

Bruno Mery.
Sous la direction de MM. Pr. Christian Bassac et Pr. Christian Retoré.
Recherche scientifique publique.
Dernière mise à jour : Mercredi 25 Février 2009.

Perspectives pour la Sémantique Lexicale

Synthèse en questions-réponses

Résumé

Ce document ne constitue ni une publication, ni un rapport technique. Il est destiné à poser contextes et problématiques afférentes aux travaux réalisés dans le cadre de la thèse doctorale intitulée *Modélisation de la sémantique lexicale dans la théorie des types*, puis à donner à chaque aspect des explications les plus simples et complètes que possible. Ainsi, ce document en perpétuelle évolution devrait répondre aux interrogations sur l'état des travaux, leurs tenants et aboutissants, ou simplement leur sujet.

Après chaque question, un texte indique l'état actuel des réflexions sur la réponse (si elle existe); n'hésitez pas à contacter l'auteur (mery@labri.fr) pour demander plus d'éclaircissements, poser des questions supplémentaires, ou faire part de tout commentaire. Les fautes, erreurs et contresens sont du fait exclusif de l'auteur, et ne demandent qu'à être corrigés.

Table des matières

1	Générales	5
1.1	Quel est le cadre de ces travaux ?	5
1.2	Qu'est-ce que la linguistique computationnelle ?	5
1.3	Comment est effectuée l'analyse d'un texte ?	5
1.4	Y a-t-il d'autres problématiques que l'analyse ?	6
1.5	Qu'est-ce que la sémantique formelle ?	6
1.6	Qu'est-ce que la sémantique lexicale ?	7
1.7	Quels sont les principes sous-tendant le système ?	8
1.8	Quel est l'état des études sur les grammaires dans la théorie des types ?	9
1.9	Qu'est-ce qu'une ontologie, et en quoi son usage est-il intéressant ici ?	9
1.10	Doit-on vraiment avoir un langage abstrait pour décrire les langues ?	10
1.11	Quels travaux ont été retenus ?	10
1.12	Quels travaux ont été rejetés ?	11
2	Types	12
2.1	Pourquoi un système de types ?	12
2.2	Comment utiliser les types pour représenter des "facettes" ?	13
2.3	Quel langage pour les types ?	14
2.3.1	Les types pointés existent-ils ?	14
2.3.2	Utilise-t-on des types dépendants ?	15
2.4	Combien de sortes ?	15
2.5	Quelles sont les caractéristiques formelles de l'option retenue ?	15
3	Termes	16
3.1	Quel langage de termes est utilisé pour le calcul des prédicats ?	16
3.2	Quel est le principe permettant l'accès à l'ensemble des sens d'un lexème ?	17
3.3	Quel est le contenu d'une entrée lexicale ?	17
3.4	Comment se calcule l'interprétation d'un terme individuel ?	18
3.5	Quelle structure pour l'ontologie lexicale ?	19
3.6	Quelle est la complexité de Kolmogorov du lexique ?	20
3.7	Comment gérer la quantification et la référence ?	21
3.8	Y a-t-il des termes spécifiques (opérateurs) ?	21
3.9	Quel est le mode de calcul ?	21
3.9.1	Quelles sont les stratégies de réduction ?	21
3.9.2	Comment choisir la stratégie correcte ?	21
3.9.3	Les termes sont-ils utilisés de façon classique ou linéaire ?	21

3.9.4	Utilise-t-on le contexte à gauche ou à droite lors des calculs?	21
3.9.5	Comment gérer le non-déterminisme?	21
3.10	Quels sont les niveaux possibles de sous-spécification? . . .	21
3.11	Comment gérer les arguments optionnels?	21
3.12	Les niveaux précédents, morphologie et syntaxe, peuvent-ils interagir avec cette analyse?	21
3.13	Quelle est la complexité du calcul?	21
4	Formules	21
4.1	Quel est le langage des formules voulues?	21
4.2	Peut-on intégrer logique modale, temporelle, hyperintensionnelle... au système sans problèmes?	21
4.3	Quel système d'inférences pour les formules?	21
4.4	Peut-on borner la complexité de Kolmogorov d'une formule résultant d'une phrase donnée?	21
4.5	Quelles propriétés montrent que le système de types, termes et formules est pertinent?	21
5	Modèles	22
5.1	Quels sont les modèles d'interprétation possibles?	22
5.2	Quel est le modèle choisi?	22
5.3	Quelles sont les raisons de ce choix?	22
6	Filtres	22
6.1	Qu'est-ce qu'un filtre lexical?	22
6.2	Dans quelles circonstances un filtre lexical est-il utile?	22
6.3	Comment mettre en œuvre ce mécanisme?	22
7	Agents	22
7.1	Quels problèmes posent la multiplicité des agents dans l'analyse sémantique?	22
7.2	Pourquoi recourir à des représentations synthétiques de mondes différenciés dès ce stade d'analyse?	22
7.3	Comment fonctionne la théorie des microcosmes?	22
7.4	Quels sont les usages possibles de connaissances ainsi représentées?	22
8	Autres	23
8.1	Le lexique est-il fait pour une langue, ou pour toutes?	23
8.2	Comment établir un lexique correspondant à ces définitions?	23
8.2.1	Comment construire le lexique, à partir d'un lexique génératif existant?	23
8.2.2	Comment construire le lexique, à la main?	23

8.2.3	Comment acquérir automatiquement un tel lexique ?	23
8.2.4	Comment enrichir automatiquement un tel lexique ?	23
8.3	Peut-on utiliser ce formalisme pour la génération plutôt que pour l'analyse ?	23
8.4	Peut-on envisager une utilisation interactive en langue humaine ?	23
8.5	Dispose-t-on d'un prototype visant à l'implémentation d'un tel mécanisme ?	23
8.6	Quels sont les projets envisagés, les applications possibles ? .	23

1 Générales

1.1 Quel est le cadre de ces travaux ?

Statut : à vérifier.

Il s'agit d'un travail de thèse doctorale, dont le sujet est intitulé *Modélisation de la sémantique lexicale dans la théorie des types*, avec un encadrement principal en Informatique et un co-encadrement en Linguistique. En détail, le terme *Modélisation* implique un travail principalement théorique, consistant à donner un modèle formel qui tient compte d'un phénomène observé ; la *sémantique lexicale* est un champ très précis de la *linguistique computationnelle* qui décrit ce phénomène, et la *théorie des types* est un cadre mathématique donnant un ensemble de méthodes pour effectuer des calculs logiques robustes.

1.2 Qu'est-ce que la linguistique computationnelle ?

Statut : à vérifier.

La linguistique computationnelle, autrement appelée linguistique informatique, est une science consistant à systématiser la formalisation des langues humaines, jusqu'à l'obtention de formules mathématiques. Au confluent des domaines de la linguistique et de l'informatique, mais également des mathématiques, de la philosophie, de l'épistémologie et des neurosciences, elle cherche ainsi à donner les outils permettant aux programmes informatiques d'appréhender les informations transcrites en langues humaines, et de s'exprimer de la même manière.

Concrètement, elle se divise en de multiples champs : l'étude de la *syntaxe*, de la *morphologie*, ou de la *sémantique*, par exemple. Elle dispose d'un versant applicatif qui est très productif, appelé *traitement automatique des langues* ; ce domaine d'applications utilise les résultats intermédiaires de la linguistique computationnelle pour réaliser correcteurs d'orthographe, modules de reconnaissance vocale ou d'aide à la traduction... qui progressent d'autant plus vite que la compréhension des théoriciens sur les mécanismes fondamentaux de la langue avancent.

1.3 Comment est effectuée l'analyse d'un texte ?

Statut : à vérifier.

L'analyse d'un texte, c'est-à-dire la traduction d'un texte d'une langue humaine vers une représentation informatique, peut commencer par une phase de *reconnaissance vocale* ou de *reconnaissance d'écriture*, si le média analysé n'est pas dans un format directement traitable (texte entré au clavier). Ensuite, une première phase *d'analyse structurelle et prosodique* peut être mise en place, afin de déterminer, avant même de s'intéresser aux mots, l'impact des conventions et "décorations" (ponctuation, espacement, gras, souligné... ou emphase, dans le cas d'un médium oral) sur la suite des ana-

lyses – *a minima*, d'effectuer un premier découpage en phrases. Vient alors une analyse de la *morphologie*, permettant de repérer, d'une part, les *flexions* grammaticales qui déterminent (suivant la langue) qu'un mot est employé avec un genre, un nombre ou un cas particulier, ou, pour les verbes, un temps ou un mode, informations qui seront précieuses ; d'autre part, les constructions telles que néologismes ou mots composés. La phase suivante est celle de l'analyse *syntactique*, consistant à utiliser l'ensemble des informations disponibles et les règles de grammaire de la langue utilisée pour construire l'*arbre syntactique* de la phrase, en déterminant où se situent, entre autres, sujet et compléments. L'analyse de la syntaxe permet alors de déterminer quels termes sont soumis (en tant qu'adjectifs, compléments du nom, adjoints...) à quels autres, et ainsi d'établir une hiérarchie. Ensuite vient la phase d'*analyse sémantique* de la phrase, qui consiste à combiner le sens des mots pour former celui de la phrase, en utilisant l'arbre donné par sa syntaxe pour produire une forme logique. Et enfin, cette forme logique est *interprétée* dans un modèle adapté, suivant l'usage voulu.

1.4 Y a-t-il d'autres problématiques que l'analyse ?

Statut : à vérifier.

Si les travaux des linguistes informaticiens (et en particulier, les nôtres) sont souvent dirigés vers l'analyse, il y a plusieurs autres problématiques dans notre domaine. En particulier, le processus symétrique de l'analyse est appelé *génération* ; il consiste, à partir d'un but déterminé par le programme, à produire une forme logique correspondant à ce que le programme a à exprimer, puis à générer l'arbre syntactique correspondant, et, en utilisant les règles et le vocabulaire adaptés, à construire un texte. La *traduction automatique* consiste (du moins en théorie) en une analyse d'un texte, suivie de la génération de la forme logique obtenue – mais en utilisant deux langues différentes pour l'analyse et la génération.

1.5 Qu'est-ce que la sémantique formelle ?

Statut : à vérifier.

Le champ particulier de la *sémantique formelle* est l'étude de la *signification* des phrases et textes. Dans le cas de la linguistique computationnelle, il s'agit, à partir d'un texte dont la syntaxe (conjugaison, accords...) est déjà établie, de donner une représentation utilisable par des outils informatiques du *sens* de ce texte. De multiples méthodes sont utilisées pour ce faire. Depuis très longtemps, le paradigme dominant par sa simplicité d'utilisation et les résultats immédiats qu'il propose est celui de la *sémantique vériconditionnelle*, où chaque texte, phrase ou partie de phrase (proposition) est traduit en une *formule logique*, dotée d'une *valeur de vérité*. À partir de sa connaissance du monde, un programme peut alors très facilement déterminer si cette phrase est vraie ou fausse.

Quelle que soit le but ultime de l'analyse sémantique (prouver que la phrase est vraie, ou aller plus loin), cette méthode, proposée par Montague (pour la première fois dans [Montague, 1974]) et maintes fois développée depuis, dispose d'avantages indéniables. En premier lieu, parce qu'elle vise la construction de *formes logiques*, qui sont d'une part probablement proches du processus de raisonnement chez l'humain (selon Chomsky, dans le célèbre [Chomsky, 1993]), et surtout directement utilisables pour des calculs supplémentaires. En second lieu, parce qu'elle est *compositionnelle* : le calcul se fait bout par bout, en partant du sens d'un mot (donné par le *lexique*, équivalent formel d'un dictionnaire), pour calculer le sens d'une locution (la combinaison de deux mots), d'une proposition (formée de plusieurs locutions), d'une phrase (formée de plusieurs propositions), et ainsi de suite jusqu'au sens du texte.

1.6 Qu'est-ce que la sémantique lexicale ?

Statut : à vérifier. La *sémantique lexicale* est une partie de la sémantique formelle qui s'intéresse particulièrement au sens des *mots*. Son étude, relativement récente comparativement aux recherches de linguistique computationnelle, est due à l'observation de certains phénomènes insolubles par l'approche suivie jusqu'alors.

En effet, la sémantique de Montague suppose que chaque mot dispose d'un sens sous forme d'un terme d'une certaine logique, terme qui se combine ensuite à d'autres de façon calculatoire pour former une formule, qui représente le sens du texte. Malheureusement, les langues humaines sont plus complexes que le paradigme de base utilisé ici : pour certains mots, il existe *plusieurs sens* (vérifier dans votre dictionnaire préféré...).

Une solution simple est d'adopter une stratégie proche d'un dictionnaire : donner, pour chaque mot, une liste de tous les sens, et choisir le bon en fonction du *contexte*, c'est-à-dire des mots adjacents. On règle ainsi les cas d'*homographie contrastive* : on peut différencier le *bar* (débit de boissons) du *bar* (ressource halieutique). Plusieurs systèmes, basés sur les statistiques, utilisent cette méthode avec un certain bonheur.

Cependant – et c'est là toute notre problématique – ces *lexiques à énumération de sens* ne permettront jamais de régler les cas de *polysémie logique*, ou un même mot, un même lexème (et non deux mots ayant la même écriture ou prononciation par un hasard historique ou linguistique) dispose de plusieurs sens différents mais logiquement liés. Prenons le terme *école* : il peut être compris de multiples façons, en tant que bâtiment (*l'école est au bout de la rue*), qu'établissement public (*l'école ouvre à 8 :30*), que groupe de personnes (*l'école est de sortie à la montagne*), ou même sous-groupe de personnes (*l'école est en grève*) ou institution nationale (*l'école est en péril*). Chacun de ces sens est différent, et cette différence doit être notée dans le calcul de la sémantique de la phrase. Mais ce ne sont pas des sens sans lien les uns

avec les autres. De plus, on peut *créer* de nouveaux sens logiquement liés aux précédents, sans limite véritable (*l'école est rapide*), par *composition* avec d'autres termes.

En se basant sur ces constats, et sur une étude de cas très complète, [Pustejovsky, 1995] a proposé une sémantique lexicale basée sur un lexique complexe, permettant, non pas de *lister* les sens de chaque mot de façon exhaustive, mais de *générer*, à partir d'informations intrinsèques à chaque terme, l'ensemble de ces sens – mais de façon *compositionnelle*. Ainsi, le formalisme entier de l'analyse sémantique doit être revu pour permettre des changements de sens pour les mots au cours même de l'analyse.

1.7 Quels sont les principes sous-tendant le système ?

Statut : à vérifier.

Les principes donnés par [Pustejovsky, 1995] restent à la base de notre approche (nous allons les rappeler très rapidement). Cependant, nous savons que les tentatives d'utiliser cette théorie directement, bien qu'ayant donné lieu à des études très intéressantes, se sont soldées soit par un échec ([Vanier et al., 2006]), soit par une approximation ([Gupta and Aha, 2005]), et ce en raison de l'immensité de la tâche. Nous avons donc décidé de prendre pour principe directeur, non pas la *création* d'un système d'analyse sémantique, mais *l'amélioration* de mécanismes existants et bien fondés, en y ajoutant petit à petit les éléments nécessaires. Ainsi, nous nous basons sur la sémantique Montagovienne bien connue (des linguistes informaticiens), et ses raffinements récents. Pour les besoins du système, nous avons modifié :

- Le système de types,
- Le lexique,
- Le langage de termes et leur mode d'application,
- Le langage des formules ciblées,
- Le modèle d'interprétation.

Les principes de [Pustejovsky, 1995] sont, esquissés très rapidement, les suivants : chaque terme du lexique comprend, dans sa description même, l'ensemble des informations qu'un locuteur de la langue pourrait donner à son sujet en-dehors de tout contexte. Ainsi, il est *richement typé* (on comprend que le verbe *manger* n'admette qu'un sujet *animé* et un objet *nourriture*) ; certains termes comportent une *structure événementielle* (le verbe *construire* suppose un avant, pendant et après le processus de construction, et un résultat final), et on définit une *structure de qualia* (d'après Aristote) permettant de déterminer les propriétés intrinsèques du terme (une *enclume* est *lourde*), son usage ou comportement typique (une *cigarette* se *fume*), ses constituants caractéristiques (une *voiture* comporte un *moteur*), et son origine typique (un *article* a un *auteur*). La somme de ces informations permet à un système informatique de faire les relations qu'un locuteur de

la langue effectuée sans aide : *une cigarette rapide* → *fumer rapidement une cigarette*, *une voiture puissante* → *une voiture au moteur puissant*, *un article conservateur* → *un article dans lequel l'auteur exprime des opinions conservatrices*, etc. Sans ces relations, le résultat de chacune de ces phrases n'aurait pas de sens : comment une cigarette peut-elle se déplacer rapidement, ou un article avoir une opinion personnelle ?

1.8 *Quel est l'état des études sur les grammaires dans la théorie des types ?*

Statut : à vérifier.

Actuellement, la vision des choses est un peu différente de celle exprimée précédemment. Plutôt que d'une chaîne grammaticale allant du texte brut à l'interprétation de sa sémantique, les scientifiques pensent que la hiérarchie est plutôt fondée sur une *forme profonde*, appelée *tectogrammaire*, qui représente la base primaire de l'organisation d'un texte. De cette forme profonde sont dérivées deux autres formes : le texte lui-même, soumis aux mécanismes d'une langue en particulier par le moyen de la *phénogrammaire*, qui régit ce qui est communément appelé syntaxe, grammaire, orthographe... et la forme logique, adaptée aux raisonnements, dérivée de la forme tectogrammaticale par la *sémantique*.

Cependant, cette vision ne change pas les principes de notre analyse, puisque nous ne nous occupons pas des impacts phénogrammaticaux sur l'analyse de la sémantique.

1.9 *Qu'est-ce qu'une ontologie, et en quoi son usage est-il intéressant ici ?*

Statut : à vérifier.

Une *ontologie* est un outil différent selon que la philosophie ou l'informatique l'utilisent. Il s'agit d'une représentation de *l'univers*, dans laquelle toute chose est cataloguée et classée ; pour le philosophe, il s'agit d'un idéal inaccessible, alors que pour l'informaticien, c'est un outil quotidien – il s'agit d'une divergence dans la notion d'*univers*.

Ici, le but n'est pas de représenter tout ce qui existe, mais simplement tout ce qui se *dit* : une ontologie d'une ou plusieurs langues. Le principe est de classer dans une structure arborescente les mots, d'une manière qui soit logique pour les concepts qu'ils représentent.

Pour [Pustejovsky, 1995], et pour nous-mêmes, l'intérêt d'utiliser une ontologie est de réduire la place prise par un lexique aussi riche que ce qui a été défini. La manière de classer les termes dans cette ontologie est simple : un terme peut hériter d'un autre s'il partage suffisamment de caractéristiques de son "ancêtre", et qu'il en apporte d'autres. Une branche de cet arbre pourrait être : à la racine, *tout*, qui généralise chacun des termes et n'a aucune propriété, puis *entité* (par opposition à *prédicat*), *objet physique* (qui dispose de constituants matériels, peut être transporté, etc.), *artefact*

(construit par un agent), *véhicule* (capable de déplacement et dont le but est le transport), *voiture* (comportant roues, moteur. . .), *Honda* (spécification du constructeur).

1.10 *Doit-on vraiment avoir un langage abstrait pour décrire les langues ?*

Statut : à vérifier.

Le fait de choisir un langage abstrait, avec ce lexique contenant des termes d'un certain langage de types, visant à formuler des calculs et, au final, des formules, est une prise de position controversée en soi. En effet, certains philosophes et linguistes pensent qu'il est illusoire de croire qu'il existe un langage autre que la langue humaine, qui serait utilisé dans le cerveau ; le principe de réduction des hypothèses inutiles leur indique que, comme les langues humaines sont, par essence, suffisamment riches pour se décrire elles-mêmes, il n'y a pas besoin d'autres langues.

Notre propos n'est pas de prendre position sur le fonctionnement du cerveau humain lors de l'utilisation de la langue, même s'il m'apparaît à titre personnel assez ambitieux de croire que seul des termes des langues sont utilisés. En effet, ces travaux portent avant tout sur la réalisation d'un modèle, qui peut ne pas refléter la réalité des processus utilisés par les locuteurs, mais doit rendre fidèlement compte des comportements observés. Pour ce faire, il doit être immédiatement applicable, afin de vérifier les hypothèses qu'il implique ; or, un modèle basé sur l'auto-description de la langue pré-suppose une phase très étendue d'apprentissage, qui pose des problèmes allant bien au-delà du sujet de ces travaux.

Si nous avons choisi de représenter les langues à l'aide d'un langage abstrait, formel, et directement très proche du mode algorithmique de fonctionnement d'un programme informatique plutôt que d'utiliser les langues qui constituent notre objet d'étude, ce n'est donc pas par *a priori* sur la nature de ce dernier, mais par pure commodité, afin de faciliter la démarche expérimentale.

1.11 *Quels travaux ont été retenus ?*

Statut : à compléter.

Comme indiqué à de multiples reprises, [Pustejovsky, 1995] a été retenu comme étant la base de notre approche. Nous avons ensuite étudié de multiples propositions. [Vanier et al., 2006] nous a servi de point de départ pour les formalisations, et [Pustejovsky and Boguraev, 1996], [Pustejovsky and Bouillon, 1996], [Pustejovsky, 2006b], [Pustejovsky, 1998], [Pustejovsky, 2001], [Pustejovsky, 2006a], [Jacquey, 2001], ont été autant d'études extrêmement précieuses pour nos travaux. Dans un autre contexte, [Dowty, 1989] a été utile pour permettre de préciser certains aspects de la sémantique. [Nunberg, 1993] est un article fondamental pour une idée très importante de notre approche : le fait que prédicat ou ar-

gument puissent, de façon égale, contribuer des informations qui permettent de modifier l'un ou l'autre par transferts de sens. [Smith, 2003] et [Guarino, 1998] sont deux études présentant des points de vues complets et contrastés sur la notion et l'utilisation concrète des ontologies, et [Nirenburg et al., 1995] permet de répondre aux nombreuses critiques formulées à ce sujet. [Girard et al., 1989] et, en particulier, [Girard, 1972] sont les références fondamentales du système logique que nous allons utiliser. D'une manière plus fondamentale, [Searle, 1979] présente les prémisses de toutes les théories linguistiques précitées : le savoir préalable.

[Cooper, 2007] est une approche connexe, intéressante pour la logique utilisée. [Marlet, 2007] a défriché les interactions entre sémantique calculatoire et Lexique génératif, et [Asher, 2008] est le dernier d'une très importante série de travaux visant à la formalisation du lexique génératif, utilisant une autre approche que la notre. Enfin, [Gupta and Aha, 2005] et [Gupta and Aha, 2003] proposent une implémentation concrète et fonctionnelle des principes précités, qui prouve la faisabilité de cette entreprise – même si très peu de la théorie originelle a été transcrite, et que de très nombreuses heuristiques aient été ajoutées.

En dernier lieu, les travaux actuels ont fait l'objet de publications préliminaires : [Mery et al., 2007a], [Mery et al., 2007b], [Bassac et al., TBP], et de plusieurs rapports non publiés, dont principalement [Mery, 2008].

1.12 *Quels travaux ont été rejetés ?*

Statut : à compléter.

[Blutner, 2002] est une critique qui semble infondée des études précédentes, l'auteur souhaitant – très artificiellement, de notre point de vue – reléguer au niveau *pragmatique* (c'est-à-dire du raisonnement en fonction des situations concrètes) les phénomènes pourtant fortement liés au langage.

Comme indiqué précédemment, [Asher, 2008] est un article très important dans le développement formel des théories de [Pustejovsky, 1995], et même plus. Cependant, nous rejetons une grande partie des bases de la modélisation qu'il effectue. Il est également fondé sur de nombreux travaux préliminaires : [Pustejovsky and Asher, 2000] ou [Asher and Pustejovsky, 2005], tous deux posant de très importants problèmes formels.

Parallèlement – et malheureusement à l'insu des auteurs des articles précédents, un autre effort important a été réalisé pour une formalisation logique de [Pustejovsky, 1995] dans [Pinkal and Kolhase, 2000]. Cet effort est fondé sur des prémisses logiquement fallacieuses et n'est pas utilisable en l'état, mais aurait pu donner lieu à un formalisme très élégant.

Enfin et très récemment, [Saba, 2007] (ainsi que d'autres études du même auteur sur le même sujet) utilise une approche extrêmement si-

milaire à [Pustejovsky, 1995] pour proposer une méthode formelle d'acquisition et d'interprétation générale de la langue. L'auteur rejette cependant l'idée qu'un lexique aux entrées de taille fini puisse être construit de cette manière, et reste dans la description d'un mécanisme excessivement général ; d'après nous, l'efficacité de cette approche ne peut pas suffire aux contraintes terre-à-terre d'un système d'analyse automatique.

2 Types

2.1 Pourquoi un système de types ?

Statut : à vérifier.

Un système de types est une composante de certaines logiques. Dans un cadre fréquemment employé en linguistique computationnelle, la *logique intuitionniste*, les types sont un mécanisme de vérification de la correction d'une formule. En effet, le calcul est effectué à l'aide de *termes* que l'on peut classer en deux catégories : *foncteurs* et *arguments*, une étape habituelle du calcul consistant en la combinaison d'un terme foncteur et d'un terme argument en un terme réduit, opération appelée *application* (d'un terme à l'autre) ou *réduction* (du nombre de termes). Ainsi, un terme $\lambda x.(P x)$, appliqué à un terme T , donnera, après réduction, $(P T)$.

(Les notations utilisées ici sont du λ -calcul : le λ est un *opérateur d'abstraction*, qui permet de spécifier l'étiquette d'une *variable* – ici, x – qui sera remplacée par l'argument lors d'une application. On note l'application par un couple entre parenthèses dont le premier élément est le foncteur, le second l'argument. Par la suite, on notera les types en exposant des termes concernés.)

Avec un système de types comportant les types élémentaires A, B, C , soient les termes $\lambda x^A.(P^{A \rightarrow B} x)$, T^A et R^C . Dans un calcul contraint par ce système (le λ -calcul simplement typé), l'application $(P T)$ est valide, mais pas l'application $(P R)$: R n'est pas un argument du type attendu par le foncteur. Ce mécanisme est couramment utilisé par les *langages de programmation* pour s'assurer de la validité des programmes informatiques.

Plus intéressante est la propriété connue sous le nom d'*isomorphisme de Curry-Howard*, qui établit une correspondance directe entre le calcul opératoire des termes et la logique des types utilisée : ici, le terme bien typé $(P^{A \rightarrow B} T^A)^B$ est une *preuve* de la déduction suivante, en logique intuitionniste :

$$\frac{A \rightarrow B \quad A}{B}$$

Tous les termes bien typés du λ -calcul typé sont ainsi des preuves de déductions valides en logique intuitionniste, et toute déduction démontrable en logique intuitionniste dispose d'au moins un terme bien typé qui en est la preuve. Cette équivalence est la base de résultats théoriques et pratiques à très grande portée.

Revenons à la sémantique formelle. [Montague, 1974] utilise un système de types basé sur la logique intuitionniste, avec deux types de base, appelés respectivement e (le type des entités) et t (le type des valeurs de vérité). Cette sémantique dispose, par essence, d'interprétations vériconditionnelles, car les *noms* disposent d'un type e et les *prédicats* d'un type $e \rightarrow t$ ("produire une valeur de vérité pour un argument de type entité"), et les propositions sont de type t (vraie ou fausse). Ainsi, *voiture* est de type e , *rouge* est de type $e \rightarrow t$, et la sémantique de "voiture rouge" est donc $(\text{rouge}^{e \rightarrow t} \text{voiture}^e)^t$. L'interprétation de cette sémantique dans un modèle est la valeur "vraie" ou "fausse", suivant qu'il existe effectivement ou non une voiture rouge dans le modèle.

Il est bien entendu possible d'utiliser ce canevas pour avoir des interprétations plus intéressantes que la simple valeur de vérité de la phrase ; ce système a été très étudié, et permet en tout cas de donner une sémantique bien fondée, avec des calculs relativement directs.

[Pustejovsky, 1995] propose également un système de types, mais pour d'autres raisons. En effet, donner un type à un terme revient à lui donner une catégorie, et l'auteur identifie les catégories à des comportements distincts, des données partagées par un grand ensemble de termes... Ainsi, il identifie un type *Agent*, *Entity*, *Event*, *Artifact*, etc. Cette approche n'est pas très innovante en soit, et a le mérite d'éviter de considérer comme valides des propositions comme "la voiture est conjoncturellement disjonctive". Pour Pustejovsky, ces types sont les éléments de base de son lexique, et des mécanismes de sémantique lexicale. De plus, il est assez simple de les intégrer à l'approche de Montague – simplement en considérant plus de deux types.

2.2 Comment utiliser les types pour représenter des "facettes" ?

Statut : à reprendre entièrement.

Le but de notre approche est d'utiliser les types, à la fois à la manière de Montague et à la manière de Pustejovsky. Nous nous démarquons cependant de ce dernier en n'utilisant pas de *types pointés* (voir section 2.3.1, p. 14), et en ne faisant pas apparaître dans le système de types en lui-même les divers composantes du lexique génératif. En effet, ces données nous apparaissent redondantes avec le lexique. De plus, notre système de termes fait appel, pour la représentation des multiples *facettes* d'un lexème, à l'utilisation de termes "optionnels" qui peuvent provenir de multiples sources : le type d'un terme peut être un guide, mais on ne peut garantir qu'il contiendra l'ensemble des opérations nécessaires. Il vaut alors mieux ne pas représenter les facettes par des informations de types, mais utiliser des types atomiques, et représenter les facettes dynamiquement, à l'aide d'autres mécanismes.

2.3 Quel langage pour les types ?

Statut : à vérifier.

Nous allons utiliser une logique de types (et donc un système de termes) proche de celle utilisée par Montague et ses suivants, et appelée (voir [Hinderer, 2008]) TY_n ou “théorie des types à n sortes”. Son principe est identique aux systèmes Montagoviens : il existe $n + 1$ types élémentaires, dont n d’entre eux sont définis dans un lexique et le dernier est t , le type des valeurs de vérité. Ensuite, en supposant que α et β sont des types, $\alpha \rightarrow \beta$ est également un type. Cette définition est récursive, et donc $\alpha \rightarrow \alpha \rightarrow \alpha \rightarrow \beta$ est également un type, par exemple.

Nous considérons ne pas avoir besoin d’autres opérations sur les types que \rightarrow , appelée *implication intuitionniste* par sa valeur dans la logique correspondante. [Pustejovsky, 1995], ainsi que les propositions de Nicholas Asher, utilisaient une construction supplémentaire : nous ne l’avons pas retenue, voir pourquoi immédiatement. . .

2.3.1 Les types pointés existent-ils ?

statut : à vérifier. Un cas particulier a fortement gêné la formalisation des théories de [Pustejovsky, 1995], et a amené l’auteur à proposer une modification du système de types qui s’est révélée lourde de conséquences formelles, notamment dans [Pustejovsky and Asher, 2000] ou [Asher and Pustejovsky, 2005]. Cette construction est dite “dot type” ou “type pointé”, notée \bullet , et est utilisée pour représenter les *objets pointés*.

Un *objet pointé* est un objet qui, apparemment, dispose de plus d’un type – et dans lequel il n’est pas forcément évident qu’il y ait un type dominant, et qui échappe également aux critères Aristotéliens définis par l’auteur. L’exemple canonique est *livre* : ce mot représente couramment un *objet physique* (avec pages, couverture. . .) et une *information* (avec entrée en matière, conclusion, un auteur ; on peut parler d’un livre en cours d’écriture). Pour permettre la gestion de cette catégorie de lexèmes, l’auteur propose de considérer les livres comme ayant le type $P \bullet I$ (P est le type des objets physiques, I celui des informations abstraites). Cependant, et de façon systématique, les règles du système de types intégrant cet opérateur – qui est défini comme un produit, mais ne doit pas se comporter comme tel – posent de graves problèmes. Voici un exemple de règle posant problème, rentrant en ligne de compte quand un terme de type pointé est utilisé – version de 2005 :

$$\frac{\{\lambda P\phi(P(x)), c(P : (\alpha \bullet \beta) \multimap \gamma) [\psi, c' (\psi : \left[\begin{smallmatrix} \alpha' \\ \beta' \end{smallmatrix} \right] \multimap \gamma)] , \text{head}(\psi)\}}{\left\{ \lambda P\phi \left[\frac{\exists v (\Delta(\phi, x) \left[\frac{v}{x} \right] \wedge \text{O-Elab}(x, v))}{\Delta(\phi, x)} \right] , c * \left(x : \left[\begin{smallmatrix} \alpha \sqcap \alpha' \\ \beta \sqcap \beta' \end{smallmatrix} \right] , v : \alpha \bullet \beta \right) \right\} [\psi, c']}$$

Notons qu’il doit s’agir d’une *règle élémentaire*, qui remplace l’application

intuitionniste, $\frac{A \rightarrow B}{B} A$. Comme l'opération $\Delta(\phi, x)$ est un algorithme dont la décidabilité même n'est pas garantie ("la plus petite sous-formule dans ϕ qui soit responsable du type original de x "), la complexité des calculs est fortement remise en question. . .

De plus, si considérer un objet comme ayant deux types simultanément pour pouvoir le considérer sous l'un ou l'autre suivant les cas est séduisant lors de l'élaboration d'une théorie, c'est beaucoup moins pertinent d'un point de vue formel. En effet, le principe même de la sémantique lexicale est de permettre le changement de types d'un objet, *modulo* une opération adaptée, suivant l'usage désiré ; choisir un type arbitraire et permettre de passer à l'autre est certes insatisfaisant d'un point de vue philosophique, mais entièrement transparent après calcul. . . de plus, une étude plus précise des cas concernés laisse à penser qu'il y a systématiquement une dissymétrie entre les deux types envisagés au départ (ainsi, on ne parle presque jamais de *livre* sans évoquer le contenu informatif).

Nous avons déterminé une alternative viable à cette construction (section 3.2, p. 17), et ne l'utiliserons donc pas – même s'il faut reconnaître que les phénomènes qui ont conduit à sa formulation sont objectivement de nature à envisager ce type de solution.

2.3.2 Utilise-t-on des types dépendants ?

Statut : à vérifier. Notre solution a une catégorie de problèmes – dont les objets pointés – utilisent le λ -calcul d'ordre supérieur de Girard (une version du Système-F), donc avec des types dépendants. Par exemple, $\Lambda \alpha \lambda x^B. (P^{B \rightarrow \alpha} x)$ est un terme dont le premier argument est un *type* : cette construction permet d'effectuer des adaptations de très haut niveau. Avec les restrictions adaptées, les propriétés de ce calcul sont bien connues.

Le système de calcul avec types dépendants sera donné plus loin. Cependant, il n'est utile qu'au moment du calcul en lui-même : il n'y a pas de types dépendants dans le résultat final, non plus que dans le lexique (mis à part les termes fonctionnels comme "et").

2.4 Combien de sortes ?

Statut : élaborer et reprendre.

Nous utilisons TY_n , mais quelle est la valeur de n ? De nombreuses variables dépendent de ce chiffre. . . Or, nous ne pouvons pas faire mieux qu'indiquer que n dépend du lexique. Suivant la complexité de l'ontologie, il peut même être très élevé.

2.5 Quelles sont les caractéristiques formelles de l'option retenue ?

Statut : de très nombreux points à clarifier ici (décidabilité, convergence, etc.), à remettre dans le contexte général, etc. À écrire.

3 Termes

3.1 Quel langage de termes est utilisé pour le calcul des prédicats ?

Statut : à vérifier.

Le second aspect de notre système formel est le langage de *termes*. Après avoir défini les types qui les régissent, il faut préciser la nature des objets, foncteurs et arguments, qui seront manipulés lors du calcul de la représentation sémantique des locutions. Le langage de termes gouverne également le lexique (car c'est à partir du lexique que seront extraits les termes qu'il s'agira de composer), et les formules logiques (car c'est à partir du résultat du calcul compositionnel des termes qu'il faudra extraire la formule logique qui sera la représentation sémantique recherchée).

Le système de types évoqué précédemment permet de retenir un certain nombre de choix : nous utilisons le *calcul des prédicats du second ordre typé à n sortes* (TY_n), avec *types dépendants* à la Girard (Système F). Autrement dit, notre langage de termes dispose :

- d'*individus* représentés par des constantes ou des variables : a, b, c, x, y, z, \dots
- de *prédicats* constants ou variables : P, Q, \dots
- d'un *typage* des individus et prédicats : $(P^{\alpha \rightarrow \gamma} x^\beta)$ est un terme valide si et seulement si $\alpha \multimap \beta$ (la relation peut ne pas être l'égalité stricte de types, selon le système retenu)
- de *types dépendants* : $\Lambda \alpha \gamma \lambda P^{\alpha \rightarrow \gamma} x^\alpha . (P x)$ représente ainsi, pour tous les types possibles, l'ensemble des applications
- une *quantification* sur les individus $(\lambda x . (P x))$, sur les types $(\Lambda \alpha \lambda x^\alpha . x)$, et sur les prédicats $(\lambda P . (P x))$.

Ces éléments suffisent à définir le langage de termes. Pourquoi l'employer plus qu'un autre ? Voici à quoi correspond chaque élément :

- Un individu constant est un objet d'un type dérivé d'*entité* : objet courant ou abstraction, qui peut être objet ou sujet d'une phrase ; il est défini tel que dans le lexique.
- Un prédicat constant est un terme correspondant à un *événement* ou une *propriété*, qui peut être verbe, adjectif ou adverbe d'une phrase, et modifie un individu ; il est défini tel que dans le lexique.
- Un individu variable fait l'objet d'une quantification ou abstraction (*toute personne* : $\forall x . (\text{personne } x, \text{ etc.})$), ou indique le type attendu par un prédicat (x mange y : $\lambda x^A \lambda y^F . ((\text{manger}^{F \rightarrow (A \rightarrow t)} y) x)$).
- Un prédicat variable représente la possibilité d'une *modification* de sens. C'est ce mécanisme particulier qui nous amène à utiliser le calcul du second ordre, ainsi que les types dépendants ; nous verrons pourquoi et comment ci-après (section 3.2, p. 17).

En dehors de cette dernière modification, le calcul se limite à la logique du premier ordre, avec prédicats et individus simplement typés.

3.2 Quel est le principe permettant l'accès à l'ensemble des sens d'un lexème ?

Statut : à vérifier.

Nous n'utilisons pas les types pour représenter les facettes d'un lexème (voir section 2.2, p. 13), car les principes utilisés habituellement pour ce faire sont impraticables (voir section 2.3.1, p. 14). En lieu et place de ces mécanismes, nous définissons et utilisons un principe portant sur l'utilisation et l'insertion, à l'aide des constructions du second ordre et de types dépendants introduite dans notre définition du langage de types (section 2.3.2, p. 15) et de termes (section 3.1, p. 15), de *termes optionnels* que nous appelons couramment *morphismes*.

Mathématiquement, un *morphisme* est un autre mot pour "fonction", ayant des consonnances particulières. Dans le cadre de ces travaux, un morphisme est un terme qui en *transforme* un autre, en permettant d'accéder explicitement à une de ses facettes ; par le mode de calcul utilisé, il s'agit d'un *terme optionnel*.

Concrètement, supposons le terme $ecole^E$, avec E le type des établissements (tout terme de ce type dispose de facettes *groupe de personnes* ou *bâtiment*, et les écoles disposent d'une distinction fine entre *groupe d'enseignants* et *groupes d'élèves*, entre autres. Pour représenter l'ensemble de ces facettes, on propose l'ensemble de morphismes suivant : $\{f_{enseignants}^{E \rightarrow P}, f_{eleves}^{E \rightarrow P}, f_{locaux}^{E \rightarrow B}\}$. Alors, les locutions suivantes sélectionneront les termes suivants :

- *l'école est en grève* : $(f_{enseignants} \ ecole)^P$
- *l'école est en vacances* : $(f_{eleves} \ ecole)^P$
- *l'école est au bout de la rue* : $(f_{locaux} \ ecole)^B$

On appelle ces morphismes des *termes optionnels* car l'un, plusieurs ou aucun peuvent être utilisés dans le calcul. Ainsi, on garde l'aspect générique :

- *l'école est en péril* : $ecole^E$

Reste à préciser comment sont mis en place le calcul et la procédure de choix des morphismes à utiliser. On peut d'ores et déjà remarquer que, si on se base exclusivement sur le type des termes (i.e. seules des personnes peuvent être en grève ou en vacances), les associations suivantes sont aussi valides :

- *l'école est en grève* : $(f_{eleves} \ ecole)^P$
- *l'école est en vacances* : $(f_{enseignants} \ ecole)^P$

Ce qui ne constitue pas forcément une erreur d'interprétation.

3.3 Quel est le contenu d'une entrée lexicale ?

Statut : à vérifier.

Les entrées du lexique doivent permettre d'utiliser les termes associés à chaque lexème dans le calcul compositionnel qui s'ensuit : non seulement le terme représentant la sémantique principale du lexème, mais également

l'ensemble des termes optionnels définis en section 3.2, p. 17, qui permettront d'accéder à l'ensemble des facettes sémantiques associées. De multiples manières d'ordonner l'ensemble des données nécessaires à ce sujet apparaissent.

Dans chaque entrée lexicale, nous avons besoin :

- D'une représentation adaptée du lexème en tant que suite de caractères, ainsi éventuellement que de traits morphosyntaxiques (esquissés section 3.12, p. 21) permettant de le repérer sous ses multiples formes.
- Éventuellement, de données permettant sa distinction immédiate de lexèmes différents mais homomorphes.
- Le λ -terme principal, associé à un hypothétique "sens premier" (souvent non spécifié) du lexème. Le terme et son type permettent également d'obtenir la liste de ses éventuels arguments obligatoires.
- Éventuellement, une liste de termes "alternatifs" permettant la gestion des *arguments optionnels*. Voir section 3.11, p. 21.
- La liste des morphismes induits par le lexème, couplés à une éventuelle contrainte sur le choix de la stratégie de réduction que chacun nécessite (voir en 3.9.2, p. 21 pour ce dernier point).
- Enfin, une description de l'interprétation de chacun des termes, voir section 3.4, p. 18.

Une entrée lexicale sera donc représentée *a minima* comme suit (exemple du lexème *Paris*) :

$$Paris :: \left(Paris^T, \frac{\lambda x^T . (f_L^{T \rightarrow L} x)}{\emptyset}, \frac{\lambda x^T . (f_P^{T \rightarrow P} x)}{\emptyset}, \frac{\lambda x^T . (f_G^{T \rightarrow G} x)}{global} \right)$$

3.4 Comment se calcule l'interprétation d'un terme individuel ?

Statut : à vérifier.

Le lexique doit permettre le calcul sémantique d'une formule logique représentant le sens de l'ensemble du texte analysé ; cependant, il doit également contenir les données permettant, lors de l'*interprétation* de cette formule (et donc l'utilisation concrète du sens donné par la phrase), d'interpréter chaque terme. En particulier, il doit permettre d'interpréter directement le terme principal et chaque terme optionnel donné dans l'entrée lexicale définie en 3.3, p. 17.

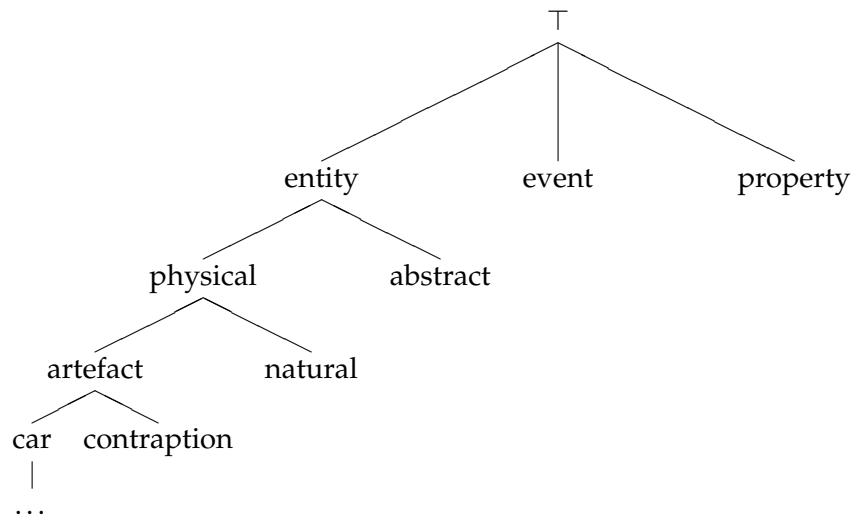
Les données d'interprétation sont fonction du modèle choisi. Afin de conserver une approche modulaire, nous ne précisons donc pas ce point ici : on pourrait avoir une structure de traits permettant de réaliser un modèle de Herbrand (approche choisie en 5.2, p. 22), un individu ou une fonction pour cadrer à une interprétation en théorie des ensembles, ou une glose pour disposer d'une interprétation en langue humaine ; les différents choix possibles sont nombreux, voir section 5.1, p. 22.

3.5 Quelle structure pour l'ontologie lexicale ?

Statut : à vérifier.

Nous utilisons la même structure ontologique que celle définie par Pustejovsky (section 1.7, p. 8), pour les raisons évoquées en section 1.9, p. 9. Il s'agit principalement d'un choix pragmatiques. Étant conscients que la mise au point d'une ontologie dans laquelle la place de chaque terme n'est pas contestable (pour différentes raisons, un concept ou un lexème peut se retrouver classifié à divers endroits d'une taxonomie), notre principe n'est pas d'effectuer des différenciations calculatoires sur la base de cette ontologie, mais simplement de l'utiliser comme un moyen de synthétiser le lexique.

L'ontologie du lexique peut être représentée comme sur le fragment suivant :



Voir [?] pour des détails d'une telle construction. Au sens strict, cette ontologie représente, à nos yeux, un ensemble de *types*, plus ou moins génériques (certains nœuds) et de lexèmes (les autres nœuds, les feuilles), et permet de synthétiser le lexique de la manière suivante :

- Chaque élément, sauf \top , dispose également des termes optionnels apportés par l'ensemble de ses ancêtres. En effet, une voiture se déplace à la manière d'un véhicule.
- Chaque élément dispose d'un terme optionnel permettant d'accéder directement à chacun de ses ancêtres. En effet, une formule mathématique dispose des facettes *raisonnement*, *information* et *abstraction* accessibles très naturellement.

Le résultat est généralement redondant, ce qui n'est pas un grave problème. Au-delà des principes d'établissement d'une telle ontologie, celle-ci n'a pour nous que le but purement fonctionnel de réduire la taille du lexique.

3.6 Quelle est la complexité de Kolmogorov du lexique ?

Statut : à reprendre en utilisant des termes mathématiques corrects.

La complexité de Kolmogorov est définie par la taille de la plus petite chaîne de caractères permettant de représenter l'objet considéré. Comme le lexique est fortement variable, et *a priori* inconnu, cette complexité est bornée par la taille maximale d'un lexique. Y compris avec la factorisation forte permise par l'ontologie précisée en 3.5, p. 19, on ne peut affirmer de propriétés sur la distribution de données communes au sein du lexique ; on doit donc considérer indépendamment chaque entrée lexicale, en raisonnant comme si elle instanciat l'ensemble de ses ancêtres.

Une entrée lexicale, dans notre définition la plus petite, est un terme principal, typé, et une suite de termes optionnels. Sans borner ses paramètres, les entrées n'ont pas de taille maximale ; il est cependant raisonnable d'instaurer certaines bornes. Sans parler de longueur maximale pour un des paramètres, il semble possible d'en borner l'espérance. Soient m une valeur bornant la moyenne des arguments sur le lexique, l une valeur bornant le nombre moyen des termes optionnels. Tous les termes optionnels que nous utilisons ont une longueur bornée (il s'agit de morphismes, ils n'ont donc qu'un argument). La taille d'un terme étant fonction du nombre de ses arguments, on a alors la taille d'une entrée lexicale moyenne bornée par $O(m + l)$. Pour un lexique de n termes, en y ajoutant des termes spécifiques définis en 3.8, p. 21 de nombre borné par $o(n)$, la taille totale du lexique est de $kn(m + l)$, k constante correspondant à la complexité de Kolmogorov d'un terme à au plus un argument.

Sur réserve d'études statistiques de la langue, on peut avoir raisonnablement, comme valeurs :

- $n = 1000000$, ordre de grandeur du nombre de mots dans les langues actuelles,
- $k = 30$. En effet, avec un identifiant unique pour le terme, l'abstraction, la donnée de la variable, et les parenthèses, on a au plus 10 caractères plus la longueur de l'identifiant, qui, si on dispose de n termes différents, est de 20 pour avoir l'unicité ($\log_2(1000000) = 20$, à peu de choses près) ; on considère qu'un caractère est représenté en un octet, unité implicite.
- $l = 10$. Il s'agit de la mesure la plus sujette à caution, mais, suivant le lexique génératif, les lexèmes disposent de 4 facettes possibles (les qualia), plus une si le type est pointé ; en pratique, il est possible d'en dégager plus pour certains termes.
- $m = 3$. Une grande partie des termes sont sans arguments ou à un seul argument (noms, adjectifs...) ; les verbes transitifs mettent en jeu deux arguments explicites, et il faut ajouter divers modes d'adjonction et de complémentation non systématiques. La valeur "3" doit permettre de gérer la quasi-intégralité des lexèmes.

Selon ces estimations, un lexique à couverture complète, tel qu'envisagé ici, occuperait moins de 400Mo d'espace disque.

3.7 *Comment gérer la quantification et la référence ?*

Statut : réflexion à reconstruire entièrement avant de la proposer.

3.8 *Y a-t-il des termes spécifiques (opérateurs) ?*

3.9 *Quel est le mode de calcul ?*

3.9.1 *Quelles sont les stratégies de réduction ?*

3.9.2 *Comment choisir la stratégie correcte ?*

3.9.3 *Les termes sont-ils utilisés de façon classique ou linéaire ?*

3.9.4 *Utilise-t-on le contexte à gauche ou à droite lors des calculs ?*

3.9.5 *Comment gérer le non-déterminisme ?*

3.10 *Quels sont les niveaux possibles de sous-spécification ?*

3.11 *Comment gérer les arguments optionnels ?*

3.12 *Les niveaux précédents, morphologie et syntaxe, peuvent-ils interagir avec cette analyse ?*

3.13 *Quelle est la complexité du calcul ?*

4 *Formules*

4.1 *Quel est le langage des formules voulues ?*

4.2 *Peut-on intégrer logique modale, temporelle, hyperintensionnelle... au système sans problèmes ?*

4.3 *Quel système d'inférences pour les formules ?*

4.4 *Peut-on borner la complexité de Kolmogorov d'une formule résultant d'une phrase donnée ?*

4.5 *Quelles propriétés montrent que le système de types, termes et formules est pertinent ?*

5 *Modèles*

5.1 *Quels sont les modèles d'interprétation possibles ?*

5.2 *Quel est le modèle choisi ?*

5.3 *Quelles sont les raisons de ce choix ?*

6 *Filtres*

6.1 *Qu'est-ce qu'un filtre lexical ?*

6.2 *Dans quelles circonstances un filtre lexical est-il utile ?*

6.3 *Comment mettre en œuvre ce mécanisme ?*

7 *Agents*

7.1 *Quels problèmes posent la multiplicité des agents dans l'analyse sémantique ?*

7.2 *Pourquoi recourir à des représentations synthétiques de mondes différenciés dès ce stade d'analyse ?*

7.3 *Comment fonctionne la théorie des microcosmes ?*

7.4 *Quels sont les usages possibles de connaissances ainsi représentées ?*

8 Autres

8.1 *Le lexique est-il fait pour une langue, ou pour toutes ?*

8.2 *Comment établir un lexique correspondant à ces définitions ?*

8.2.1 *Comment construire le lexique, à partir d'un lexique génératif existant ?*

8.2.2 *Comment construire le lexique, à la main ?*

8.2.3 *Comment acquérir automatiquement un tel lexique ?*

8.2.4 *Comment enrichir automatiquement un tel lexique ?*

8.3 *Peut-on utiliser ce formalisme pour la génération plutôt que pour l'analyse ?*

8.4 *Peut-on envisager une utilisation interactive en langue humaine ?*

8.5 *Dispose-t-on d'un prototype visant à l'implémentation d'un tel mécanisme ?*

8.6 *Quels sont les projets envisagés, les applications possibles ?*

Références

- [Asher, 2008] Asher, N. (2008). A Type Driven Theory of Predication with Complex Types. *Fundamenta Informaticæ*, 84(2) :151–183.
- [Asher and Pustejovsky, 2005] Asher, N. and Pustejovsky, J. (2005). Word Meaning and Commonsense Metaphysics. Semantics Archive.
- [Bassac et al., TBP] Bassac, C., Mery, B., and Retoré, C. (TBP). Towards a Type-Theoretical Account of Lexical Semantics. *Journal of Language, Logic, and Information*. To appear.
- [Blutner, 2002] Blutner, R. (2002). Lexical Semantics and Pragmatics. *Linguistische Berichte*.
- [Chomsky, 1993] Chomsky, N. (1993). A Minimalist program for linguistic theory. In Hale, K. and Keyser, S. J., editors, *The view from Building 20 : Essays in linguistics in honor of Sylvain Bromberger*, pages 1–52. MIT Press, Cambridge, Massachusetts.
- [Cooper, 2007] Cooper, R. (2007). Copredication, dynamic generalized quantification and lexical innovation by coercion. In *Fourth International Workshop on Generative Approaches to the Lexicon*.
- [Dowty, 1989] Dowty, D. R. (1989). On the semantic content of the notion of ‘thematic role’. In Chierchia, G., Partee, B. H., and Turner, R., editors, *Properties, Types and Meaning, Volume II : Semantic Issues*. Kluwer Academic Publishers.
- [Girard, 1972] Girard, J. Y. (1972). Interprétation fonctionnelle et élimination des coupures de l’arithmétique d’ordre supérieur. Thèse de Doctorat d’État, Université Paris VII.
- [Girard et al., 1989] Girard, J.-Y., Taylor, P., and Lafont, Y. (1989). *Proofs and types*. Cambridge University Press, New York, NY, USA.
- [Guarino, 1998] Guarino, N. (1998). Formal ontology and information systems.
- [Gupta and Aha, 2003] Gupta, K. M. and Aha, D. M. (2003). Nominal Concept Representation in Sublanguage Ontologies. In *Second International Workshop on Generative Approaches to the Lexicon*.
- [Gupta and Aha, 2005] Gupta, K. M. and Aha, D. W. (2005). Interpreting Events Using Generative Sublanguage Ontologies. In *Third International Workshop on Generative Approaches to the Lexicon*.
- [Hinderer, 2008] Hinderer, S. (2008). *Automatisation de la Construction Sémantique dans TYN*. PhD thesis, Université Henri Poincaré – Nancy 1.
- [Jacquey, 2001] Jacquey, E. (2001). *Ambiguïtés lexicales et traitement automatique des langues : modélisation de la polysémie logique et application aux déverbaux d’action ambigus en français*. PhD thesis, Université de Nancy 2.

- [Marlet, 2007] Marlet, R. (2007). When the Generative Lexicon meets Computational Semantics. In *Fourth International Workshop on Generative Approaches to the Lexicon*.
- [Mery, 2008] Mery, B. (2008). Perspectives sur la Sémantique Lexicale : Principes descriptifs. Technical report.
- [Mery et al., 2007a] Mery, B., Bassac, C., and Retoré, C. (2007a). A montagovian generative lexicon. In *Formal Grammar*.
- [Mery et al., 2007b] Mery, B., Bassac, C., and Retoré, C. (2007b). A montague-based model of generative lexical semantics. In Muskens, R., editor, *New Directions in Type Theoretic Grammars*. ESSLLI, Foundation of Logic, Language and Information.
- [Montague, 1974] Montague, R. (1974). The proper treatment of quantification in ordinary English. In Thomson, R. H., editor, *Formal Philosophy*, pages 188–221. Yale University Press, New Haven Connecticut.
- [Nirenburg et al., 1995] Nirenburg, S., Raskin, V., and Onyshkevych, B. (1995). Apologiae Ontologiae. In *Memoranda in Computer and Cognitive Science MCCS-95-281*.
- [Nunberg, 1993] Nunberg, G. (1993). Transfers of meaning. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 191–192, Morristown, NJ, USA. Association for Computational Linguistics.
- [Pinkal and Kolhase, 2000] Pinkal, M. and Kolhase, M. (2000). Feature Logic for Dotted Types : A Formalism for Complex Word Meanings. In *ACL 2000*.
- [Pustejovsky, 1995] Pustejovsky, J. (1995). *The Generative Lexicon*. MIT Press.
- [Pustejovsky, 1998] Pustejovsky, J. (1998). Generativity and Explanation in Semantics : a reply to Fodor and Lepore. *Linguistic Inquiry*, 29 :289–311.
- [Pustejovsky, 2001] Pustejovsky, J. (2001). Type construction and the logic of concepts.
- [Pustejovsky, 2006a] Pustejovsky, J. (2006a). Models of Lexical Meaning. <http://www.cs.brandeis.edu/~jamesp/projects/models.html>.
- [Pustejovsky, 2006b] Pustejovsky, J. (2006b). Type Theory and Lexical Decomposition. Semantics Archive.
- [Pustejovsky and Asher, 2000] Pustejovsky, J. and Asher, N. (2000). The Metaphysics of Words in Context. *Objectual attitudes, Linguistics and Philosophy*, 23 :141–183.

- [Pustejovsky and Boguraev, 1996] Pustejovsky, J. and Boguraev, B. (1996). Introduction : Lexical Semantics in Context. In Pustejovsky, J. and Boguraev, B., editors, *Lexical Semantics : The Problem of Polysemy*. Oxford University Press.
- [Pustejovsky and Bouillon, 1996] Pustejovsky, J. and Bouillon, P. (1996). Aspectual Coercion and Logical Polysemy. In Pustejovsky, J. and Boguraev, B., editors, *Lexical Semantics : The Problem of Polysemy*. Oxford University Press.
- [Saba, 2007] Saba, W. S. (2007). Compositional Semantics Grounded in Commonsense Metaphysics. In *EPIA 2007*.
- [Searle, 1979] Searle, J. (1979). *Expression and Meaning*. Cambridge University Press.
- [Smith, 2003] Smith, B. (2003). Ontology and information systems. In Floridi, L., editor, *Blackwell Guide to the Philosophy of Computing and Information*, pages 155–166. Blackwell, Oxford.
- [Vanier et al., 2006] Vanier, J., Bassac, C., Henry, P., Marlet, R., and Retoré, C. (2006). Toward a knowledge representation model dedicated to the semantic analysis of the sentence. Technical report, INRIA.