sont pertinents quand ils sont associés à “*man*”.

(hand#13, man#1) Distance=11
 (hand#13, man#7) Distance=11
 (hand#13, man#2) Distance=12
 (hand#13, man#6) Distance=13
 (hand#13, man#8) Distance=13
 (hand#13, man#9) Distance=14

Quelques autres exemples

Nous avons effectué les mesures suivantes sur quelques exemples. Comme on le voit, cette approche permet, dans le meilleur des cas, de complètement lever l'ambiguïté lexicale.

Phrase	Holonyme	Nbre sens	Méronyme	Nbre sens	Nombre de couples possibles	Nombre de couples avec méronymie
<i>The pocket of the jacket</i> (la poche de la veste)	<i>jacket</i>	5	<i>pocket</i>	9	45	1
<i>The car's wheels</i> (les roues de la voiture)	<i>car</i>	5	<i>wheel</i>	6	30	2
<i>The button of the radio</i> (le bouton de la radio)	<i>radio</i>	3	<i>button</i>	4	12	1

Autres pistes possibles

Bien d'autres heuristiques sont envisageables pour aider à désambiguïser des noms. On pourrait par exemple chercher à exploiter les conjonctions de coordinations. Par exemple, le nom « orange » peut désigner un fruit ou une couleur ; mais dans les deux phrases « j'aime l'orange et la pomme » et « j'aime l'orange et le vert », aucune ambiguïté n'est possible compte tenu de la présence de l'autre nom à droite du « et ».

L'idée sous-jacente est que deux mots reliés par une conjonction de coordination partagent souvent des caractéristiques communes. Soient un nom A possédant plusieurs sens $A\#1$, $A\#2$, $A\#3$, etc. et un autre nom B possédant également plusieurs sens $B\#1$, $B\#2$, $B\#3$, etc. Si on rencontre une construction de type « ... A et B... », « ... A ou B... » ou « ... ni A, ni B... », on peut calculer la similarité entre mots pour chaque couple ($A\#i$, $B\#j$), et retenir comme candidats possibles les couples ayant la distance la plus faible entre les sens de $A\#i$ et $B\#j$.

Plusieurs algorithmes de calcul de la similarité entre mots sont implémentés en utilisant les relations d'hyponymie de WordNet. Par exemple, le projet *WordNet::Similarity*¹ implémente les mesures de Resnik, Lin, Jiang-Conrath, Leacock-Chodorow, Hirst-St. Onge, Wu-Palmer, Banerjee-Pedersen, et Patwardhan-Pedersen.

Conclusion

L'identification de schémas peut également être mise en œuvre pour faciliter la désambiguïstation lexicale du groupe nominal.

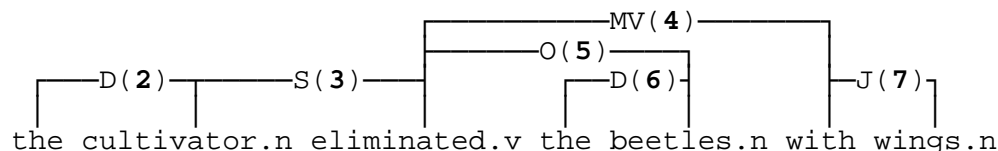
¹ Distribué sur <http://wn-similarity.sourceforge.net>.

Prise en compte simultanée des contraintes possibles

A l'issue de la phase d'identification de schémas, nous avons reconnu plusieurs constructions dans chaque interprétation syntaxique du texte de départ. Certaines interprétations peuvent se combiner ensemble ; d'autres s'excluent mutuellement. La question qui se pose alors est : lesquelles retenir ?

Un exemple

Pour la phrase “*the cultivator eliminated the beetles with wings*”, deux analyses syntaxiques sont possibles (chaque lien précise entre parenthèses son identifiant) :



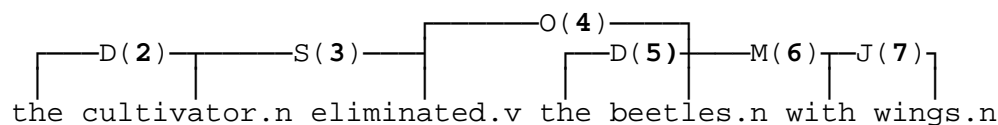
- Sur la première construction syntaxique (a), nous identifions les schémas suivants :

[*frame_murder_2_a_1*] the **cultivator**_(Agent) **eliminated**_(Verb) the **beetles**_(Patient) with **wings**_(Instrm)
 Contraintes de sélection : *cultivator*_(animate && concrete) *beetles*_(animate)
 Sens (WordNet) du verbe : #2
 Liens consommés : 3, 4, 5 et 7

[*frame_remove_1_a_1*] the **cultivator**_(Agent) **eliminated**_(Verb) the **beetles**_(Theme) with wings
 Contraintes de sélection : *cultivator*_(int control) *with*_(!src)
 Sens (WordNet) du verbe : #1, #3, #4 ou v#5
 Liens consommés : 3 et 5

[*article_a_1*] **the** cultivator eliminated the beetles with wings
 Lien consommé : 2

[*article_a_2*] the cultivator eliminated **the** beetles with wings
 Lien consommé : 6



- Sur la seconde construction syntaxique (b), nous identifions les schémas suivants :

[*frame_murder_1_b_1*] the **cultivator**_(Agent) **eliminated**_(Verb) the **beetles**_(Patient) with wings
 Contraintes de sélection : *cultivator*_(animate && concrete) *beetles*_(animate)
 Sens (WordNet) du verbe : #2
 Liens consommés : 3 et 4

[*frame_remove_1_b_1*] the **cultivator**_(Agent) **eliminated**_(Verb) the **beetles**_(Theme) with wings
 Contraintes de sélection : *cultivator*_(int control) *with*_(!src)
 Sens (WordNet) du verbe : #1, #3, #4 ou v#5
 Liens consommés : 3 et 4

[*article_b_1*] **the** cultivator eliminated the beetles with wings

Lien consommé : 2

[*article_b_2*] the cultivator eliminated **the** beetles with wings

Lien consommé : 5

[*holonym_meronym_b_1*] the cultivator eliminated the **beetles with wings**

Sens (WordNet) du nom holonyme : beetle #1

Sens (WordNet) du nom méronyme : wing #1

Lien consommé : 7

Les principales différences entre les deux constructions syntaxiques sont :

- La *frame* “**murder**” couvre quatre rôles thématiques dans la première construction, et seulement trois dans la seconde construction,
- Dans la seconde construction, on identifie une éventuelle relation supplémentaire de méronymie entre *beetle* et *wing*.

Algorithme énumérant les choix possibles

Nous allons présenter maintenant un algorithme qui permet d'énumérer toutes les combinaisons valides de schémas identifiés. Deux conditions doivent être vérifiées dans le passage de la forme syntaxique à la forme sémantique :

- Chaque lien doit être consommé une fois exactement. En effet :
 - Si un lien n'est pas consommé, c'est qu'on n'a pas complètement « compris » la phrase,
 - Si un lien est utilisé par plusieurs schémas, on ne peut de toute façon choisir qu'un seul d'entre eux,
- Pour un schéma donnée, soit tous ses liens sont consommés, soit aucun de ses liens n'est consommé (pas de consommation partielle d'un schéma).

La première condition a pour conséquence que tous les mots sont consommés. Pendant le processus de transformation, tous les nœuds et arcs du graphe syntaxique sont donc progressivement consommés.

Application à l'exemple

Dans la première construction syntaxique de l'exemple, [*frame_remove_1_a_1*] est incompatible avec [*frame_murder_2_a_1*] car tous deux utilisent les liens 3 et 5 ; mais si on retenait [*frame_remove_1_a_1*], les liens 4 et 7 ne seraient jamais consommés. Ces conditions imposent donc que la seule combinaison valide est :

- { [*frame_murder_2_a_1*], [*article_a_1*], [*article_a_2*] }

Dans la seconde construction, [*holonym_meronym_b_1*] est forcément utilisé (sinon le lien 7 ne serait jamais consommé), de même que [*article_b_1*] et [*article_b_2*]. Ensuite soit [*frame_murder_1_b_1*] soit [*frame_remove_1_b_1*] est choisie. Les combinaisons sont donc :

- { [*frame_murder_1_b_1*], [*article_b_1*], [*article_b_2*], [*holonym_meronym_b_1*] }
- { [*frame_remove_1_b_1*], [*article_b_1*], [*article_b_2*], [*holonym_meronym_b_1*] }

Mise en œuvre

Pour être complet sur l'algorithme énumérant les choix possibles, nous devons apporter deux précisions.

Tenir compte des restrictions imposées sur le sens des verbes et des noms

Il faut également tenir compte des éventuelles restrictions imposées sur le sens des verbes et des noms. Ces restrictions peuvent en effet également résulter en une incompatibilité. Par exemple, un même nom peut se voir imposer une contrainte de sélection *Humain* en tant que rôle thématique dans une construction, et une autre contrainte *Animal* dans le cadre d'une autre construction (qui ne recouvre pas la première) reconnue dans la même phrase.

Tenir compte des conjonctions de coordination

La condition « si un lien est utilisé par plusieurs schémas, on ne peut de toute façon choisir qu'un seul d'entre eux » doit être adaptée pour tenir compte des conjonctions des coordinations. En effet, dans une phrase comme “*she gave apples to Mary and Paul*”, le *Link Grammar Parser* produit une structure syntaxique où on identifie les schémas suivants :

[*frame_give_1_a_1*] *she*_(Agent) *gave*_(Verb) *apples*_(Theme) to *Mary*_(Recipient) and Paul
 Contraintes de sélection : *she*_(animate) *apples*_(!animate && !organization) *Mary*_(animate)
 Liens consommés : 2, 3, 4 et 5

[*frame_give_1_a_2*] *she*_(Agent) *gave*_(Verb) *apples*_(Theme) to *Mary* and *Paul*_(Recipient)
 Contraintes de sélection : *she*_(animate) *apples*_(!animate && !organization) *Paul*_(animate)
 Liens consommés : 2, 3, 4 et 6

Ces deux schémas partagent les liens 2, 3 et 4. Toutefois, il ne faut exclure aucun des deux. En cas de présence de conjonctions de coordination, il faut « fusionner » les schémas partageant une telle conjonction avant d'appliquer l'algorithme. En l'occurrence, on peut considérer qu'on a un unique schéma identifié :

[*frame_give_1_a_1_et_2*] *she*_(Agent) *gave*_(Verb) *apples*_(Theme) to *Mary*_(Recipient) and *Paul*_(Recipient)
 Contraintes de sélection : *she*_(animate) *apples*_(!animate && !organization) *Mary*_(animate) *Paul*_(animate)
 Liens consommés : 2, 3, 4, 5 et 6

Expérimentation

Ces différentes contraintes peuvent être vérifiées automatiquement par un solveur de contraintes, qui en les prenant en compte, énumère les solutions possibles (s'il en existe). Nous avons réalisé un prototype préliminaire qui implémente avec succès cet algorithme.

Indice de vraisemblance d'une interprétation de phrase

Définition

Les combinaisons valides de schémas reconnus, déclinées par les différentes combinaisons de sens des verbes et des noms, constituent l'ensemble des interprétations possibles pour une phrase. Nous pouvons définir un « indice de vraisemblance » pour chaque interprétation de phrase ; c'est un nombre qui est d'autant plus élevé qu'on reconnaît un grand nombre d'éléments pertinents dans le passage de la syntaxe à la sémantique. Nous pouvons alors classer les différentes interprétations d'une phrase selon leur indice de vraisemblance.

Calcul

Principes

L'indice de vraisemblance est d'autant plus élevé qu'on reconnaît un grand nombre d'éléments pertinents lors de l'identification de schémas. On essaie donc de déterminer des heuristiques¹ qui permettent de valoriser la reconnaissance d'une configuration particulière.

Heuristique utilisant le nombre de rôles thématiques reconnus

Par exemple, en ce qui concerne l'identification d'un verbe et des rôles thématiques associés, la première heuristique qu'il semble raisonnable d'établir est la suivante : l'indice de vraisemblance est directement proportionnel au nombre de rôles thématiques reconnus.

Dans l'exemple "*the cultivator eliminated the beetles with wings*", rappelons que les combinaisons valides de schémas identifiés sont² :

- { [frame_murder_2_a_1] }
- { [frame_murder_1_b_1], [holonym_meronym_b_1] }
- { [frame_remove_1_b_1], [holonym_meronym_b_1] }

Si on pondère la présence d'un rôle thématique (ou du verbe) par +10 et la présence d'un couple (holonyme/méronyme) par +15, ces combinaisons se retrouvent pondérées respectivement par :

- $4 \times 10 = +40$,
- $3 \times 10 + 15 = +45$,
- $3 \times 10 + 15 = +45$.

Heuristique utilisant la distance nécessaire pour réaliser le rôle thématique

Collins et Quillian ont proposés en 1969 d'utiliser les réseaux sémantiques pour décrire l'organisation des connaissances en mémoire. Ils ont montré qu'il faut plus de temps pour vérifier « un canari est un animal » (distance de deux niveaux dans un graphe d'héritage simplifié) que pour vérifier « un canari est un oiseau » (distance d'un niveau).

On peut formuler ici une hypothèse analogue portant sur la reconnaissance d'un rôle thématique ; nous considérons qu'un rôle thématique est d'autant plus pertinent qu'il a fallu parcourir une courte distance dans le graphe des hyperonymes pour le réaliser (avec l'approche décrite page 24 et suivantes).

Heuristique utilisant la fréquence d'apparition d'un mot

Toutes choses égales par ailleurs, il est plus probable d'utiliser un mot fréquent qu'un mot rare. Dans notre lexical, nous avons une information de fréquence provenant de WordNet. Nous pouvons l'utiliser dans le calcul de l'indice de vraisemblance.

¹ Une heuristique est l'utilisation de règles empiriques pratiques, simples et rapides, facilitant l'analyse d'une situation, dans un objectif de résolution de problèmes et de prise de décision.

² Dans les trois cas, la contribution des reconnaissances des articles étant identique, on les ignorera.

Conclusion

La méthode proposée pour calculer l'indice de vraisemblance s'appuie sur de nombreuses heuristiques. La difficulté consiste à les pondérer correctement. Nous n'avons pas disposé du temps nécessaire pour établir un protocole de validation adéquat pour contrôler ce point, qui méritera un approfondissement.

Conclusion

Résultats obtenus

Notre principal jeu de tests résidait dans l'ensemble des phrases d'exemple de VerbNet. Nous obtenons dessus un taux de reconnaissance de l'ordre de 90%, ce qui est encourageant.

Limitations actuelles

VerbNet est perfectible

Notre système reconnaît correctement la *frame* “murder” dans “*Paul*_(Agent) *killed*_(Verb) *Mary*_(Patient) *with a knife*_(Instrument)” (« Paul a tué Marie avec un couteau »). En revanche, il estime également que la phrase “*a knife*_(Instrument) *killed*_(Verb) *Mary*_(Patient)” (« un couteau a tué Marie ») est correcte ; or, bien que le couteau soit effectivement l'*Instrument* (en tant que rôle thématique¹) de la mort de Marie, on ne peut pas à proprement parler utiliser un instrument comme sujet de tuer.

Si nous établissons un contraste avec “*the tempest*_(Instrument) *killed*_(Verb) *Mary*_(Patient)” (« la tempête a tué Marie »), nous jugeons que cette dernière phrase est correcte. Nous arrivons à la conclusion qu'il faudrait certainement raffiner les contraintes de sélections, et ajouter une nouvelle contrainte *ForcePhysique* sur le rôle thématique *Instrument* dans la construction (*Instrument Verbe Patient*).

D'une façon plus générale, beaucoup d'ajustements permettraient d'améliorer VerbNet ; mais cette ressource, même imparfaite, a l'avantage d'exister et d'être librement disponible.

La prise en compte de constructions grammaticales complexes reste insuffisante

Nous avons commencé à prendre en compte des constructions grammaticales complexes (Cf. page 46). Toutefois, l'analyse correcte de « vraies » phrases en provenance de corpus est encore assujettie à la prise en compte de davantage de constructions grammaticales complexes.

Améliorations possibles

Utilisation de ressources de niveau pragmatique

D'autres ressources auraient également été intéressantes à exploiter dans notre contexte. Il s'agit par exemple des projets *ConceptNet*² et *Open Mind Commonsense*³ (menés au MIT) ou du projet *CYC*⁴ (dirigé par Doug Lenat).

¹ La définition du rôle thématique *Instrument* est « un objet ou une force physique qui provoque un changement dans quelque chose, généralement par contact direct ».

² <http://web.media.mit.edu/~hugo/conceptnet/>

³ <http://commonsense.media.mit.edu/cgi-bin/search.cgi>

⁴ <http://www.opencyc.org/>

De tels projets regroupent un ensemble d'informations de niveau pragmatique, sous forme d'assertions et de règles visant à capturer le sens commun humain : « les chiens sont petits », « les chiens ne peuvent pas voler », « les avions militaires transportent souvent des bombes »...

Forme syntaxique profonde

Nous gagnerions à utiliser une structure intermédiaire pour représenter la syntaxe profonde. Une telle structure faciliterait la prise en compte de constructions grammaticales complexes.

Editeur de schémas

Nous envisageons de développer un éditeur visuel permettant de définir à partir d'exemples de nouveaux schémas à identifier. La motivation de cette approche est d'augmenter la facilité d'utilisation, et de permettre de définir et maintenir rapidement des dizaines (voire des centaines) de schémas.

Travaux analogues

Dans les travaux en rapport avec les nôtres, le plus proche dans l'esprit est celui décrit dans [Shi & Mihalcea, 2005]. Le but de ce projet est identique à celui que nous avons mené, à savoir faire coopérer des ressources de large couverture sur la langue. Ils ont combiné WordNet, VerbNet, mais aussi FrameNet¹, pour construire un analyseur sémantique robuste.

[Dzikovska, 2004] présente un analyseur sémantique multi domaines qui part de messages oraux. Elle définit une algèbre sur les contraintes de sélection, ainsi qu'une méthode d'apprentissage des contraintes de sélection à partir d'un corpus.

[Mc Cord, 2004] présente un algorithme de désambiguïsation utilisant un analyseur syntaxique (English Slot Grammar) et WordNet, en exploitant notamment l'information de fréquence d'un mot.

Synthèse et perspectives

Ce projet de recherche a représenté un travail passionnant. Il a prouvé que des ressources de large couverture peuvent être fédérées et, qu'utilisées ensemble, elles permettent de construire un analyseur sémantique robuste. Il a aussi démontré l'intérêt d'automatiser la génération de code à partir des données des ressources.

Une fois ses limitations actuelles levées, un tel outil peut être utilisé pour de nombreuses tâches telles que l'extraction d'information ou un système de questions-réponses. Nous envisageons de l'appliquer dans le futur d'abord à la fusion de dictionnaires, puis à l'analyse d'une encyclopédie complète pour en extraire le contenu informationnel sous une forme exploitable par la machine.

¹ FrameNet (<http://framenet.icsi.berkeley.edu>) a pour principal objectif la création d'une ressource lexicale anglaise, basée sur la structure sémantique. FrameNet vise à documenter les nombreuses possibilités sémantiques et syntaxiques pour chaque mot pour chacune des significations possibles, grâce à des annotations manuelles. Ces annotations sont ensuite extraites et organisées. FrameNet étudie les mots, décrit la structure conceptuelle sous-jacente, examine les phrases en utilisant un vaste corpus de textes anglais contemporains contenant ces mots et enregistre les diverses manières dont l'information est exprimée dans ces diverses structures de phrases. Voir [Baker & Collin & Fillmore & Lowe, 1998] ou [Ruppenhofer, Ellsworth, Petruck & Johnson, 2005].

Bibliographie

- [Aubin, 2003] *Evaluation comparative de deux analyseurs produisant des relations syntaxiques*. <http://www.sciences.univ-nantes.fr/irin/taln2003/articles/eval2.pdf>
- [Baker & Collin & Fillmore & Lowe, 1998] *The Berkeley FrameNet project*.
- [Dzikovska, 2004] *A practical semantic representation for Natural Language Parsing*. <http://www.cs.rochester.edu/u/myros/thesis.pdf>
- [Fellbaum, 1998] *WordNet: an electronic lexical database*.
- [Fillmore, 1968] *The case for case*.
- [Gruber, 1965] *Studies in lexical relations*.
- [Jackendoff, 1972] *Semantic interpretation in generative grammar*.
- [Kahane, 2004] *Grammaires d'unification polarisées*.
- [Kipper-Schuler, 2003] *VerbNet: a broad coverage, comprehensive, verb lexicon*.
- [Levin, 1993] *English Verb Classes and Alternation: A Preliminary Investigation*.
- [Mc Cord, 2004] *Word Sense Disambiguation in a Slot Grammar Framework*. <http://domino.watson.ibm.com/library/CyberDig.nsf/0/b1c725cf5922df9785256f49004bb615?OpenDocument>
- [Miller, 1995] *Wordnet: A lexical database*.
- [Mueller, 2003] *Story understanding through multi-representation model construction* <http://www.signiform.com/erik/pubs/tm2003.pdf>
- [Mueller, 2004] *Understanding script-based stories using commonsense reasoning*. <http://www.signiform.com/erik/pubs/csr2004.pdf>
- [Niles & Pease, 2003] *Linking Lexicons and Ontologies: Mapping WordNet to the Suggested Upper Merged Ontology*. <http://ontology.teknowledge.com/nilesWordNet.pdf>
- [Pustejovski, 1995] *The Generative Lexicon*.
- [Ruppenhofer, Ellsworth, Petruck & Johnson, 2005] *FrameNet: Theory and Practice*. <http://framenet.icsi.berkeley.edu/book/book.pdf>
- [Shi & Mihalcea, 2005] *Putting Pieces Together: Combining FrameNet, VerbNet and WordNet for Robust Semantic Parsing*. <http://www.cs.unt.edu/~rada/papers/shi.cicling05.pdf>
- [Sleator & Temperley, 1991] *Parsing English with a Link Grammar*. <http://www.cs.cmu.edu/afs/cs.cmu.edu/project/link/pub/www/papers/ps/tr91-196.pdf>
- [Szolovits, 2004] *Adding a Medical Lexicon to an English Parser*. <http://medg.csail.mit.edu/projects/text/amia03c.pdf>

Annexe : de VerbNet à PROLOG

Exemple de fichier VerbNet : murder.xml

```

<VNCLASS ID="murder-42.1">
  <MEMBERS>
    <!-- décrit les verbes membres qui appartiennent à la classe -->
    <!-- WN précise l'identifiant du sens dans le lexique WordNet -->
    <MEMBER name="assassinate" wn="%2:41:00"/>
    <MEMBER name="butcher" wn="%2:35:00"/>
    <MEMBER name="dispatch" wn="%2:41:01"/>
    <MEMBER name="eliminate" wn="%2:30:00"/>
    <MEMBER name="execute" wn="%2:41:00 %2:41:01"/>
    <MEMBER name="immolate" wn="%2:40:00"/>
    <MEMBER name="liquidate" wn="%2:35:00"/>
    <MEMBER name="massacre" wn="%2:30:00"/>
    <MEMBER name="murder" wn="%2:41:00"/>
    <MEMBER name="slaughter" wn="%2:35:00 %2:30:00"/>
    <MEMBER name="slay" wn="%2:41:00"/>
  </MEMBERS>
  <THEMROLES>
    <!-- indique les rôles thématiques de la classe -->
    <THEMROLE type="Agent">
      <!-- précise les éventuelles contraintes de sélections -->
      <SELRESTRS>
        <SELRESTR Value="+" type="animate"/>
      </SELRESTRS>
    </THEMROLE>
    <THEMROLE type="Patient">
      <SELRESTRS>
        <SELRESTR Value="+" type="animate"/>
      </SELRESTRS>
    </THEMROLE>
    <THEMROLE type="Instrument">
      <SELRESTRS/>
    </THEMROLE>
  </THEMROLES>
  <FRAMES>
    <!-- indique chacune des constructions typiques -->
    <FRAME>
      <DESCRIPTION primary="Basic Transitive" xtag="0.2"/>
      <EXAMPLES>
        <!-- un ou plusieurs exemples -->
        <EXAMPLE>"Brutus murdered Julius Cesar"</EXAMPLE>
      </EXAMPLES>
      <SYNTAX>
        <!-- syntaxe de la frame -->
        <NP value="Agent">
          <SYNRESTRS/>
        </NP>
        <VERB/>
        <NP value="Patient">
          <SYNRESTRS/>
        </NP>
      </SYNTAX>
    </FRAME>
  </FRAMES>

```

```

</SYNTAX>
<SEMANTICS>
<!-- sémantique de la frame -->
<PRED value="cause">
  <ARGS>
    <ARG type="ThemRole" value="Agent"/>
    <ARG type="Event" value="E"/>
  </ARGS>
</PRED>
<PRED value="alive">
  <ARGS>
    <ARG type="Event" value="start(E)"/>
    <ARG type="ThemRole" value="Patient"/>
  </ARGS>
</PRED>
<PRED bool="!" value="alive">
  <ARGS>
    <ARG type="Event" value="result(E)"/>
    <ARG type="ThemRole" value="Patient"/>
  </ARGS>
</PRED>
</SEMANTICS>
</FRAME>
<FRAME>
<DESCRIPTION primary="Transitive" secondary="+ Instrument PP"/>
<EXAMPLES>
  <EXAMPLE>"Brutus killed Caesar with a knife"</EXAMPLE>
</EXAMPLES>
<SYNTAX>
  <NP value="Agent">
    <SYNRESTRS/>
  </NP>
  <VERB/>
  <NP value="Patient">
    <SYNRESTRS/>
  </NP>
  <PREP value="with">
    <SELRESTRS/>
  </PREP>
  <NP value="Instrument">
    <SYNRESTRS/>
  </NP>
</SYNTAX>
<SEMANTICS> ... </SEMANTICS>
</FRAME>
</FRAMES>
<SUBCLASSES>
<!-- éventuel regroupement en sous-classes de cas particulier -->
<VNSUBCLASS ID="murder-42.1-1">
  <MEMBERS>
    <MEMBER name="kill" wn="%2:35:00 %2:35:01 %2:42:00 %2:35:02"/>
  </MEMBERS>
  <THEMROLES>
    <THEMROLE type="Instrument">
      <SELRESTRS>
        <!-- les contraintes de sélections peuvent être affinées -->

```

```

    <SELRESTR Value="+ type="concrete"/>
  </SELRESTRS>
</THEMROLE>
</THEMROLES>
<FRAMES>
  <FRAME>
    <DESCRIPTION primary="Instrument Subject Alternation"/>
    <EXAMPLES>
      <EXAMPLE> "The pesticide killed the insects" </EXAMPLE>
    </EXAMPLES>
    <SYNTAX>
      <NP value="Instrument">
        <SYNRESTRS/>
      </NP>
      <VERB/>
      <NP value="Patient">
        <SYNRESTRS/>
      </NP>
    </SYNTAX>
    <SEMANTICS> ... </SEMANTICS>
  </FRAME>
</FRAMES>
<SUBCLASSES/>
</VNSUBCLASS>
</SUBCLASSES>
</VNCLASS>

```

Le programme PROLOG correspondant : murder.pp

```

/* MEMBERS - VNCLASS ID="murder_42_1" */

isMember_murder_42_1("assassinate", 2408713:nil).
isMember_murder_42_1("butcher", 1283599:nil).
isMember_murder_42_1("dispatch", 2408144:nil).
isMember_murder_42_1("eliminate", 457540:nil).
isMember_murder_42_1("execute", 2408980:2409921:nil).
isMember_murder_42_1("immolate", 2258341:nil).
isMember_murder_42_1("liquidate", 1287938:nil).
isMember_murder_42_1("massacre", 465851:nil).
isMember_murder_42_1("murder", 2408144:nil).
isMember_murder_42_1("slaughter", 1283599:465851:nil).
isMember_murder_42_1("slay", 2408144:nil).
isMember_murder_42_1(Verb, WN) :- isMember_murder_42_1_1(Verb, WN).

/* THEMATIC ROLES */

/* FirstDeclaration */
isAgent_murder_42_1(NAgent) :- testSelRestr(20 /* animate */, NAgent), !.

/* FirstDeclaration */
isPatient_murder_42_1(NPatient) :- testSelRestr(20 /* animate */, NPatient), !.

/* FirstDeclaration */
isInstrument_murder_42_1(NInstrument) :- .

/* FRAMES */

/* FRAMES ID="murder_42_1" */

/* FRAME "BasicTransitive" */
/* SYNTAX: NP(Agent) VERB NP(Patient) */
/* Agent Verb Patient */

```

```

/* CANONICAL ORDER: Agent Patient */
/* Syntagms before verb = 1 - after verb = 1 */
/* EXAMPLE 0 - "Brutus murdered Julius Cesar" */
/*
  graph TD
    Root[ ] --- Ss[Ss]
    Root --- Os[Os]
    Ss --- Brutus[Brutus]
    Os --- murdered[murdered.v]
    Os --- G[G]
    G --- Julius[Julius Cesar]
  
```

Brutus murdered.v Julius Cesar

```

SYNCHRO:
  Agent (LinkToVerb W1=Brutus L2=Ss)
  Verb (NoLink W2=murdered.v)
  Patient (LinkToVerb W4=Cesar L3=Os)
*/

frame_murder_1(1, WN, L2:L3:nil, NAgent, NVerb, NPatient) :-
link(L2, s, NAgent, NVerb),
link(L3, o, NVerb, NPatient),
isAgent_murder_42_1(NAgent),
isPatient_murder_42_1(NPatient),
word(NVerb, VerbDerivedForm, verb(Verb, VerbForm)),
isMember_murder_42_1(Verb, WN).

/* FRAME "Transitive + Instrument PP" */
/* SYNTAX: NP(Agent) VERB NP(Patient) PREP(with) NP(Instrument) */
/* Agent Verb Patient Instrument */
/* CANONICAL ORDER: Agent Patient Instrument */
/* Syntagms before verb = 1 - after verb = 2 */
/* EXAMPLE 0 - "Brutus killed Caesar with a knife" */
/*
  graph TD
    Root[ ] --- Ss[Ss]
    Root --- Os[Os]
    Root --- MVp[MVp]
    Root --- Js[Js]
    Ss --- Brutus[Brutus]
    Os --- killed[killed.v]
    Os --- Caesar[Caesar.n]
    MVp --- with[with]
    Js --- Ds[Ds]
    Ds --- knife[with a knife.n]
  
```

Brutus killed.v Caesar.n with a knife.n

```

SYNCHRO:
  Agent (LinkToVerb W1=Brutus L2=Ss)
  Verb (NoLink W2=killed.v)
  Patient (LinkToVerb W3=Caesar.n L4=Os)
  With (LinkToVerb W4=with L3=MVp)
  Instrument (LinkToPrevWord W6=knife.n L5=Js)
*/

frame_murder_2(2, WN, L2:L4:L3:L5:nil, NAgent, NVerb, NPatient, NInstrument) :-
link(L2, s, NAgent, NVerb),
link(L4, o, NVerb, NPatient),
link(L3, mv, NVerb, NWith),
word(NWith, "with", preposition(nil)),
link(L5, j, NWith, NInstrument),
lowerStrict(NPatient, NInstrument),
isAgent_murder_42_1(NAgent),
isPatient_murder_42_1(NPatient),
isInstrument_murder_42_1(NInstrument),
word(NVerb, VerbDerivedForm, verb(Verb, VerbForm)),
isMember_murder_42_1(Verb, WN).

/* VNCLASS ID="murder_42_1_1" */

/* MEMBERS */

isMember_murder_42_1_1("kill", 1284688:1286230:2667731:1286468:nil).

/* THEMATIC ROLES */

/* DeclaredInAncestor */
isAgent_murder_42_1_1(NAgent) :- isAgent_murder_42_1(NAgent), !.

/* DeclaredInAncestor */

```

```

isPatient_murder_42_1_1(NPatient) :- isPatient_murder_42_1(NPatient), !.

/* LocallyOverriden */
isInstrument_murder_42_1_1(NInstrument) :- isInstrument_murder_42_1(NInstrument),
testSelRestr(1 /* concrete */, NInstrument), !.

/* FRAMES */

/* FRAMES ID="murder_42_1_1" */

/* FRAME "InstrumentSubjectAlternation " */
/* SYNTAX: NP(Instrument) VERB NP(Patient) */
/* Instrument Verb Patient */
/* CANONICAL ORDER: Patient Instrument */
/* Syntagms before verb = 1 - after verb = 1 */
/* EXAMPLE 0 - "The pesticide killed the insects" */
/*

```

```

the pesticide.n killed.v the insects.n

SYNCHRO:
  Instrument (LinkToVerb W2=pesticide.n L3=Ss)
  Verb (NoLink W3=killed.v)
  Patient (LinkToVerb W5=insects.n L4=Op)
*/

frame_murder_3(3, WN, L3:L4:nil, NInstrument, NVerb, NPatient) :-
link(L3, s, NInstrument, NVerb),
link(L4, o, NVerb, NPatient),
isInstrument_murder_42_1_1(NInstrument),
isPatient_murder_42_1_1(NPatient),
word(NVerb, VerbDerivedForm, verb(Verb, VerbForm)),
isMember_murder_42_1_1(Verb, WN).

/* BRANCHES */

frame_murder_AgentPatient(NFrame, WN, Links, NVerb, NAgent, NPatient) :-
frame_murder_1(NFrame, WN, Links, NAgent, NVerb, NPatient).

frame_murder_AgentPatientInstrum(NFrame, WN, Links, NVerb, NAgent, NPatient, NInstrument) :-
frame_murder_2(NFrame, WN, Links, NAgent, NVerb, NPatient, NInstrument).

frame_murder_PatientInstrument(NFrame, WN, Links, NVerb, NPatient, NInstrument) :-
frame_murder_3(NFrame, WN, Links, NInstrument, NVerb, NPatient).

/* MAIN murder */

frame_murder(NFrame, WN, Links, NVerb, NAgent, NPatient, _Instrument) :-
frame_murder_AgentPatient(NFrame, WN, _Links, NVerb, NAgent, NPatient),
not(frame_murder_AgentPatientInstrum(_1, WN, _2, NVerb, NAgent, NPatient, NInstrument)),
flatten(_Links, Links).

frame_murder(NFrame, WN, Links, NVerb, NAgent, NPatient, NInstrument) :-
frame_murder_AgentPatientInstrum(NFrame, WN, _Links, NVerb, NAgent, NPatient, NInstrument),
flatten(_Links, Links).

frame_murder(NFrame, WN, Links, NVerb, _Agent, NPatient, NInstrument) :-
frame_murder_PatientInstrument(NFrame, WN, _Links, NVerb, NPatient, NInstrument),
not(frame_murder_AgentPatientInstrum(_1, WN, _2, NVerb, NAgent, NPatient, NInstrument)),
flatten(_Links, Links).

/* METAINFO */

get_signature("frame_murder", "Verb Agent Patient Instrument").

```