Université Paris VIII

Laboratoire de Recherche en Informatique Avancée **Equipe de Cognition Artificielle**

Expression langagière ambiguë et modélisation cognitive symbolique

Un modèle informatique du traitement de la polysémie d'usage

THÈSE

présentée et soutenue publiquement le 11 juin 1999

pour obtenir le grade de

Docteur de l'Université Paris VIII
Discipline Informatique – spécialité Intelligence Artificielle

présentée par

Sylvain Surcin

Président :	R. Ghiglione, professeur, Université Paris VIII
Rapporteurs :	C. Choppy, professeur, Université Paris XIII A. Nicolle, professeur, Université de Caen B. Victorri, directeur de recherches, Elsap - CNRS
Examinateur :	G. Vignaux, directeur de recherches, Inalf - CNRS
Directeur :	V. Prince, professeur, Université Paris VIII

La paix de l'esprit
Le silence du Nirvana
Tout est vanité

Remerciements

Je tiens tout d'abord à remercier mes rapporteurs, Anne Nicolle (Professeur à l'Université de Caen), Bernard Victorri (Directeur de Recherches à l'ELSAP), à qui je voudrqis exprimer ma gratitude pour sa patience et sa compréhension à mon égard, et Christine Choppy (Professeur à l'Université de Paris 13) d'avoir accepté de corriger et de juger mon travail. Ma reconnaissance va tout particulièrement à Christine Choppy, qui m'a fait découvrir et aimer la recherche.

Je remercie également Georges Vignaux (Directeur de Recherche à l'INALF), qui a accepté de faire partie de mon jury, pour ses encouragements et sa confiance lorsque j'ai commencé cette thèse.

Merci aussi à Gérard Sabah, du LIMSI, pour m'avoir si gentiment proposé de lire mon manuscrit et de me livrer ses commentaires à un moment où j'en avais vraiment besoin.

Merci infiniment à Robin pour son soutien sans faille durant toutes ces années.

Merci aussi à mes parents pour leur foi en moi et leur indéfectible affection.

Merci aux chercheurs et thésards du LIMSI, ainsi qu'à mes amis hors du milieu de la recherche, pour m'avoir écouté et conseillé à maintes reprises. Merci aussi à mes collègues du laboratoire Communication Homme-Machine de Thomson-CSF/LCR qui m'ont accueilli, soutenu, et aidé durant les derniers mois de cette thèse.

Enfin merci à tous les autres que je n'ai pas cité et qui se sentiraient lésés.

Table des matières

Liste des figures x

1 Introduction 1

2 Modèles linguistiques 5
2.1 Introduction . . . 5
2.2 De l'ambiguïté à la polysémie . . . 6
2.2.1 Le modèle "homonymie" . . . 8
2.2.2 Le modèle de l'indétermination . . . 11
2.2.3 Le modèle "polysémie" . . . 14
2.2.4 Conclusion sur le modèles linguistiques de l'ambiguïté . . . 16
2.3 Les polysémies . . . 16
2.3.1 Caractérisation de la polysémie . . . 19
2.3.1.1 Les énoncés . . . 20
2.3.1.2 Critères de comparaison des énoncés . . . 20
2.3.1.3 Sens des énoncés . . . 22
2.3.2 Polysémie fonctionnelle . . . 24
2.3.3 Polysémie d'acception . . . 25
2.3.4 Polysémie d'usage . . . 28
2.4 Polysémie d'usage et ambiguïté . . . 31
2.4.1 Occurrences de polysémie d'usage . . . 31
2.4.1.1 Transferts d'usage . . . 31
2.4.1.2 Cumuls d'emplois délibérés . . . 33
2.4.1.3 Mots jokers . . . 35
2.4.1.4 Jeux de langage . . . 36
2.4.2 Caractérisation . . . 38
2.4.3 Faut-il résoudre l'ambiguïté ? . . . 38
2.5 Cadres sémantiques théoriques . . . 42
2.5.1 Paradigmes sémantiques classiques . . . 42
2.5.1.1 La sémantique référentielle . . . 43
2.5.1.2 La sémantique inférentielle . . . 47
2.5.2 La sémantique différentielle . . . 49
2.5.2.1 La valeur . . . 49
2.5.2.2 Les éléments constitutifs de la sémantique différentielle . . . 50
2.5.2.3 Signification et interprétation . . . 53
2.5.2.4 Isotopie . . . 54
2.6 Les limites de la dénotation . . . 55
2.7 Connotation, profondeur et continuité . . . 58
2.7.1 Le rôle de la connotation . . . 58

2.7.2 La profondeur sémantique . . . 59
2.7.3 La continuité sémantique . . . 60
2.8 Récapitulation . . . 62

3 Modèles computationnels 65
3.1 Introduction . . . 65
3.2 Le traitement automatique des langues naturelles et l'ambiguïté . . . 66
3.2.1 Hypothèses et architectures . . . 66
3.2.2 Statut du lexique en traitement automatique des langues naturelles . . . 68
3.3 Quelques modèles computationnels classiques . . . 74
3.3.1 Le Word Expert Parser . . . 74
3.3.2 Les Polaroid Words . . . 74
3.3.3 Le système CARAMEL . . . 76
3.4 Un système de traitement de la polysémie d'acception . . . 77
3.5 Un système connexioniste de traitement de la polysémie . . . 79
3.5.1 Le modèle morphodynamique . . . 79
3.5.2 La mise en œuvre connexioniste . . . 80
3.5.3 Critique . . . 81
3.6 Lexiques dynamiques . . . 81
3.6.1 Le GENERATIVE LEXICON . . . 83
3.6.1.1 Les qualiae . . . 83
3.6.1.2 Coercion de type . . . 85
3.6.1.3 Niveaux d'abstraction . . . 86
3.6.1.4 Limitations . . . 87
3.6.1.5 Récapitulation . . . 91
3.6.2 Le modèle EDGAR . . . 91
3.6.2.1 Originalités d'EDGAR . . . 92
3.6.2.2 Composante statique . . . 94
3.6.2.3 Composante dynamique . . . 94
3.6.2.4 Récapitulation . . . 96
3.7 Récapitulation . . . 98

4 Le modèle PELEAS 99
4.1 Introduction . . . 99
4.2 Experience préliminaire . . . 99
4.2.1 L'interprétation comme traduction . . . 100
4.2.2 Observations . . . 101
4.2.2.1 Recueil des données . . . 101
4.2.2.2 Description synthétique . . . 102
4.2.2.3 Interprétation théorique . . . 104
4.3 PELEAS . . . 106
4.3.1 Entrées figurant dans le lexique . . . 106
4.3.2 Structure descriptive d'une entrée lexicale . . . 107
4.3.2.1 Taux d'activité . . . 107
4.3.2.2 Potentiel sémantique . . . 109
4.3.2.3 Relations entre les sommets . . . 113
4.3.2.4 Conditions contextuelles . . . 114
4.3.3 Propagation de l'activité . . . 116
4.3.3.1 Règles de propagation par défaut . . . 116
4.3.3.2 Algorithme de propagation . . . 117

4.3.4 Estimation des interprétations . . . 120
4.4 Construction d'un lexique . . . 120
4.4.1 Pôles sémantiques . . . 121
4.4.2 Fonctionnalités souhaitées . . . 121
4.4.3 Modélisation . . . 122
4.5 Récapitulation . . . 126
4.6 Expérimentation du modèle et de sa mise en œuvre . . . 127
4.6.1 Objectifs de l'expérimentation . . . 129
4.6.2 Caractéristiques et mesures statistiques . . . 129
4.6.2.1 Acquisition statistique des données . . . 129
4.6.2.2 Description statique du lexique . . . 130
4.6.2.3 Principe du test . . . 131
4.6.2.4 Mesures dynamiques . . . 131
4.6.2.5 Test de corrélation . . . 132
4.6.2.6 Test de pertinence . . . 133
4.6.3 Résultats de l'expérimentation . . . 134

5 Mise en œuvre **137**
5.1 Introduction . . . 137
5.1.1 Choix concernant la mise en œuvre . . . 137
5.1.1.1 Multitâche, modularité et communication . . . 138
5.1.1.2 Choix d'une plate-forme de développement . . . 139
5.1.2 Architecture . . . 140
5.1.2.1 Architecture logicielle . . . 141
5.1.2.2 Architecture fonctionnelle . . . 142
5.1.2.3 Architecture structurelle . . . 143
5.1.3 Présentation des logiciels . . . 144
5.2 Le modèle objet LIGHTPELEAS . . . 145
5.2.1 Les coclasses de LIGHTPELEAS . . . 145
5.2.1.1 La coclasse Engine . . . 148
5.2.1.2 La coclasse Entries . . . 149
5.2.1.3 La coclasse Entry . . . 149
5.2.1.4 La coclasse Domains . . . 150
5.2.1.5 La coclasse Domain . . . 151
5.2.1.6 La coclasse Notions . . . 152
5.2.1.7 La coclasse Notion . . . 152
5.2.1.8 La coclasse CViews . . . 153
5.2.1.9 La coclasse CView . . . 153
5.2.1.10 La coclasse Features . . . 154
5.2.1.11 La coclasse Feature . . . 155
5.2.1.12 La coclasse Constraints . . . 155
5.2.1.13 La coclasse Constraint . . . 156
5.2.1.14 La coclasse Libraries . . . 156
5.2.1.15 La coclasse Library . . . 157
5.2.1.16 La coclasse Predicates . . . 157
5.2.1.17 La coclasse Predicate . . . 158
5.2.1.18 La coclasse Rules . . . 158
5.2.1.19 La coclasse Rule . . . 158
5.2.1.20 La coclasse Condition . . . 159

5.2.1.21 La coclasse Stack . . . 160
5.2.1.22 La coclasse Activations . . . 161
5.2.1.23 La coclasse Activation . . . 161
5.2.1.24 La coclasse Guesses . . . 161
5.2.1.25 La coclasse Guess . . . 161
5.2.2 Format de stockage des entrées lexicales . . . 162
5.2.2.1 Organisation des enregistrements . . . 162
5.2.2.2 Le format PDL . . . 162
5.3 L'éditeur graphique MELISANDE . . . 164
5.3.1 Gestion des fichiers . . . 164
5.3.2 Edition d'une entrée lexicale . . . 165
5.3.2.1 Commandes d'édition . . . 166
5.3.2.2 Contraintes sémantiques . . . 167
5.3.2.3 Règles contextuelles . . . 168
5.4 Développement d'une bibliothèque de prédicats . . . 170
5.4.1 Format des prédicats . . . 170
5.4.2 Mise en œuvre des prédicats . . . 171
5.4.3 Spécificités d'une DLL "bibliothèque de prédicats" . . . 172
5.4.4 Format des prédicats dans la DLL . . . 173
5.5 L'analyseur de corpus BARD . . . 173
5.5.1 Hypothèses utilisées . . . 173
5.5.1.1 Sélection des contextes . . . 173
5.5.1.2 Lemmatisation . . . 174
5.5.1.3 Calcul du score d'écart réduit . . . 175
5.5.2 Le prototype BARD . . . 175
5.6 L'analyseur morphologique AMORPH . . . 180
5.6.1 Spécifications du composant AMORPH . . . 180
5.6.1.1 Interface IAnaLex . . . 181
5.6.1.2 Interface IFlexedForms . . . 182
5.6.1.3 Interface IFlexedForm . . . 182
5.6.2 Remarques d'ordre technique . . . 183
5.7 Récapitulation . . . 183

6 Bilan et perspectives **185**
6.1 Récapitulation . . . 185
6.2 Apports du/au génie logiciel . . . 188
6.3 Perspectives . . . 189

Bibliographie **191**

A Corpus et jeux d'essais **199**
A.1 Structures descriptives . . . 199
A.2 Corpus de test . . . 237
A.3 Résultats des jeux d'essais . . . 244

B Introduction à ActiveX **333**
B.1 Le standard (D)COM . . . 333
B.1.1 Un peu de terminologie . . . 335
B.1.2 COM et les interfaces . . . 336
B.1.2.1 Les classes et leurs relations dans les langages traditionnels . . . 336

B.1.2.2 Les classes de COM . . . 338
B.1.2.3 L'interface IUnknown . . . 340
B.1.2.4 Quelques règles d'or . . . 342
B.1.3 Création d'une instance . . . 343
B.1.3.1 Les GUID . . . 343
B.1.3.2 Les Class Factories . . . 344
B.1.4 Utilisation d'un objet COM . . . 345
B.1.5 Les relations entre coclasses . . . 346
B.1.5.1 Inclusion d'objet et délégation d'interface . . . 347
B.1.5.2 Agrégation . . . 347
B.2 Automation . . . 349
B.2.1 L'interface IDispatch . . . 349
B.2.1.1 Les identificateurs DispID . . . 350
B.2.1.2 Les bibliothèques de types . . . 350
B.2.2 Les interfaces duales . . . 351
B.2.3 Les types standards d'Automation . . . 351
B.2.4 Les collections au sens de Visual Basic . . . 352
B.3 La partie immergée de l'iceberg . . . 353

C Fonctionnement des DLL 355
C.1 Principe des bibliothèques dynamiques . . . 355
C.2 Mécanisme d'édition de liens dynamique . . . 356

D Spécifications formelles 359
D.1 Spécifications des coclasses de LIGHTPELEAS . . . 359
D.1.1 Spécification de la coclasse Engine . . . 359
D.1.2 Spécification de la coclasse Entries . . . 360
D.1.3 Spécification de la coclasse Entry . . . 362
D.1.4 Spécification de la coclasse Domains . . . 364
D.1.5 Spécification de la coclasse Domain . . . 365
D.1.6 Spécification du type pelConstraint . . . 367
D.1.7 Spécification de la coclasse Constraint . . . 367
D.1.8 Spécification de la coclasse Condition . . . 368
D.1.9 Spécification de la coclasse Stack . . . 370
D.2 Spécifications des coclasses d'AMORPH . . . 372
D.2.1 Spécification de la coclasse AnaLex . . . 372
D.2.2 Spécification de la coclasse FlexedForms . . . 374
D.2.3 Spécification de la coclasse FlexedForm . . . 374

E Base de données morphologique 377
E.1 Schémas de la base de données . . . 377
E.1.1 Schéma de la table des mots . . . 378
E.1.2 Schéma de la table des conjugaisons . . . 378
E.1.3 Schéma de la table des terminaisons . . . 379
E.1.4 Schémas supplémentaires . . . 381
E.2 Algorithme d'analyse morphologique . . . 381
E.3 Requêtes d'analyse morphologique . . . 382

F Représentation hypergraphique des entrées lexicales **385**
F.1 Motivation de l'équivalence graphe / hypergraphe 385
F.2 Définition d'une bijection 386
F.2.1 Construction de la bijection pour les graphes pyramidaux de profondeur 1 386
F.2.2 Extension de la bijection de la profondeur n à la profondeur $n+1$ 387
F.2.3 Conclusion 390

Liste des figures

2.1 Alternatives d'interprétation pour *La chute de la maison Usher* . . . 28
2.2 Variations paradigmatiques pour *mère* et *fille* . . . 32
2.3 Variations paradigmatiques pour *roi* . . . 32
2.4 Triangle sémiotique . . . 43
2.5 Semi-treillis conceptuel pour *to eat* . . . 61

3.1 Réseau sémantique pour *buy* . . . 70
3.2 Réseau de règles WEP pour *deep* (tiré de [1]) . . . 75
3.3 Arborescence d'acceptions pour *garçon* (tiré de [16]) . . . 79
3.4 Fragment d'une hiérarchie de types (extrait de [23]) . . . 84

4.1 Comparaison entre interprétation et traduction . . . 101
4.2 Processus de traduction . . . 102
4.3 Repérage d'un acte énonciatif par son contexte . . . 109
4.4 Exemple de graphe pyramidal de hauteur 3 . . . 111
4.5 Sous-graphe vide . . . 113
4.6 Sommet fantôme . . . 113
4.7 Sous-classement d'un trait sémantique . . . 124
4.8 Spécialisation d'un trait sémantique . . . 125

5.1 Architecture logicielle . . . 141
5.2 Flot général d'information autour de LIGHTPELEAS . . . 143
5.3 Flot d'information à l'intérieur de LIGHTPELEAS . . . 144
5.4 Le modèle objets de LIGHTPELEAS . . . 147
5.5 Echanges de données entre AMORPH et Access . . . 181

B.1 Coopération de composants grâce à COM et (D)COM . . . 335
B.2 Relations entre objets . . . 337
B.3 Relation d'héritage . . . 338
B.4 Relations entre le système d'exploitation et les composants ActiveX . . . 339
B.5 Normalisation de l'accès à un objet COM . . . 341
B.6 Objet COM avec plusieurs interfaces . . . 342
B.7 Représentation d'un objet COM . . . 346
B.8 Exemple de délégation . . . 347
B.9 Exemple d'agrégation . . . 348

E.1 Dictionnaire de la table *MOTS* . . . 379
E.2 Dictionnaire de la table *CONJUGAISONS* . . . 380
E.3 Dictionnaire de la table *TERMINAISONS* . . . 380
E.4 Dictionnaires des tables *GROUPES* et *TEMPS* . . . 381

F.1 Définition graphique du foncteur K 388

Chapitre 1

Introduction

> « Ce qui importe le plus, c'est la recherche elle-même. Elle est plus importante que les chercheurs. La conscience doit rêver ; il lui faut un territoire pour ses rêves et, rêvant, elle doit invoquer des rêves toujours nouveaux. »
>
> **Frank Herbert**, *Destination Vide*

L'étude du langage est longtemps restée le domaine réservé des philosophes et des grammairiens, et ce n'est qu'au début de ce siècle que les mathématiciens ont formé le projet de faire rentrer la description du langage à l'intérieur du cadre des formalismes mathématiques. Ce projet a tout naturellement trouvé son "laboratoire" dans l'informatique : le traitement automatique des langues naturelles. Projet ambitieux et optimiste, puisque l'on pouvait lire dans l'*Encyclopedia Universalis* en 1969, à la rubrique Intelligence Artificielle que « les progrès de l'Intelligence Artificielle dans le domaine du langage ont été impressionnants ces dernières années, et les vingt années à venir verront certainement l'avènement d'ordinateurs avec qui converser sera aussi facile qu'avec un être humain. »

Malheureusement, cette prédiction ne s'est pas réalisée. Au contraire, plus les informaticiens explorent le traitement automatique des langues naturelles, et plus nombreux apparaissent les problèmes apparemment insurmontables qui viennent briser la vision primitive et limpide qu'ils avaient du langage comme empilement de systèmes successifs et autonomes, mettant en œuvre des unités toujours plus vastes à mesure que l'on s'élève vers des niveaux supérieurs, et gouvernés par l'universelle règle de compositionnalité : chaque unité d'un niveau donné se réduit à la composition d'unités du niveau inférieur. Or, plus personne dans le domaine du traitement automatique des langues naturelles, maintenant, n'envisagerait de construire un correcteur orthographique, par exemple, sans un minimum de recours à la syntaxe. De même, aucune analyse syntaxique ne peut s'affranchir de la sémantique (l'étude du sens), qui elle-même subit l'influence de phénomènes que l'on qualifie de "pragmatiques". Les influences montent et descendent à travers les différents niveaux d'analyse de la langue, qui n'ont plus rien d'étanche.

On peut certes minimiser ces influences dérangeantes pour un large ensemble de phrases et de textes "normalisés", mais l'analyse de textes sans contraintes fait apparaître un cauchemar pour les informaticiens : les ambiguïtés. Une ambiguïté, qu'elle soit syntaxique, sémantique, pragmatique, etc. met en échec un programme d'analyse de la langue en l'empêchant de produire une représentation correcte et complète du fragment de texte analysé, qui soit traitable en machine.

Dans cet ouvrage, nous nous intéressons plus précisément à une catégorie particulière d'ambiguïtés : les *ambiguïtés lexicales*. Le problème est le suivant : au cours de l'analyse d'un texte, il faut assigner un sens à chacun des mots du texte dans le but de construire une représentation du sens de ce texte dans sa globalité. Une ambiguïté lexicale survient sur un mot donné lorsque le programme d'analyse est incapable de retrouver le sens de ce mot, et ceci peut mettre en échec tout le processus d'analyse ultérieur.

Ce type d'ambiguïtés a été intensivement étudié en informatique et en linguistique, et de nombreuses techniques ont été inventées pour permettre aux programmes d'analyse de la langue de se tirer de ces mauvais pas.

Néanmoins, en repartant de l'observation des ambiguïtés lexicales, nous mettrons ici l'accent sur un certain phénomène éminemment producteur d'ambiguïtés : la *polysémie*, c'est-à-dire le cas où un même mot a plusieurs sens possibles. L'objectif classique des recherches en traitement automatique des langues naturelles concernant la polysémie est de trouver un moyen (programmable) de sélectionner parmi tous ces sens, lequel correspond effectivement au mot polysémique dans le contexte où il a été rencontré.

De là, nous montrerons qu'il existe un type particulier de polysémie, la *polysémie d'usage*, qui n'a été que très peu étudié. Nous montrerons également qu'il s'agit d'un phénomène beaucoup plus répandu qu'on ne pourrait le penser, notamment dans le langage quotidien, journalistique et poétique. La polysémie d'usage concerne un large ensemble de mots, apparemment banals, à qui des règles d'usage enracinées dans la culture liée à la pratique d'une langue permettent d'attribuer des sens nombreux, difficiles à séparer les uns des autres, et surtout, qui peuvent coexister en même temps. On se retrouve alors dans une situation doublement difficile : d'une part on ne dispose plus de règles catégoriques et précises (donc programmables) permettant de trier les différents sens possibles, mais en plus la stratégie habituelle consistant à sélectionner un sens et à rejeter tous les autres se révèle au mieux inadéquate et au pire catastrophique, puisque plusieurs sens peuvent (et souvent doivent) être acceptables simultanément.

Notre travail va donc être tout d'abord de chercher à caractériser au maximum les phénomènes de polysémie d'usage. Puis, en interrogeant tour à tour la linguistique et l'informatique, nous chercherons à déterminer des cadres théorique et pratique au sein desquels nous pourrons développer un modèle de la polysémie d'usage qui puisse donner lieu à la réalisation d'une application informatique.

La principale propriété dont nous voudrions doter ce modèle est la *robustesse*, c'est-à-dire que nous souhaitons qu'il soit capable de fournir une interprétation pour n'importe quelle situation de polysémie d'usage. Nous tolérerons donc des baisses de qualité dans la réponse (i.e. des réponses un peu "brouillées") pour des situations franchement dégradées, c'est-à-dire lorsque les informations disponibles sont très peu nombreuses ou très floues.

Ce modèle, PELEAS, donnera lieu à une mise en œuvre informatique qui pourra agir comme un module spécialisé au sein d'une architecture plus large de traitement de la langue. Il devra fournir aux autres modules, dans le meilleur des cas l'ensemble de tous les sens simultanément présents pour un mot affecté de polysémie d'usage, et au pire, c'est-à-dire en cas d'information insuffisante, il devra "limiter les dégâts" en proposant un ensemble (forcément plus large, donc bruité) de sens probables.

Nous verrons en dernier lieu comment ce module peut effectivement s'insérer dans une architecture de traitement automatique des langues naturelles, et quels développements ultérieurs on peut envisager du modèle PELEAS, notamment pour l'aide à la traduction, et pour la représentation des connaissances en machine.

Plan

Chapitre 2 En partant de l'observation des ambiguïtés lexicales, qui sont des perturbations gênantes pour les processus de traitement automatique des langues naturelles, on va montrer qu'il existe trois cadres théoriques pour les expliquer : comme résultant d'une *homonymie*, d'une *indétermination* ou d'une *polysémie* (cf. §2.2). Après avoir justifié l'abandon des modèles liés à l'homonymie et à l'indétermination, jugés trop réducteurs, on conservera le cadre de la polysémie.

De là, on constatera qu'il existe plusieurs types de polysémie (cf. §2.3). On essaiera tout d'abord de caractériser ce qu'il y a de commun à toutes les formes de polysémie, puis on les détaillera en fonction de leur complexité croissante : *polysémie fonctionnelle*, *polysémie d'acception* et *polysémie d'usage*. Nous introduisons le concept de polysémie d'usage, qui n'a jamais été défini en tant qu'objet d'étude autonome, ni en linguistique, ni en informatique.

Nous nous attacherons donc à combler cette lacune en faisant de la polysémie d'usage le thème central de cette thèse. Nous détaillerons tout d'abord ses relations avec les ambiguïtés lexicales (cf. §2.4) en dégageant une typologie des ambiguïtés qu'elle engendre et en les caractérisant. De ces observations, nous serons amenés à conclure que, dans le cas de la polysémie d'usage, la stratégie consistant à *résoudre les ambiguïtés* n'est pas pertinente.

Pour comprendre cette assertion, nous interrogerons tout à tour les trois grands modèles théoriques de la sémantique fournis par la linguistique (cf. §2.5) : les sémantiques *référentielle*, *inférentielle* et *différentielle*. Cette observation nous amènera à rejeter les deux premiers modèles comme inadéquats pour décrire la polysémie d'usage.

Ayant adopté le cadre de la sémantique différentielle, nous aborderons enfin trois notions que nous jugeons fondamentales dans la polysémie d'usage : la *connotation*, la *profondeur* et la *continuité* (cf. §2.7).

Chapitre 3 Nous nous pencherons dans ce chapitre sur les modèles computationnels proposés par le traitement automatique des langues naturelles pour traiter les problèmes d'ambiguïté lexicale. Pour cela, nous commencerons par prendre la mesure des conceptions *a priori* qui sont à la base des modèles computationnels (cf. §3.2).

Pour illustrer notre jugement, à la fois sur ces conceptions et sur les modèles sémantiques classiques du chapitre précédent, nous présenterons quelques modèles computationnels classiques de traitement des ambiguïtés, ainsi que leurs limitations (cf. §3.3).

Puis nous aborderons des modèles plus novateurs et plus marginaux : un système de traitement de la polysémie d'acception (cf. §3.4), un système connexioniste de traitement de la polysémie au sens large (cf. §3.5), et enfin nous aborderons les modèles que nous jugeons les plus prometteurs pour notre propos : les *lexiques dynamiques* (cf. §3.6).

A cette occasion, nous étudierons deux représentants de ces lexiques : le GENERATIVE LEXICON, ainsi que le modèle EDGAR. Nous verrons que ce dernier modèle, conçu pour traiter la polysémie des mots courants, est aisément adaptable (au prix d'un certain nombre d'extensions) au traitement de la polysémie d'usage, notamment en étendant la notion de "mot courant".

Chapitre 4 Pour construire notre propre modèle PELEAS, nous avons tout d'abord procédé à une expérience préliminaire afin de glâner des informations pratiques supplémentaires permettant de compléter le cadre purement théorique de la sémantique différentielle (cf. §4.2).

Le modèle PELEAS, décrit au §4.3, fusionne les principes du modèle EDGAR et les informations issues de notre expérience. Il s'agit d'un lexique dynamique qui structure ses données de manière hiérarchique et dont le processus d'interprétation est basé sur un mécanisme de propagation d'activité par défaut avec relaxation de contraintes sémantiques. L'algorithme d'interprétation que nous avons mis au point est agréablement efficace car il est linéaire en temps au pire.

Puis nous nous interrogerons (cf. §4.4) sur la possibilité d'automatiser la construction d'un lexique de structures adaptées à PELEAS. Nous verrons quelles modalités un tel logiciel devrait présenter, et nous proposerons un ébauche de modélisation pour ce logiciel sous forme de spécifications.

Finalement, nous procéderons à une première évaluation du modèle PELEAS sur des cas concrets (cf. §4.6). Il s'agit d'un test mené pour un lexique de 21 mots polysémiques sur un jeu de 110 phrases présentant délibérément des difficultés d'interprétation rédhibitoires pour un analyseur sémantique classique (nous justifierons également la "petitesse" de ce corpus de test). Cette évaluation nous permettra de vérifier la *robustesse* du modèle PELEAS, ainsi que sa *puissance* et l'*efficacité* de sa mise en œuvre, en plus de diverses autres questions d'ordre pratique.

Chapitre 5 Nous présenterons ici la mise en œuvre informatique que nous avons réalisée. Nous commencerons par exposer les divers choix que nous avons fait quand à cette réalisation (cf. §5.1) donnant lieu à des spécifications architecturelles. Puis, nous présenterons dans l'ordre le moteur d'inférence LIGHTPELEAS (cf. §5.2), l'éditeur graphique de lexiques MELISANDE (cf. §5.3), le format des bibliothèques de prédicats utilisés par les conditions contextuelles (cf. §5.4), l'analyseur statistique de corpus BARD pour l'acquisition de données (cf. §5.5) et enfin l'analyseur morphologique AMORPH dont nous nous servons à des fins de lemmatisation (§5.6).

Chapitre 6 Nous terminerons ce travail par un bilan récapitulant notre travail et les conclusions auxquelles nous avons abouti, et nous introduirons un certain nombre de perspectives de recherche et d'applications qui découlent naturellement de ce travail.

Chapitre 2

Modèles linguistiques

> «Tout symbole comporte des prémisses cachées. Tout mot véhicule des hypothèses tacites enfouies dans l'histoire de la langue et dans les expériences qui ont conditionné ceux qui la parlent. Si vous arrachez aux mots ces sens cachés, un flot de compréhension nouvelle se déversera dans votre conscience.»
>
> **Frank Herbert**, *Destination Vide*

2.1 Introduction

Dans ce chapitre, nous abordons le problème de l'ambiguïté dans les langues naturelles d'un point de vue d'informaticien, c'est-à-dire sans être rompu aux subtilités de la linguistique. Notre objectif principal est de choisir une théorie linguistique qui décrive suffisamment les faits que nous observerons, et qui soit en outre raisonnablement informatisable.

Nous concentrerons plus particulièrement notre étude sur les formes d'ambiguïtés lexicales complexes. Eu égard aux deux critères donnés au paragraphe précédent, notre choix d'une théorie linguistique se portera sur celle de la polysémie.

Cependant, le terme *polysémie* recouvre un vaste ensemble de phénomènes distincts. Nous chercherons donc, à partir des travaux de linguistes, à caractériser ces phénomènes, notamment afin de mieux nous repérer dans le continuum qui s'étend de l'homonymie à la polysémie. Ces deux aspects de l'ambiguïté possèdent en effet une frontière difficilement discernable. Pour cela, nous passerons par une formalisation des notions fondamentales d'*énoncé* et de *sens d'un énoncé*. De là, nous pourrons procéder à une catégorisation des différents sous-types de polysémie.

A partir de ce point, nous nous concentrerons plus particulièrement sur la catégorie de polysémie qui a été la moins étudiée : la *polysémie d'usage*. Nous dresserons pour cela une liste systématique des différentes occurrences de polysémie d'usage rencontrées dans des exemples issus de textes très généraux (presse écrite, livres, médias audiovisuels, extraits de conversations, etc.). Cela nous permettra de déduire les principes communs à toutes ces formes de la polysémie d'usage, ainsi que leurs caractéristiques spécifiques.

Nous chercherons ensuite parmi les différents cadres proposés par la linguistique une théorie sémantique adaptée à la description de ces phénomènes. En particulier, nous verrons que les théories classique-

ment utilisées en traitement automatique des langues naturelles (sémantique référentielle et sémantique inférentielle) posent de sérieux problèmes. Cela nous conduira à nous intéresser à la sémantique différentielle.

Puis, un cadre théorique ayant été choisi, nous aborderons les notions linguistiques de connotation, de profondeur et de continuité qui sont au cœur de notre problème.

2.2 De l'ambiguïté à la polysémie

La langue quotidienne présente un nombre élevé d'occurrences de polysémie (i.e. de mots ou de groupes de mots ayant plusieurs sens). Cette polysémie ne pose généralement pas de problème aux locuteurs humains. Ils n'ont souvent même pas conscience d'avoir à choisir entre plusieurs interprétations. Or, si cette inconscience des traitements est souhaitable chez l'humain, il n'en va plus de même pour des locuteurs artificiels. C'est pourquoi une compréhension en profondeur de ce phénomène s'avère nécessaire pour son informatisation.

Notamment, le domaine du traitement automatique des langues naturelles a pour objectif de construire des systèmes permettant de passer du matériau linguistique à des représentations informatiques calculables, et inversement. Il faut donc trouver des moyens de traduire des représentations appartenant à un système qui n'est pas encore entièrement exploré (la langue), obéissant à des systèmes de règles peut-être ouverts, utilisant une logique fortement multivaluée et d'un ordre non déterminé mais certainement élevé[1] à un système déterministe, basé sur des ensembles de primitives et de règles finis, à logique binaire et d'ordre 1. Au cours de cette tâche, des difficultés importantes viennent souvent gêner les processus permettant de passer du texte aux représentations informatiques. Il s'agit en particulier des ambiguïtés qui apparaissent dans la langue avec une fréquence choquante. Un réflexe légitime pour l'informaticien est de se tourner vers les linguistes pour exiger d'eux des systèmes de règles d'analyse qu'on puisse mettre en œuvre sur machine afin de se débarrasser de ces ambiguïtés. Or la linguistique ne fournit pas de description exhaustive des phénomènes d'ambiguïté. C'est pourquoi nous allons nous permettre, en tant qu'informaticien, une incursion dans le domaine de la linguistique. Nos objectifs ne sont pas les mêmes que ceux des linguistes : notre critère est avant tout que "ça tourne", en sacrifiant éventuellement l'exactitude théorique.

Nous allons voir ici comment l'analyse linguistique des ambiguïtés des langues naturelles fait émerger la notion de polysémie comme cadre de réflexion. Pour cela, nous nous appuyons sur une synthèse des approches linguistiques présentée par Le Goffic dans [64].

Différentes interprétations sont possibles en fonction des *a priori*

En linguistique, analyser l'ambiguïté requière un parti pris, des postulats obligatoires et antérieurs à la démarche d'analyse. Pour illustrer ces orientations, nous allons partir d'un exemple fourni par Le Goffic [64] :

Paul a apprécié la critique de Jakobson.

Cette phrase s'interprète de différentes façons. Un premier type d'analyse possible consiste à supposer que cet énoncé n'a que l'apparence d'un énoncé unique, et qu'il s'agit en fait de deux énoncés

[1] cf. l'argumentation de Prince dans [90].

différents, soit deux structures d'analyse. Ainsi, on peut interpréter cet énoncé soit comme "Paul a apprécié que Jakobson critique X", soit comme "Paul a apprécié que X critique Jakobson"[1]. Ces deux structures ont accidentellement la même forme d'expression ; l'énoncé correspondant n'a que l'apparence de l'unicité. A charge pour le récepteur de l'énoncé de retrouver laquelle de ces deux structures correspond réellement à l'intention de l'émetteur. C'est l'approche qui est défendue par Chomsky dans [18] : l'ambiguïté est une forme d'*homonymie*, i.e. il y a coïncidence de forme fortuite entre des éléments linguistiques qui n'ont rien à voir entre eux.

On peut également considérer qu'il ne s'agit pas d'un énoncé ambigu, mais indéterminé : il n'y a qu'une seule structure, correspondant à une seule "valeur linguistique" indéterminée : *Paul a apprécié la critique qui* a à voir *avec Jakobson*. Cette indétermination ne perdure que tant que l'énoncé est coupé de ses conditions d'énonciation. A de rares exceptions près, le contexte de la communication spécifie les informations nécessaires pour lever l'indétermination. Dans cette approche, les dysfonctionnements de communication sont considérés comme accidentels et imputables aux locuteurs : l'émetteur n'a pas suffisamment spécifié le contexte d'élocution et/ou le récepteur n'a pas produit une analyse suffisamment pertinente. Mais ce genre d'accident est considéré comme ponctuel, et n'affecte pas la description de la langue au niveau théorique. Cette approche relève d'une tendance structuraliste radicale, soutenue, par exemple, par Weydt [134].

Ces deux positions correspondent à des attitudes extrêmes. On ne peut nier leur validité dans des exemples typiques comme *Jean n'a pas aimé l'avocat* ou *Jacques s'est cassé la jambe*. Dans le premier cas, l'analyse par homonymie s'impose puisque le mot *avocat* peut désigner un fruit ou un homme de loi. Et dans le second, l'énoncé n'est pas ambigu, mais indéterminé, puisqu'on ne sait pas s'il s'agit de la jambe gauche ou droite. Reste à déterminer, pour des cas moins tranchés, quelle importance relative donner à l'une et l'autre approche. C'est un problème délicat car les partisans de ces deux approches n'arrivent pas eux-mêmes à tracer une limite précise entre homonymie et indétermination, tout cela dans la plus stricte ignorance d'un dernier type d'analyse tout aussi valide.

Il s'agit de concilier la dualité de l'approche par homonymie et l'unicité de l'approche par indétermination. Cette troisième approche considère qu'il existe des "effets de sens" construits à partir d'une "valeur structurale fondamentale". Ainsi, dans l'énoncé que nous avons cité, le mot *de* est considéré comme ayant une fonction (ou valeur structurale) unique : la marque d'un génitif[2] indéterminé. Cette approche ne considère cependant pas que l'indétermination est levée durant la communication, mais qu'on peut formaliser théoriquement (hors contexte d'énonciation) la coexistence des deux interprétations possibles que nous avons vues dans ce que nous appellerons dorénavant l'approche "homonymie". Dans l'énoncé, l'ambiguïté concerne une structure N *de* N (i.e. deux noms séparés par la préposition *de*). L'approche abordée ici considère que la structure N *de* N correspond à deux "schémas actanciels" (cf. Tesnière [124]), c'est-à-dire à deux façons possibles de distribuer les rôles sujet et objet entre les deux noms. C'est pour cela qu'on peut fournir deux interprétations possibles : un génitif objectif (c'est Jakobson qui critique) et un génitif subjectif (c'est Jakobson qui se fait critiquer). Le Goffic appelle cette approche le modèle "polysémie".

[1] Sans compter que l'on peut aussi hésiter entre une interprétation de *critique* comme action ou comme document.

[2] Il existe entre *critique* et *Jakobson* un rapport d'attribution ou d'appartenance.

2.2.1 Le modèle “homonymie”

Plusieurs structures profondes univoques

L'idée principale du modèle “homonymie” est que, sous une même apparence linguistique, se cachent *plusieurs* énoncés distincts correspondant à autant de structures d'analyse profondes (au sens de la grammaire transformationnelle-générative[1]). On considère que ces structures sont totalement caractérisables et mutuellement exclusives. Cela signifie que, pour peu qu'on dispose des informations nécessaires, on peut décider quelle est la bonne interprétation de la séquence ambiguë et lesquelles doivent être rejetées. Dans la langue vécue (ailleurs que dans les articles de linguistique), il en résulte que l'énoncé n'est pas ambigu, puisque l'émetteur sait exactement quelle structure il a voulu exprimer. Le récepteur se retrouve, lui, face à un problème de choix : il doit décider, en fonction de ce qu'il sait, de quelle structure son interlocuteur a voulu lui transmettre.

Cela s'applique aux phénomènes d'homophonie dans le discours oral, c'est-à-dire aux énoncés différents qui se prononcent de la même façon (comme *un jour pluvieux* / *un jour plus vieux*) et aux phénomènes d'homographie dans le discours écrit, i.e. aux énoncés qui s'écrivent de la même façon sans avoir de rapport entre eux (comme *il est resté prostré devant la bière*, où *bière* peut être aussi bien une boisson qu'un cercueil). Mais cela est également étendu aux constructions syntaxiques, telles que l'exemple cité plus haut (*Paul a apprécié la critique de Jakobson*). On peut construire beaucoup d'expressions semblables basées sur l'opposition génitif objectif / génitif subjectif du groupe N *de* N (par exemple *la peur du gendarme*) ou sur les ellipses anaphoriques du groupe verbal, comme dans *Paul propose à Jean de partir* (est-ce Paul ou Jean qui part ?) : on a un premier groupe verbal *Paul propose à Jean de* X, qui est complété par le groupe verbal *partir* (la variable X est instanciée en *partir*). Or ce dernier groupe est élidé. Il devrait contenir au moins un sujet : Z *partir*, qui permettrait d'instancier la variable Z avec l'un ou l'autre des rôles du premier groupe verbal (avec *Paul* ou avec *Jean*). C'est l'impossibilité de procéder à l'unification de Z avec *Paul* ou *Jean* qui produit l'ambiguïté de l'énoncé.

Dans tous ces cas, l'idée est la même : n structures profondes[2] correspondent à une seule structure superficielle. Quelque part entre le niveau profond et le niveau de surface, il s'est passé “quelque chose” qui a rendu les choses confuses, et il faut que le récepteur de l'énoncé démêle cet écheveau pour retrouver l'intention originelle de l'émetteur.

Eclatement des représentations lexicales

Lorsque cette attitude est étendue aux ambiguïtés lexicales, comme c'est généralement le cas en traitement automatique des langues naturelles, les formes lexicales ambiguës sont considérées comme forcément homonymiques. C'est la position qu'adoptent Katz et Fodor [56] : chaque sens ou acception d'un mot est une entité lexicale distincte, qui reçoit un numéro pour la différencier des autres, comme :

— *Marché*$_1$: lieu, dans *Je vais au marché* ;

— *Marché*$_2$: activité, dans *Je fais mon marché* ;

— *Marché*$_3$: institution, dans *Pour réussir, il faut suivre les lois du marché* ;

[1] Cette école, amorcée par Chomsky [18], s'occupe d'étudier les transformations que subit l'intention sémantique de l'émetteur d'un message (ou *structure profonde*) et qui vont lui permettre d'engendrer un certain nombre d'énoncés effectivement exprimés (ou *structures de surface*)

[2] Dans le vocabulaire de la grammaire transformationnelle-générative, la structure profonde d'un énoncé est son contenu logique, prédicatif, tandis que la structure superficielle est le phénomène syntaxique observable, l'apparence de l'énoncé.

— *Marché*$_4$: acteurs économiques, dans *C'est le marché qui décidera si ce produit est bon ou pas* ;

— *Marché*$_5$: tendance économique, dans *Aujourd'hui, le marché est en hausse*.

On retrouve cette attitude de façon assez répandue en traitement automatique des langues naturelles lorsqu'il s'agit de construire des bases de concepts ou de types à apparier aux unités lexicales des énoncés analysés. C'est vrai quel que soit le mode de représentation des concepts utilisé, mais un formalisme bien connu, qui va nous servir d'illustration, est celui des treillis conceptuels de références pour les graphes conceptuels de Sowa [116]. Un treillis conceptuel est une catégorie de graphe composé de *nœuds* qui sont des concepts (plus précisément des étiquettes dénotant des concepts) et d'*arcs* qui sont des relations *est-une-sorte-de* ou *est-un* entre les concepts. Le schéma $@A \longrightarrow @B$ représente graphiquement que le concept B est une sorte de (ou une instance particulière du) concept A. On dit de A et de B qu'ils sont respectivement supérieur et inférieur l'un par rapport à l'autre. Puisqu'un treillis est un graphe, un concept particulier peut être supérieur ou inférieur à plusieurs autres concepts. La particularité d'un treillis est en outre qu'il admet un unique *maximum global* (un nœud qui n'est inférieur à aucun autre) et un unique *minimum global* (un nœud qui n'est supérieur à aucun autre). Ces concepts particuliers sont notés généralement $\top$, qui représente tout, et $\bot$ qui ne représente rien.

Avec cet outil de représentation, il semble naturel de faire "éclater" le lexique, comme Katz et Fodor, en associant des étiquettes de concepts différentes à chaque interprétation d'une unité lexicale ambiguë. Pour l'exemple du mot *marché*, on aurait cinq étiquettes correspondant à l'entrée lexicale, avec les relations suivantes dans le treillis : $@$*lieu* $\longrightarrow$ $@$*marché*$_1$, $@$*activité* $\longrightarrow$ $@$*marché*$_2$, $@$*institution* $\longrightarrow$ $@$*marché*$_3$, $@$*entité-économique* $\longrightarrow$ $@$*marché*$_4$ et $@$*variation-de-quantité* $\longrightarrow$ $@$*marché*$_5$. Afin de sélectionner la bonne étiquette conceptuelle à apparier à une occurrence de *marché* dans un énoncé donné, il faut chercher des informations dans le contexte disponible afin de déterminer quels peuvent être ses concepts supérieurs, et en déduire la bonne version du concept $@$*marché*$_X$ par induction et transitivité. Cependant, si l'on dispose d'un vrai treillis conceptuel (i.e. si un concept peut être inférieur à plus d'un seul autre concept), la situation devient inconfortable. Prince montre dans [88] qu'il est possible, voire fréquent, que plus d'un chemin puisse conduire de $\top$ jusqu'au concept cherché, en traversant éventuellement des nœuds supérieurs à plusieurs versions de ce concept. On se retrouve alors face à une explosion combinatoire du nombre de chemins qu'il est possible d'emprunter à partir de $\top$, et le problème du choix reste entier.

Transparence et opacité

Le modèle "homonymie" a été appliqué jusque dans le domaine du contenu logique des énoncés, au niveau de ce que les logiciens appellent transparence / opacité, en sémantique générative. Il s'agit d'ambiguïtés logiques portant sur le rôle et la portée des quantificateurs dans les formules logiques correspondant à des énoncés. Par exemple, l'énoncé *l'assassin pense tuer un homme ce soir* peut se lire de deux façons différentes :

— *lecture transparente* :

$$\exists x \Big(\text{homme}(x) \wedge \text{pense}\big(\text{l-assassin}, \text{tuer}(\text{l-assassin}, x, \text{ce soir})\big) \Big)$$

signifiant « il existe un individu x particulier qui est un homme et que l'assassin pense tuer ce soir ». Le prédicat *pense* porte sur une variable qui est quantifiée avant lui, donc l'existence de x ne dépend pas de ce que pense l'assassin : x existe bel et bien, c'est un homme, et c'est lui que l'assassin veut tuer.

— *lecture opaque* :

$$\text{pense}\Big(\text{l-assassin}, \exists x \big(\text{homme}(x) \wedge \text{tuer}(\text{l-assassin}, x, \text{ce soir})\big)\Big)$$

signifiant « l'assassin pense qu'il existe un individu x qui est un homme et qu'il va tuer ce soir ». Le prédicat *pense* porte sur une variable qui est quantifiée après lui (à l'intérieur de son arbre syntaxique) et donc c'est l'existence de x qui est pensée par l'assassin : l'assassin pense qu'il existe certainement un x qui est un homme et qu'il va pouvoir tuer.

Le modèle "homonymie" postule l'existence de deux structures sous-jacentes distinctes et univoques. Ce genre d'analyse est très fréquent pour les énoncés qui contiennent des verbes de modalité comme *penser, pouvoir, croire, vouloir...* ou des verbes d'énonciation tels que *dire, énoncer, affirmer*, etc.

L'ambiguïté naît entre les profondeurs et la surface

Dans ce modèle, l'ambiguïté relève de la compétence[1] des locuteurs, et non pas de leur performance[2] en situation de discours. La coïncidence des formes de surface correspondant aux structures profondes est envisagée comme une propriété des énoncés. Il revient entièrement au récepteur d'en assumer la désambiguïsation. En outre, puisque toutes les structures profondes envisagées sont parfaitement définies et univoques, c'est que le langage se décompose en une "réalité" interne univoque, mais s'exprime en surface par un médium qui ne l'est pas. L'ambiguïté s'explique par le fait qu'il "se passe quelque chose" entre ce niveau le plus profond (l'intention de l'émetteur) et la surface, i.e. les énoncés réellement exprimés. Analyser une ambiguïté revient à en trouver les différentes interprétations univoques et à retracer les transformations syntaxiques qui ont conduit à la forme ambiguë. Une critique généralement adressée au modèle "homonymie" (cf. Le Goffic [64]) est la suivante : comment expliquer que des informations ayant une pertinence sémantique non négligeable puissent ainsi "se perdre" entre les structures profondes et les formes de surface à tel point qu'elles sont totalement absentes des énoncés ? Ce genre de dysfonctionnement serait tolérable s'il n'était que ponctuel, mais il semble au contraire systématique, comme dans le cas de la structure *N de N*, qui se retrouve dans d'autres langues que le français. Aucune explication sur cette systématicité n'est avancée par la grammaire transformationnelle-générative, qui pourrait sinon fournir une méthodologie permettant de retrouver les structures profondes pour tout énoncé ambigu donné.

Un "bon" énoncé est transparent

Enfin, dans le modèle "homonymie", les formes de surface sont ignorées (car jugées peu fiables) au profit des structures profondes. Le sens à tendance à être réduit aux référents, en inhibant les signifiés. Cette tendance est très visible dans le cas des ambiguïtés logiques qui posent une alternative "réaliste" : soit le contenu logique porte sur un objet réel, i.e. il existe un référent défini (l'homme que l'assassin veut tuer précisément, et pas un autre), soit il n'y a pas de référent, et dans ce cas, c'est tout une catégorie d'objets qui sont évoqués par la description d'une propriété saillante (l'assassin veut tuer n'importe qui du moment que c'est un homme, et pas une femme par exemple). Le langage est envisagé comme un outil pour établir des références : l'important est de référer, et les moyens utilisés pour le faire sont secondaires. Il y a une nette distinction entre le sens (phénomène linguistique et psychologique) et la

[1] La *compétence* d'un locuteur est son aptitude innée à maîtriser le langage.

[2] La *performance* d'un locuteur consiste en sa maîtrise effective des énoncés réels qu'il émet ou reçoit. Compétence et performance ne correspondent pas obligatoirement à des niveaux de maîtrise équivalents.

référence (qui relève plutôt de la logique, voire de la métaphysique) tout comme chez Frege [37] de sorte que c'est soit l'un soit l'autre qui est à l'œuvre à un moment donné. Soit on est dans un "bon"[1] cas où le langage assume son rôle référentialiste (ce qui correspond à une lecture transparente des énoncés pour auxquels il est possible d'attribuer des valeurs de vérité), soit on est dans un "mauvais" cas sans référence définie, et où il faut passer par un "sens linguistique" pour dégager au moins une propriété caractéristique d'un ensemble sous-déterminé de références possibles. La transparence référentielle est envisagée comme mode de fonctionnement naturel du langage, tandis que les contextes opacifiants sont des accidents du discours.

2.2.2 Le modèle de l'indétermination

L'ambiguïté n'existe pas (ou presque)

Ce modèle s'appuie sur une conception opposée à celle du modèle précédent : l'existence de l'ambiguïté est niée au profit d'une indétermination du sens. On considère que dans un énoncé "ambigu" il n'y a qu'une seule structure, et non pas plusieurs comme précédemment. Cet énoncé est unique car construit à partir d'une unique *valeur linguistique* (ce qui fait qu'il diffère de tous les autres énoncés). Les différentes interprétations possibles dépendent de la situation d'énonciation et sont des accidents du discours ; elles n'entrent pas dans le champ de la linguistique. Dans cette conception, on se désintéresse du fonctionnement du discours. Les intentions de communication de l'émetteur d'un énoncé se situent au même niveau de spécification sémantique que le processus d'interprétation des récepteurs, i.e. au niveau des effets de discours et non pas du langage lui-même. A l'intérieur de ce cadre, seuls les cas absolument flagrants d'homonymie sont considérés comme ambigus, tous les autres relèvent uniquement de l'indétermination du sens.

Martinet [71] cite l'usage de l'article *on* qui peut être utilisé en remplacement de *nous*, *tout le monde*, *vous*, ... et n'est pourtant pas ambigu puisqu'on peut identifier très précisément sa fonction (sa valeur). Cette fonction est de ne pas spécifier la personne de l'agent et de ne pas entrer dans un jeu d'opposition avec les pronoms *je*, *tu*, *il*, ... On est là en franche opposition avec le modèle "homonymie" puisque le signifié linguistique est mis en avant par rapport à la référence : le pronom *on* peut avoir des référents très hétéroclites sans qu'on puisse remettre en question l'unicité de son signifié[2].

D'ailleurs, Weydt [134] entreprend de démontrer qu'un grand nombre d'exemples fournis par les partisans du modèle "homonymie" ne sont pas "fonctionnellement" ambigus, mais qu'il s'agit plutôt d'ambiguïtés concernant le référent, donc en dehors de la linguistique. Ainsi, l'exemple *l'assassin pense tuer un homme ce soir* n'est pas ambigu, mais simplement indéterminé quant à savoir si *un homme* réfère à un individu particulier ou à une classe d'individus par l'intermédiaire d'une propriété commune. Du coup, le sens de cet énoncé est unique, quelles qu'en puissent être les interprétations pragmatiques en contexte.

L'unicité du signe

Le Goffic [64] pense que cette attitude est un prolongement direct du structuralisme linguistique, dans la lignée de Saussure. L'objectif est de déterminer le système de la langue en la ramenant à un réseau d'oppositions distinctives et significatives. Pour ce faire, il faut effectuer le même travail au niveau de la

[1] Cette articulation *bon*/*mauvais* est celle de Le Goffic [64].

[2] Le *signifié* d'un *signifiant* (i.e. un mot ou un énoncé prononcé ou écrit) est sa fonction dans la langue considérée en tant que système.

sémantique que ce qui a été fait en phonologie. Dégager ces oppositions permettrait de définir la valeur de chaque unité signifiante sans se perdre dans les effets de discours, le contexte d'énonciation et la pragmatique. Pour ce genre de structuralisme, il est crucial d'établir l'*unicité* des signes, car c'est elle qui autorise la multiplicité de leurs emplois (quel que soit l'emploi mis en œuvre à un moment donné, le signe reste toujours égal à lui-même). Pour mettre en lumière cette unicité d'un signe, il faut pouvoir préalablement recenser toutes ses oppositions paradigmatiques[1] et tous ses emplois syntagmatiques[2]. Parler de cette *unité structurale* du signe revient à considérer qu'au mépris de l'aspect arbitraire de son émergence dans la langue[3], il se met à assurer lui-même une sorte de cohérence interne en maintenant fermement sa valeur linguistique par opposition aux autres signes. Sans cette valeur unique, le signe "éclate" (pour reprendre l'expression de Le Goffic). Tout cela repose sur un double postulat propre au structuralisme :

— un changement de forme entraîne un changement de sens, puisque deux formes différentes (sauf exception de "variantes") supportent en fait deux signes différents, donc deux valeurs différentes (et par conséquent, la synonymie totale n'existe pas) ;

— et donc à toute forme (sauf accident homonymique) correspond une valeur fondamentale unique.

Ce postulat fait mention d'exceptions : il existe des changements de forme sans changement de sens (comme les radicaux du verbe aller tels que *all-*, *ir-*, et *v-*) et des identités de forme sans identité de sens (c'est le cas des homonymes comme *avocat*-fruit et *avocat*-homme de loi). Il faudrait donc impérativement pouvoir recenser ces exceptions, ce qui n'a manifestement pas été effectué, à charge de preuve la difficulté de tracer la limite entre homonymie et polysémie (cf. §2.3).

Définition absolue de l'ambiguïté : le problème du choix

La notion de valeur qui est employée dans ce modèle présente une part d'incertitude puisqu'il s'agit d'une construction théorique impossible à observer dans la réalité. On ne peut que la déduire d'un réseau d'oppositions. Il est du coup difficile d'établir une frontière entre ambiguïté et indétermination, car il faudrait pouvoir connaître ces réseaux d'opposition de manière exhaustive (tâche impossible sauf pour des cas très restreints ou triviaux). Si l'on se rapporte à l'observation des emplois en contexte d'un signe, on est ramené à un *problème de choix*. C'est ainsi que l'ambiguïté est définie, de manière négative, par F. et D. François dans [36] : il y a ambiguïté, et non pas indétermination, lorsque le récepteur d'un énoncé se retrouve face à un choix à la fois nécessaire et impossible à faire. Pour ces auteurs, un énoncé peut être aussi vague et imprécis que possible sans être ambigu s'il comporte les informations nécessaires pour savoir quelle était l'intention de l'émetteur. Au contraire, dans le cas d'un énoncé réellement ambigu, le récepteur n'a pas à sa disposition de critère déterminant, et doit prendre une décision arbitraire. Le problème de cette définition de l'ambiguïté (cf. Le Goffic [64]) est l'imprécision concernant ce qu'est un "choix nécessaire". En fait, on perd rapidement toute objectivité puisque l'évaluation de cette nécessité repose en dernière instance sur la réaction subjective du récepteur en face d'un énoncé qui lui semble insuffisamment spécifié.

Le Goffic se demande même ce qu'il advient de la compréhension des énoncés, puisque si l'on dit que comprendre un énoncé, c'est pouvoir reconstruire les choix qui donnent accès aux intentions de

[1] En quoi il diffère des autres signes de même catégorie, par exemple en quoi le nom commun *voiture* diffère de tous les autres noms communs.

[2] Quelles sont toutes les positions possibles auxquelles on peut le retrouver dans une phrase.

[3] Nous ne reviendrons pas ici sur "l'arbitraire du signe", et on pourra plutôt se référer aux classiques, notamment Saussure [26].

l'émetteur, et si l'on peut subjectivement avoir l'impression de se retrouver face à un choix à la fois nécessaire et impossible, sans pouvoir prévoir quand cela va arriver, cela signifie que l'on passe une partie de sa vie à ne pas comprendre ce qu'on nous dit. Et puisqu'il n'y a pas de moyen pour prédire quand cette impression va survenir, tout énoncé serait potentiellement ambigu.

Nouvelle définition relative de l'ambiguïté

Pour répondre à cette critique, F. et D. François [36] proposent une définition relative de l'ambiguïté : un énoncé n'est ambigu que par rapport à un autre énoncé, qui permet de différencier les interprétations possibles. Ils proposent d'utiliser des énoncés comme *Je l'ai rencontré dans un café* et *Servez-moi un café* pour trouver le sens de *café* dans *Le café est plein de punaises*. Il s'agit ici d'un cas où les énoncés différenciateurs n'ont pas vraiment besoin d'être produits pour effectuer un choix : n'importe quel francophone sait très bien que *café* possède deux sens bien différents, et tout énoncé comportant ce mot se verra appliquer le sens *café*-boisson et le sens *café*-local avant de choisir lequel est employé. Mais ce cas présente l'avantage d'être franchement tranché : on peut considérer que les deux significations de *café* en font un mot homonymique, issu d'un mot polysémique par métonymie (cf. §2.3).

Pour des cas où les différentes significations ne sont pas aussi bien séparées, il faudra qu'une propriété marquante du contexte se soit présentée au moins une fois afin que le récepteur de l'énoncé se rende compte qu'il y a une distinction de sens à faire, mais que le contexte différenciateur auquel il est habitué est absent. Cela signifie que le contexte doit à la fois aiguiller le récepteur en lui faisant ressentir qu'il a un "choix nécessaire" à faire entre les sens de l'énoncé, et qu'en même temps, ce contexte ne présente pas les marques différentiatrices habituelles, de telle sorte que le choix s'avère "impossible". Le problème est qu'il s'agit encore de "ressentir" la nécessité d'un choix, ou la pauvreté d'un contexte, ce qui fait que différents récepteurs pourront avoir différentes sensations.

On est passé pour le choix entre les sens d'un énoncé ambigu d'une notion de nécessité à une notion de pertinence opportuniste. Opportuniste signifie ici que les critères de pertinence dépendent de la stratégie de lecture (ou d'écoute) adoptée par le récepteur. Par conséquent, les effets du recours à un contexte différentiateur peuvent être très différents suivant les moments ou suivant les récepteurs, en opérant une désambiguïsation à court ou à long terme, ou bien en rendant un énoncé rétrospectivement ambigu.

Similitudes et différences de l'homonymie et de l'indétermination

Le modèle "homonymie" et celui de l'indétermination réservent un sort différent aux éléments de surface, i.e. aux formes réellement manifestées dans le discours. Ces éléments sont jugés primordiaux dans le cadre de l'indétermination, puisque c'est un modèle qui s'appuie sur une théorie du signe. Au contraire, ils se voient attribuer un rôle secondaire dans le cadre de l'homonymie, où ce sont les structures profondes qui sont seules réellement importantes. D'autre part, la signification elle-même n'est pas envisagée de la même façon. Dans le modèle "homonymie", la signification est rapportée à la référence : il y a effacement du rôle de médiateur que joue le langage entre les signes et les états du monde. La triade saussurienne signifiant (i.e. la forme) / signifié (la représentation à l'intérieur du langage ou de l'esprit) / objet référé est réduite à une dyade signifiant / référent. Au contraire, dans la conception structuraliste de l'indétermination, la relation entre le langage et le monde est rejetée hors du champ d'étude. La référence et l'extralinguistique sont considérés comme non pertinents.

Cependant, ces oppositions entre les deux modèles ne doivent pas cacher des ressemblances profondes. Notamment, il présentent tous deux l'ambiguïté comme un accident du langage. La seule forme

d'ambiguïté "réelle" que ces deux modèles reconnaissent est l'homonymie, même si elle est perçue comme fréquente dans l'un et comme rare dans l'autre. Cela implique que le langage est normalement non ambigu, univoque par nature et que l'ambiguïté n'est tolérable que parce que nous disposons de mécanismes de sécurité propres au langage qui permettent d'assurer une relative sauvegarde de la compréhension. De plus, dans les cas réellement inconfortables, c'est-à-dire ambigus, l'ambiguïté est présentée comme propre à certains énoncés relativement à d'autres (les énoncés désambiguïsants) de telle façon qu'elle relève de la compétence et non de la performance. L'ambiguïté est envisagée comme une faille du langage et n'a rien à voir avec les conditions d'énonciation ni le contexte extralinguistique qui produisent, eux, de l'indétermination (encore nommée "vagueness").

2.2.3 Le modèle "polysémie"

Le troisième angle d'attaque présenté par Le Goffic [64] cherche à concilier les visions de l'ambiguïté comme pluralité du modèle "homonymie", et comme unicité du modèle de l'indétermination, en la qualifiant de polysémique. L'idée principale est qu'un énoncé ambigu est bien unique, qu'il ne correspond qu'à une seule structure profonde en termes de sémantique générative, mais qu'il s'interprète avec plusieurs "effets de sens" possibles. Par conséquent, pour décrire ce phénomène, il est impératif d'observer ces effets de sens, qui ne peuvent advenir que dans un contexte d'énonciation. Ainsi, la séparation entre le langage et le discours (entre le linguistique théorique et l'extralinguistique observable) est nettement réduite.

Comportement paradoxal des énoncés ambigus

Cette attitude a déjà été évoquée plus haut concernant les structures en *N de N*. Celles-ci ont été analysées par des structuralistes tels que Tesnière [124] et Pottier [81]. On retrouve dans leurs analyses l'unicité du signe : *de* a toujours la même fonction, qui est d'établir une relation de dépendance entre les éléments qui l'entourent (d'après Pottier), et c'est à tort qu'on veut lui attribuer une valeur sémantique au-delà de cette fonction structurale (d'après Tesnière). Mais en sus de cette unicité vient s'ajouter une multiplicité d'interprétations possibles correspondant à des constructions différentes (d'après Pottier) qui dépendent du schéma actanciel[1] qui serait visible s'il n'y avait pas "translation[2]" (d'après Tesnière). Les tenants de la sémantique générative, et les structuralistes comme Weydt n'ont retenu qu'un aspect ou bien l'autre de ce genre d'analyse. Cependant, même cette analyse semble un peu confuse, car si aucun aspect (dualité et unicité) n'est sacrifié, leur articulation n'est pas clairement explicitée. Le Goffic [64] estime qu'il existe une zone d'ombre dans les analyses fournies par Tesnière, Pottier ainsi que Benveniste [4]. Il attribue cette lacune au fait qu'aucun d'eux n'ose affirmer simultanément les deux propositions contradictoires suivantes :

— un énoncé a un sens unique lié à sa forme ;

— un énoncé peut avoir plusieurs sens liés à des "structures sous-jacentes".

Toujours d'après Le Goffic, ces deux propositions ne sont contradictoires qu'en apparence car elles correspondent à des niveaux différents de spécification de l'interprétation.

[1] La distribution des rôles sujet et objet entre les deux *N*.

[2] Une sorte de symétrie potentielle ou un effet de "champ visuel", comme dans les figures ambiguës des gestaltistes.

Calcul des effets de sens

Culioli [24] a également mené une analyse de ce type sur l'adverbe *bien*, où il essaye de comprendre à la fois l'unicité du signe en recherchant une opération fondamentale dont *bien* serait le marqueur, et la multiplicité de ses interprétations en calculant, à partir de cette opération fondamentale, des effets de sens possibles en fonction du domaine sur lequel porte l'opération. Il peut alors définir un énoncé ambigu comme un énoncé pour lequel plusieurs effets de sens sont calculables.

Par exemple, l'énoncé *il a bien expédié une lettre* peut être interprété comme *il a bien expédié une lettre, lui* (et pourquoi pas moi ? ou pourquoi n'a-t-il pas téléphoné ?), ou *parmi toutes les possibilités, il a choisi d'expédier une lettre, mais cela va-t-il marcher ?*, ou encore comme *il a effectivement envoyé une lettre*[1]. Pour ramener tous ces emplois a une seule opération fondamentale, il faut alors construire une classe de valeurs possible par rapport à un domaine, puis la parcourir, et enfin choisir une valeur.

L'opinion de Culioli est que le calcul de ces valeurs ne peut se faire qu'en considérant les combinaisons marquantes de *bien* avec d'autres éléments de l'énoncé comme les personnes, les modalités, les temps, les aspects, etc. Il s'attache donc à décrire des faits qui sont généralement considérés comme sortant du cadre de la linguistique (de l'ordre du contexte d'énonciation ou de la pragmatique) dans et par des règles grammaticales.

Théorie de l'énonciation

On pourra aussi dire que des phénomènes extralinguistiques, notamment les normes et conventions sociales, ainsi que les croyances très largement partagées au sein d'une communauté, peuvent influencer l'usage des unités linguistiques jusqu'à produire de nouvelles règles de combinaison tacitement reconnues par les locuteurs. C'est ainsi que l'on expliquera, par exemple, le fait que l'adverbe *bien* employé avec la deuxième personne du singulier ouvre une classe de valeurs moins grande qu'avec d'autres personnes. Cette position justifie que l'on puisse retrouver et manipuler des éléments extralinguistiques avec des règles linguistiques, mais seulement parce que les éléments linguistiques gardent une "trace" fossilisée des phénomènes extérieurs propres au vécu de la langue. Il s'agit donc d'une manipulation indirecte : l'extralinguistique ne peut être capté par la linguistique, mais sa trace dans la langue peut l'être.

L'intérêt d'une telle position est qu'elle permet l'élaboration d'une théorie de l'interprétation basée sur la continuité des faits sémantiques, parce qu'elle n'opère pas de coupure entre les phénomènes syntaxiques et les effets de discours, entre l'ambiguïté structurale et la multiplicité des interprétations subjectives possibles. Il s'agirait là d'une théorie se démarquant franchement de l'approche générale concernant la pragmatique et les actes de langage, puisque des distinctions habituelles comme illocutoire *vs* locutoire ou rhétorique *vs* syntaxe s'estompent. Le Goffic [64] appelle cette théorie la "théorie de l'énonciation".

Remarques synthétiques

Ce troisième modèle d'analyse souhaite établir une linguistique de la polysémie, c'est-à-dire de la "diffraction" (Le Goffic [64]) des effets de sens autour de signes uniques. C'est une approche linguistique qui est, comme le modèle de l'indétermination, issue du structuralisme, bien que la polysémie ne soit jamais abordée dans le *Cours de Linguistique Générale* de Saussure. Cela n'est pas étonnant puisque la polysémie pose d'énormes problèmes à une théorie du signe en autorisant les signifiés à se multiplier

[1] Exemples tirés de Le Goffic [64].

sans affecter le signifiant. Cette façon d'appréhender la multiplicité des sens et des emplois des mots à travers la polysémie (et une linguistique des effets de sens) est le résultat d'un désir moderne de réunifier la langue[1] et le discours[2]. Le problème majeur est qu'ils ont été si bien séparés depuis Saussure qu'on ne voit plus vraiment comment articuler les deux : qu'est-ce qui relève du système de la langue proprement dit et qu'est-ce qui est dû aux conditions d'énonciation du discours ?

D'autre part, nous avions souligné la difficulté de séparer l'indétermination de l'ambiguïté, et en définissant l'ambiguïté comme la sensation d'un choix à la fois nécessaire et impossible à faire, tout ou presque dans la langue peut être considéré comme ambigu. Or, à partir du moment où l'on accepte l'existence de la polysémie, presque tous les cas d'ambiguïté, en dehors de l'homonymie stricte, s'avèrent relever de ce phénomène. Tout énoncé est composé d'éléments linguistiques susceptibles d'avoir plusieurs sens, entre lesquels s'opère un filtrage, une exclusion mutuelle. Un énoncé non ambigu est alors un énoncé pour lequel ce filtrage est tellement automatique et naturel qu'il ne demeure aucune impression d'alternative dans l'esprit du récepteur. Au contraire, si le récepteur peut hésiter durant l'interprétation, c'est que l'énoncé est ambigu. La polysémie est donc un élément majeur de la langue, prégnance sur laquelle nous allons revenir dans les paragraphes suivants. Retenons simplement qu'étant donnée la façon dont la polysémie a été introduite jusqu'ici, celle-ci appartient à la langue, tandis que l'ambiguïté caractérise plutôt le discours : *la polysémie est une potentialité d'ambiguïté*.

2.2.4 Conclusion sur le modèles linguistiques de l'ambiguïté

Nous venons de voir comment, à partir de phénomènes observables, les linguistes en sont venus à construire des cadres théoriques permettant d'expliquer ces phénomènes. Alors que le cadre de l'indétermination est souvent préféré en linguistique et que le cadre de l'homonymie a été abondamment repris en traitement automatique des langues naturelles dans la lignée de la grammaire transformationnelle-générative, le cadre de la polysémie est le seul qui parvient à réunir dans un même effort de modélisation la langue et le discours en supposant l'unicité du mot et la pluralité potentielle de ses interprétations. De plus, la polysémie semble pouvoir unifier des cas allant de la simple indétermination référentielle jusqu'à la franche homonymie dans un seul continuum, et de ce fait vient remplir le champ du langage dans son entier. Il s'agit donc d'un concept à la fois riche, souple et prometteur d'une certaine efficacité pour le traitement des ambiguïtés du discours. Mais le concept de polysémie est difficile à définir précisément et à étudier, et c'est pourquoi nous allons tout d'abord en rechercher des formalisations rigoureuses avant de voir comment il peut être appliqué au traitement automatique des langues naturelles.

2.3 Les polysémies

Le terme de *polysémie* a été introduit à la fin du XIXe siècle par Bréal dans le but de caractériser la capacité des mots de "prendre un sens nouveau" qui coexiste avec l'ancien :

> *« Le sens nouveau, quel qu'il soit, ne met pas fin à l'ancien. Ils existent tous les deux l'un à côté de l'autre. Le même terme peut s'employer tour à tour au sens propre ou au sens métaphorique, au sens restreint ou au sens étendu, au sens abstrait ou au sens concret... A mesure qu'une signification nouvelle est donnée au mot, il a l'air de se multiplier et de produire des exemplaires nouveaux, semblables de forme, mais différents de valeur. Nous appellerons ce phénomène de multiplication la polysémie. » [11, p. 155]*

[1] Qui relève de la *compétence*.
[2] Qui relève de la *performance*.

La polysémie caractérise à l'origine un phénomène diachronique[1] d'enrichissement de la langue par multiplication des significations d'un même signifiant. Cet accroissement de significations possibles est souvent le fait d'un enchaînement de *métonymies* successives au fil du temps. La métonymie est une « figure de rhétorique dans laquelle une concept est dénommé au moyen d'un terme désignant un autre concept, lequel entretient avec le premier une relation d'équivalence ou de contiguïté (la cause pour l'effet, la partie pour le tout, le contenant pour le contenu, etc.) » (DFEL[2]) comme dans l'expression *La salle applaudit* qui peut remplacer *Les spectateurs applaudissent*. Victorri et Fuchs [130] font remarquer que le terme bureau a évolué à partir d'un sens aujourd'hui disparu (étoffe de laine qui servait à recouvrir des tables) pour acquérir, à travers de nombreuses métonymies, les différents sens qu'on lui connaît aujourd'hui : meuble sur lequel on écrit, pièce contenant ce type de meuble, lieu de travail possédant ce type de pièce, ensemble d'employés travaillant dans ce type de lieu, etc.

Polysémie ou homonymie ?

En parallèle de la polysémie existe un autre phénomène qui associe des sens différents à une même forme physique : l'*homonymie*. Toujours d'après le DFEL, des mots sont homonymes lorsqu'ils partagent la même prononciation mais ont des significations différentes (comme *pair, père* et *paire*), mais cette définition est couramment étendue aux mots qui partagent également la même orthographe sans que leurs significations soient liées, comme *avocat* (homme de loi) qui provient du latin *advocatus* et *avocat* (fruit) qui, d'après le Petit Robert, provient du nahuatl en passant par l'espagnol.

Le critère généralement accepté pour différentier la polysémie de l'homonymie est l'existence d'un ancêtre, ou *étymon*, commun aux différentes significations. Ainsi, *bureau* est un terme polysémique puisque toutes ses significations remontent au mot désignant une étoffe recouvrant les tables, tandis que les deux exemplaires du mot *avocat* sont homonymes car ils proviennent de deux mots différents. Cependant, la continuité diachronique qui caractérise la polysémie s'est souvent perdue dans la langue contemporaine et des mots qui partagent un même étymon sont perçus comme homonymes. Cela est généralement dû à la disparition de la signification originelle. On est donc obligé de recourir à des critères supplémentaires d'ordre sémantique afin de déterminer si deux exemplaires d'un mot sont polysémiques ou homonymes : éléments de sens communs, sens intermédiaires ou processus de transformation réguliers d'une signification en l'autre grâce à la métonymie ou à la métaphore. Il ne semble cependant pas exister de méthode généralement acceptée pour introduire tel ou tel critère sémantique, ni effectuer de jugement à ce propos.

Les dictionnaires sont un exemple de cette indétermination. L'usage veut que dans un dictionnaire, deux mots homonymes correspondent à deux entrées distinctes, tandis que les différentes significations d'un mot polysémique figurent en tant que subdivisions d'une même entrée. Le Petit Robert considère que les deux acceptions de *grève* (rivage *vs* arrêt de travail) sont homonymes alors que *côte* (os *vs* rivage) est polysémique. De même, on y lit que *pompe* (chaussure) et *pompe* (appareil) sont polysémiques, tandis que *pompe* (appareil) et *pompe* (faste) sont homonymes. Cette indécision se retrouve dans les autres langues. Par exemple, Dollan [28] remarque le même arbitraire dans les définitions du LDOCE[3]. Benveniste donne son avis sur la question lorsqu'il cherche à définir la place de la sémantique dans les sciences du langage :

[1] Qui s'étend dans le temps.
[2] Dictionnaire Francophone En Ligne de l'AUPELF-UREF / Hachette.
[3] Longman's Dictionary of Contemporary English.

> *« Qu'est-ce que c'est que le sens ? Si on regarde de près, on s'aperçoit que les dictionnaires juxtaposent quantité de choses très disparates. Si nous cherchons à* soleil *nous trouvons une définition plus ou moins développée de l'astre que l'on appelle ainsi. Si nous cherchons à* faire*, nous trouverons une douzaine ou une quinzaine de rubriques. Chez Littré, avec les subdivisions il y en a 80. Est-ce que c'est le même sens ? Est-ce que c'est beaucoup de sens ? On ne sait pas. [...] Alors, en général, on dit : l'usage de la langue règle tout ça. Mais nous butons alors sur des questions fondamentales : comment la langue admet-elle cette "polysémie" ? Comment le sens s'organise-t-il ? » [5, p. 20]*

Omniprésence de la polysémie

La polysémie est un phénomène massivement présent dans la langue. Si l'on se fie aux critères énoncés ci-dessus pour différencier les mots polysémiques des mots homonymes dans les dictionnaires, 40% des articles du Petit Robert portent sur des polysèmes, tandis que les mots homonymes comptent pour moins de 5% des articles. Une observation rapide amène également à constater que les mots monosémiques (ceux qui ont un seul sens) sont majoritairement des *termes*, i.e. des éléments appartenant à des nomenclatures techniques. Au contraire, les mots polysémiques appartiennent surtout au vocabulaire courant, celui des mots employés au quotidien (cf. Prince [86]).

Nous nous sommes pour l'instant limité à des exemples de polysémie concernant les "mots pleins", i.e. les noms communs, les adjectifs et les verbes. Mais les autres unités lexicales, telles que les adverbes, les verbes modaux et les prépositions présentent aussi fréquemment un caractère polysémique, alors qu'elles relèvent de la catégorie des *unités grammaticales* plutôt que de celles des *unités lexicales*. A ce sujet, on peut consulter les travaux de Fuchs *et al.* [39] sur le verbe modal *pouvoir*, ou ceux de Le Goffic [65] sur le marqueur *comme*. Cette polysémie des unités grammaticales est contraire à l'intuition : seul le lexique entre en interaction avec les connaissances des locuteurs, la syntaxe étant un système figé. Mais les travaux que nous venons de citer montrent que les unités grammaticales sont elles-mêmes en interaction avec nos connaissances sur le monde, malgré leur contenu lexical jugé pauvre, voire nul. C'est flagrant dans le cas de la marque *-ait* de l'imparfait qui peut donner naissance à de multiples interprétations (cf. Ducrot [29], Berthonneau [6] ou Gosselin [44]) : marque imperfective (*Au moment où je l'ai vu, il partait*), marque hypothétique (*S'il pleuvait, je ne serais pas dehors*), ponctualité perfective (*Ce jour là, l'Allemagne envahissait la Pologne*), modulation hypocoristique (*Il faisait risette à maman, le bébé*), etc. Le contenu lexical de la marque *-ait* est nul. Et cependant, son interaction avec les systèmes de connaissances personnels et conventionnels des locuteurs engendre une multiplicité d'interprétations potentielles qui peuvent entraîner une ambiguïté polysémique (au sens de la définition du DFEL). Par exemple, la phrase *Sans cet argent, je ne pouvais pas m'en sortir* peut être interprétée comme "privé de cet argent, je n'ai pas pu m'en sortir" (perfectif) ou bien comme "privé de cet argent, je serais incapable de m'en sortir" (hypothétique). Cependant, ce genre d'ambiguïté relève d'un type particulier de polysémie pour laquelle on peut *décider* entre les différentes interprétations (la polysémie fonctionnelle, cf. §2.3.2). Les unités lexicales peuvent également présenter d'autres formes de polysémies, plus complexes, pour lesquelles il est plus difficile, voire impossible, de prendre une décision (cf. §2.3.3 et §2.3.4).

La polysémie affecte aussi bien les unités lexicales que les unités grammaticales, ou sublexicales, mais également des unités mésosémantiques de taille supérieure, comme les expressions ou les phrases. De telles phrases ambiguës nécessitent de connaître un contexte plus large afin de décider quelle interprétation est valide. Victorri et Fuchs [130] citent *Il l'a vu avec les jumelles*, qu'on peut interpréter comme "il l'a vu accompagné des jumelles" ou comme "il l'a vu en utilisant les jumelles", ainsi que *Paul aime sa femme et Pierre aussi* dont des interprétations possibles sont "Paul aime sa femme et Pierre aime

également sa propre femme", "Paul aime sa femme et Pierre aime la femme de Paul" ou "Paul aime sa femme et il aime Pierre également". Dans le premier exemple, l'ambiguïté provient de la rencontre entre la polysémie de *avec* (instrumentation ou accompagnement) et l'homonymie de la forme *jumelles* (instrument d'optique *vs* individus homozygotes de sexe féminin). Dans le deuxième exemple, l'ambiguïté est due à une ellipse anaphorique du groupe verbal (cf. p. 8).

On peut aller encore plus loin dans la taille des unités concernées par la polysémie, jusqu'aux paragraphes ou aux textes. Pour s'en convaincre, on peut regarder un des exemples donnés par Prince à propos de problèmes de dysfonctionnement de la communication. Il s'agit d'un texte juridique portant sur le montant des charges déductibles de la déclaration d'impôts :

> *« Au terme de l'article 156 II 2°du Code Général des Impôts, les pensions alimentaires versées aux ascendants sont déductibles pour leur montant réel si les pensions sont versées en espèces. Il appartient au contribuable en cas de litige de faire la preuve devant la juridiction contentieuse de la réalité des versements effectués. Toutefois, le contribuable qui s'acquitte de l'obligation alimentaire en recueillant sous son toit un ascendant dans le besoin peut déduire de son revenu global, sans avoir à fournir de justifications, une somme correspondant à l'évaluation forfaitaire des avantages en nature retenue pour le calcul des cotisations de Sécurité sociale.» [91, citation pp. 93-94]*

Cet exemple fait état d'une interprétation faite par un contribuable dans laquelle la phrase commençant par *toutefois* est considérée par le contribuable comme une extension des possibilités de déduction de ses charges, alors qu'elle est interprétée par un inspecteur des impôts comme une alternative exclusive entre la déduction des avantages en nature et celle des charges en numéraire. Ce texte, pourtant long et détaillé, se révèle donc ambigu. Il s'agit bien d'un phénomène de polysémie (deux interprétations différentes).

La polysémie est donc un phénomène potentiellement omniprésent, qui se retrouve d'un bout à l'autre du spectre des unités linguistiques. Elle est présente avec les unités lexicales (les mots pleins), les unités grammaticales (adverbes, modaux, prépositions), les unités syntaxiques (marqueurs, désinences, voire constructions syntaxiques), et vers l'autre extrémité du spectre, avec les expressions, les phrases et même les parties de discours.

Dans les paragraphes suivants, on cherchera la raison de cette omniprésence, puis on tentera de trouver parmi les travaux des linguistes un *principe directeur* pour les divers cas de polysémie que nous avons vus jusqu'ici. Cela permettra par la suite d'aborder la typologie des diverses formes de polysémie que nous voulons déterminer afin de juger ce qu'il est envisageable de faire dans le cadre du traitement automatique des langues naturelles.

2.3.1 Caractérisation de la polysémie

Nous donnons ici des éléments théoriques plus précis qui vont nous permettre de caractériser les phénomènes polysémiques. Nous pourrons ainsi nous former une idée rigoureuse de ce qui est commun à tous les cas de polysémie, et de leurs rapports avec d'autres phénomènes voisins tels que l'homonymie, la métaphore ou la métonymie. Ces éléments théoriques sont formalisés par Victorri et Fuchs dans [130], et ils vont nous donner l'occasion d'introduire des notions cruciales pour la suite de ce travail.

2.3.1.1 Les énoncés

Le langage est usuellement considéré comme un concept abstrait. Ce qui est réellement observable consiste en phénomènes acoustiques ou visuels : la parole, dite ou écrite. Cette position est déjà exprimée par Saussure [26] au début du siècle. Le langage n'est en fait qu'un concept abstrait à partir de la parole, et qui est plus facilement théorisable que cette dernière. On considérera ici que la parole est constituée d'objets matériels : les énoncés prononcés ou écrits. Cela permet d'articuler le rapport d'abstraction entre parole et langage en séparant deux types d'objets :

— des objets observables, que Victorri et Fuchs nomment *énoncés-occurrences*, qui ne sont pas séparables de leurs conditions d'énonciation (orale ou écrite) ;

— des objets abstraits, ou *énoncés-types*, qu'on peut définir comme les ensembles ordonnés des marqueurs linguistiques utilisés pour exprimer les énoncés-occurrences correspondants (si l'on se restreint au langage écrit, il s'agit des mots, signes de ponctuation et annotations typographiques qui composent la transcription d'un énoncé).

La première différence entre un énoncé-occurrence et un énoncé-type tient dans le fait qu'on peut isoler un énoncé-type de tout contexte, car il s'agit d'une retranscription figée d'un énoncé-occurrence. Au contraire, un énoncé-occurrence est indissociable du contexte dans lequel il a été produit ou reçu[1]. Il en résulte que le sens d'un énoncé-occurrence et le sens d'un énoncé-type n'ont pas le même statut. La parole n'est plus seulement un moyen de communiquer. C'est aussi un outil important de construction du monde dans l'esprit des locuteurs : la parole a pour effet de produire et d'organiser des connaissances sur le monde (ou sur des chimères) aussi bien pour l'émetteur que pour les récepteurs (cf. Vygotsky [132]). On est alors tenté de dire que le sens d'un énoncé-occurrence réside dans sa participation à cette opération de construction du monde, i.e. dans la mise en référence de l'énoncé-occurrence et des objets du monde. Au contraire, l'existence d'un sens pour les énoncés-types est plus discutable, puisqu'ils sont indépendants d'une situation de communication et qu'ils sont composés de signes linguistiques, i.e. d'objets sans impact psychologique direct. Nous choisissons néanmoins de faire avec Victorri et Fuchs [130] le postulat que les énoncés-types ont un sens qui sera défini comme « la contribution constante du matériau linguistique dont [ils sont constitués] au sens de toute occurrence de [ces énoncés]. » En d'autres termes, on appellera *sens* d'un énoncé-type le "potentiel" de sens des occurrences de cet énoncé. Le sens d'un énoncé-type est en quelque sorte la trace des sens de ses énoncés-occurrences dans la langue. Cette définition va à l'encontre d'une opinion assez répandue qui veut que la construction du sens d'un énoncé-type soit le processus "premier". Le sens serait tout d'abord donné par la combinaison des éléments linguistiques, puis mis en rapport avec les conditions d'énonciation. Au contraire, on partira du principe que le sens se situe dans l'énonciation, puis que les formes linguistiques se voient attribuer un sens abstrait à partir de ces événements de la parole que sont les énoncés-occurrences.

2.3.1.2 Critères de comparaison des énoncés

La structuration du discours en énoncés, permet d'approfondir l'intuition que l'essence des phénomènes de polysémie se situe au niveau des relations qu'entretiennent les sens des énoncés. Afin de pouvoir mieux étudier ces relations, nous avons besoin de comparer le sens de deux énoncés. Nous présentons ici des critères méthodologiques de comparaison d'énoncés à travers leurs sens.

[1] L'isoler de son contexte nécessiterait de connaître un nombre virtuellement infini de variables environnementales qui peuvent influencer "l'état d'esprit" des locuteurs au moment de l'énonciation.

La mise en correspondance d'un ensemble d'énoncés-occurrences avec un énoncé-type permet d'évaluer dans une certaine mesure la "proximité" sémantique de deux énoncés-types. Si l'on reprend la définition donnée plus haut, seuls les énoncés-occurrences ont un impact sur l'esprit des locuteurs, et c'est en observant ces effets que l'on peut juger du sens d'un énoncé. Si l'on se contente d'étudier les énoncés-types, on ne peut qu'en déduire des propriétés appartenant au système conceptuel abstrait du langage. On se retrouve alors dans une situation similaire aux mathématiques : les seuls résultats qu'on peut tirer sont des conséquences d'un système d'axiomes et de postulats arbitraires, et il n'y a aucune relation évidente avec la réalité. Pour comparer deux énoncés-types, on va se rapporter à leurs énoncés-occurrences, en "plongeant" les énoncés-types dans des situations d'énonciation. On dira que, pour deux énoncés-types donnés, *plus il y a de conditions d'énonciation dans lesquelles les énoncés-occurrences correspondants on exactement le même effet, et plus les sens des deux énoncés-types sont proches*.

Bien sûr, il est virtuellement impossible que deux énoncés-occurrences différents aient *exactement le même effet quelles que soient les conditions d'énonciation*. Cela signifie que la synonymie parfaite de deux énoncés n'existe pas. Mais on peut tout de même se ramener à des cas optimaux : les *paraphrases*, définies ainsi : deux énoncés-types sont des paraphrases l'un de l'autre lorsque le remplacement d'un énoncé-occurrence de l'un par un énoncé-occurrence de l'autre ne change rien dans "la plupart" des conditions d'énonciation (cf. Victorri et Fuchs [130]). Cette définition de la paraphrase pose deux problèmes méthodologiques :

1. tester exhaustivement l'ensemble des énoncés-occurrences relatifs à un énoncé-type peut s'avérer être une tâche monumentale, et
2. que signifie exactement *la plupart des conditions d'énonciation* ?

Jugement de paraphrase

Victorri et Fuchs proposent de considérer qu'on peut utiliser la notion de *contexte par défaut* (cf. Kleiber [59]) composé de traits prototypiques, présumés partagés par tous les locuteurs. Ils postulent que l'on peut porter des *jugements de paraphrase* sans avoir à tester l'ensemble des conditions d'énonciation (il s'agit d'une pratique courante en linguistique).

Cela dit, la définition des énoncés-type donnée plus haut peut être mise en défaut par les énoncés dits *ambigus*, comme *L'avocat était marron* (l'homme de loi ou le fruit ?) ou bien *Cet ouragan a dévasté notre maison*[1] (le phénomène météorologique ou un individu particulièrement agité ?) Un énoncé-type a été défini comme un ensemble ordonné de signes linguistiques, et l'on se trouve ici dans des cas où un énoncé-type, uniquement caractérisé par sa forme, possède manifestement deux sens radicalement différents. On peut soit décider qu'il s'agit d'un seul énoncé-type malgré l'intuition qu'il doit être séparé en deux éléments à un certain niveau, soit qu'il ne s'agit pas d'un énoncé-type à part entière. Cette dernière solution semble confortée par le fait qu'on peut souvent lever l'ambiguïté en regardant le contexte de l'énoncé en amont ou en aval. On pourrait alors dire qu'une partie de l'énoncé réside dans le contexte dans lequel il s'insère. Cependant, il existe des énoncés ambigus pour lesquels le contexte ne permet pas de lever l'ambiguïté, comme l'exemple de *La chute de la maison Usher*[2], puisque même la lecture de la nouvelle entière ne permet pas de décider entre les deux sens "la ruine du bâtiment dans lequel habitent les Usher" et "la déchéance sociale de la famille Usher". Ce genre de cas, tous comme les jeux de mots, appartiennent à la famille des jeux de langage qui exploitent le fait qu'un même énoncé, dans des

[1] Exemples tirés de Victorri et Fuchs [130].

[2] Donné par Prince dans [86] et que nous reprendrons nous-même plus loin.

conditions d'énonciation particulières, puisse évoquer deux scènes complètement différentes de manière concurrente. On peut alors considérer qu'il s'agit d'énoncés-types dont le sens (c'est-à-dire le potentiel de sens en contexte) possède une topologie à (au moins) deux connexités qui provoque une sensation d'alternative lors de leur interprétation en contexte. Fuchs [38] propose de les appeler *énoncés ambigus*. On retrouve donc la notion de "choix nécessaire" (cf. §2.2.2).

Jugement d'ambiguïté

En plus du *jugement de paraphrase*, Fuchs et Victorri considèrent qu'on peut également porter un *jugement d'ambiguïté* sur des énoncés-types, toujours en se ramenant à l'observation des énoncés-occurrences correspondants. Si l'on se rapporte à la définition de la paraphrase donnée ci-dessus, seul un énoncé qui conserve l'ambiguïté peut être dit paraphrase d'un énoncé ambigu. Cela permet de définir une autre classe d'énoncés : si un énoncé ambigu peut évoquer deux scènes différentes, c'est qu'il existe deux familles d'énoncés qui sont paraphrases de l'une ou l'autre alternative. Ces derniers énoncés seront appelés *énoncés désambiguïsants* et serviront à porter des jugements d'ambiguïté sur les énoncés-types.

2.3.1.3 Sens des énoncés

Grâce au jugement de paraphrase et au jugement d'ambiguïté, on peut se pencher à nouveau sur ce qu'est le sens d'un énoncé-type, et sur la construction dynamique en contexte du sens d'un énoncé-occurrence. De là, on peut tirer une propriété caractéristique de la polysémie, à savoir ses effets sur ce processus de construction du sens.

Segments et expressions

Un énoncé n'est pas une entité monolithique : il est composé de séquences de signes linguistiques qu'on peut isoler des autres de façon à ce qu'ils conservent une certaine signification, même en dehors de l'énoncé. Il ne s'agit pas des mots, dont la définition n'est affaire que de conventions typographiques et qui ne correspondent pas forcément à des unités sémantiquement chargées. Pour le moment, on se contentera d'appeler ces séquences des *segments* de l'énoncé et de noter qu'ils peuvent être discontinus (comme *ne... pas...*) D'autre part, la "quantité" du sens d'un segment qu'il porte lui-même n'est pas toujours identique. Ainsi, dans l'exemple fourni par Launay [63] qui compare les énoncés suivants :

1. *L'homme n'est qu'un roseau pensant.*
2. *L'homme arriva exténué.*

l'auteur estime que le segment *l'homme* a le même sens, mais des références différentes. Pour lui, le sens dépend uniquement du segment, tandis que la référence dépend d'un calcul portant sur l'énoncé entier. Cette position revient à voir la construction du sens d'un énoncé comme le résultat de deux mouvements de statuts différents : un mouvement ascendant purement linguistique qui calcule la *signifiance*[1] de l'énoncé à partir des plus petits segments en vertu d'un principe de compositionnalité, et d'un mouvement descendant exprimant l'influence de l'énoncé sur ses segments qui se ramène alors à un pur calcul référentiel. Au contraire, d'après Victorri et Fuchs [130], on ne peut supposer l'unicité du sens d'un segment et reléguer les problèmes dans le domaine de la référence. Ils considèrent dans l'exemple

[1] Le sens sans la référence dans la terminologie de Launay.

ci-dessus que les deux occurrences du segment *l'homme* ont des sens différents. Cela conduit à admettre l'existence dans un énoncé de segments dont le sens n'est pas entièrement contenu en eux-mêmes. Le sens dépend alors du segment et de l'énoncé dans lequel il s'insère. On appellera un tel segment une *expression linguistique* et lorsqu'on parlera du sens d'un énoncé, il s'agira en fait du *sens d'une occurrence donnée d'une expression linguistique donnée dans un énoncé donné*. C'est ce qui explique que des phrases comme *Une femme est une femme* ne sont pas tautologiques. Du coup, le sens d'une expression dans un énoncé n'est qu'une composante du sens de l'énoncé. C'est la part du sens de l'énoncé qui est attribuée à cette expression en termes de *potentiels de sens contextuels*.

Segmentation structuraliste

Admettre l'existence des expressions linguistiques est une chose, mais comment faire pour les identifier dans un énoncé donné ? En linguistique structuraliste, on identifie les expressions d'un énoncé en remplaçant tout résultat possible d'une segmentation de l'énoncé par un autre segment, de manière à rechercher de nouveaux énoncés acceptables (pour lesquels il existe des conditions d'énonciation où l'énoncé-occurrence correspondant a un sens). Une version particulièrement efficace de cette méthode est d'essayer de produire des paraphrases de l'énoncé de départ, et en observant la stabilité des transformations opérées par le remplacement du segment, on en déduit s'il est ou non une expression. Pour appliquer ce genre de méthode, il faut cependant postuler l'existence d'expressions de taille minimale (sinon le nombre de segmentations possibles d'un énoncé devient infini) qui sont des unités proches des *morphèmes* ou des *monèmes* de Martinet [72].

Comportement paraphrastique

On dispose maintenant des outils nécessaires pour chercher comment caractériser le sens d'une expression dans un énoncé. Victorri et Fuchs proposent d'étudier son *comportement paraphrastique*. Pour cela, on considère une expression linguistique donnée dans un énoncé donné. En regard de ce qui a été dit précédemment, on peut construire une famille d'énoncés de sens "très proche" de celui de l'énoncé de départ, grâce au jugement de paraphrase en remplaçant uniquement l'expression considérée. On construit donc un ensemble de paraphrases de l'expression relativement à l'énoncé donné. Observer le comportement paraphrastique de l'expression consiste à considérer que le sens de cette expression est caractérisée par l'ensemble des paraphrases qu'on peut en construire par rapport à l'énoncé de départ.

Apport propre

A partir d'ici, on peut introduire une dernière définition du sens, qui va servir à proposer en conclusion une caractérisation de la polysémie. On cherche maintenant à définir le sens d'une expression hors énoncé, ce que Victorri et Fuchs appellent le *sens d'une expression* (par opposition au sens d'une expression relativement à un énoncé). C'est ce sens qui figure dans les dictionnaires sous forme d'une formulation abstraite accompagnée d'exemples. En fait, cette structuration des entrées des dictionnaires correspond à un phénomène qui a été évoqué plus haut, à savoir que le sens d'une expression n'est pas entièrement contenu par elle. Il y a donc une part de sens porté par l'expression et qui représente *l'apport propre* de l'expression, le facteur invariant de tous ses sens possibles. Le reste de son sens réside dans les interactions de l'expression avec l'énoncé dans lequel elle s'insère (les exemples). Il s'agit en quelque sorte de règles particulières qui viennent modifier l'emploi de l'expression de façon à ce qu'elle "fonctionne" dans un contexte donné.

On peut réaliser un classement des expressions linguistiques en fonction de l'importance relative de leur apport propre et de leurs règles d'interaction. Un cas extrême est celui des expressions sans règle d'interaction : les expressions monosémiques, dont tout le sens est contenu dans leur apport propre. C'est le cas des termes techniques définis par une nomenclature conventionnelle. A l'autre extrême, deux cas sont possibles :

— les mots lexicalement vides, comme les prépositions dites "incolores" (*à, de...*) qui ont un apport propre réduit au minimum (mais cependant pas nul, puisque comme nous l'avons vu plus haut, ces unités grammaticales peut être soumise à la polysémie), et

— les vrais homonymes, i.e. les expressions ayant une même forme mais des sens qui n'ont rien à voir les uns avec les autres. Dans ce cas, on peut dire que leur apport propre est nul (puisque la forme pure ne détermine pas les sens).

Définition de la polysémie

Au terme de cette introduction méthodologique, nous adopterons la définition de la polysémie proposée par Victorri et Fuchs :

> *« Une expression polysémique est une expression dont le sens ne se réduit ni à son apport propre, ni à l'apport du reste de l'énoncé dans l'interaction avec les énoncés dans lesquels elle est insérée. » [130, p. 38]*

Cette définition présente bien sûr un caractère très général des phénomènes de polysémie. Or, comme nous l'avons dit plus haut, ces phénomènes peuvent présenter des différences de nature assez importantes, tout en demeurant dans le cadre de cette définition. A partir d'ici, nous allons nous concentrer plus particulièrement sur les phénomènes de polysémie qui concernent les unités lexicales pleines, i.e. les noms communs, les verbes et les adjectifs, ainsi que sur les ambiguïtés lexicales qui en résultent.

Les ambiguïtés lexicales ont été abondamment étudiées en traitement automatique des langues naturelles. C'est en nous basant sur de telles études que nous allons pouvoir catégoriser différents types de phénomènes de polysémie. Nous présentons dans les paragraphes qui suivent une typologie des polysémies basée sur la quantité d'informations requises pour traiter automatiquement les ambiguïtés lexicales qu'elles engendrent, et sur la difficulté de faire aboutir ce traitement.

2.3.2 Polysémie fonctionnelle

La première catégorie de phénomènes de polysémie que nous abordons est étudiée depuis des années en traitement automatique des langues naturelles et en sciences cognitives. Les linguistes peuvent objecter qu'il ne s'agit pas toujours de polysémie au sens strict, i.e. d'occurrences d'unités lexicales dont toutes les interprétations sont fortement reliées les unes aux autres. Mais ce qui nous importe est qu'ils soient considérés dans les faits comme polysémiques par les chercheurs en traitement automatique des langues naturelles et que des systèmes aient été proposés, qui permettent de les traiter en tant que tels.

Les cas dont nous parlons concernent des unités lexicales pouvant donner lieu à de multiples interprétations. Leur caractéristique essentielle est qu'il est possible d'effectuer un *choix* entre les différentes interprétations possibles en fonction du contexte linguistique dans lequel l'unité lexicale apparaît. Cette forme de polysémie est appelée *polysémie fonctionnelle* par Prince et Bally-Ipsas dans [93]. Les auteurs

entendent par là que c'est la polysémie qu'on peut résoudre par la *fonction* associée à l'unité lexicale dans son contexte. Parmi les exemples donnés par les auteurs, on trouve : *les cordes* (à linge) ou *les cordes* (de l'orchestre), *le plateau* (de Gif-sur-Yvette) ou *le plateau* (en argent). Un autre exemple donné par Kayser [58] concerne *élection* (comme procédure) dans *nous procédons à l'élection du représentant* et *élection* (comme résultat) dans *l'élection de M. Dubois à la tête du parti...* Ce genre de phénomènes ne constituent pas un obstacle majeur au traitement automatique car ils peuvent être pris en charge et résolus par des "gestionnaires de personnages" et des "gestionnaires de thèmes" (cf. Sabah [107, 108]).

Il est souvent fait mention dans la littérature du fait que les ambiguïtés lexicales constituent un "faux problème", artefact du traitement automatique des langues naturelles. Car si les systèmes automatisés considèrent qu'ils se trouvent face au problème d'une alternative à résoudre, les locuteurs humains choisissent naturellement *la* bonne interprétation sans même avoir conscience d'un choix. Or, la plupart des auteurs en linguistique considèrent que ce problème du choix existe bel et bien (cf. §2.2.2 et 2.2.3) et qu'il se caractérise par une "impression d'alternative". Ces deux opinions sont cependant conciliables, car les informaticiens ont tendance à ne considérer que les ambiguïtés lexicales qui sont produites par la polysémie fonctionnelle et pas les autres formes de polysémie. Or, la polysémie fonctionnelle conduit en effet généralement à des ambiguïtés artificielles (invisibles pour l'humain), au contraire des autres polysémies. Le problème, en ce qui concerne le traitement automatique des langues naturelles, est donc la volonté de faire entrer toutes les ambiguïtés lexicales dans le seul formalisme de la polysémie fonctionnelle. Nous reviendrons plus en détails sur ce qui ne semble être qu'un problème d'*adéquation de formalismes* au §3.2.

L'approche informatique des problèmes en général privilégie un mode de réflexion *non-ambigu*, basé sur des *critères de détermination univoques*. Or, dans le cas de la polysémie fonctionnelle, on se trouve naturellement dans une situation où, face à un ensemble d'interprétations possibles d'une unité lexicale ambiguë, une seule d'entre elle correspond effectivement à la *fonction* de l'unité dans l'énoncé où elle s'insère. Il y a là, pour une fois, adéquation entre le formalisme utilisé et les faits.

Restriction de la sélection

La polysémie fonctionnelle correspond en effet à des situations où il est possible de déterminer le bon sens d'une unité lexicale ambiguë grâce à la présence d'un trait sémantique générique récurrent dans le contexte proche, sachant que ce trait ne peut être compatible qu'avec une seule des interprétations possibles de l'unité ambiguë. La méthode qui consiste à identifier ce trait et à s'en servir comme critère de choix est la *restriction de la sélection du signifié* (ou *filtrage sémantique*). Ce type de polysémie est relativement aisé à traiter par des systèmes informatiques classiques car il s'apparente beaucoup au fonctionnement des grammaires de cas ou de traits. Cette polysémie est donc représentable au sein d'un paradigme de modèles issus de la théorie des langages formels. Mais tous les cas de polysémie ne relèvent pas de la polysémie fonctionnelle. Il existe des catégories de polysémie beaucoup plus difficiles à traiter en traitement automatique des langues naturelles, et qui ont été beaucoup moins étudiées.

2.3.3 Polysémie d'acception

Emplois, acceptions et sens

Les linguistes ayant adopté l'approche dite d'*analyse componentielle* (cf. p. 68), notamment Martin [70] et à sa suite Rastier [99], mentionnent la nécessité de distinguer *emplois*, *acceptions* et *sens* en ce qui concerne les ambiguïtés lexicales engendrées par l'homonymie et la polysémie. Dans les paragraphes

suivants, nous n'allons pas recourir à la terminologie originelle (que nous présenterons au §2.5.2.2), mais à un vocabulaire *ad hoc* plus proche de celui utilisé en intelligence artificielle.

L'analyse componentielle consiste à représenter un signifié lexical par un ensemble de traits sémantiques. Ce sont eux, ainsi que leurs statuts, qui permettent de différencier les sens des emplois et des acceptions. En résumant rapidement, un trait sémantique possède trois attributs, qui peuvent prendre chacun deux valeurs mutuellement exclusives :

— le trait peut être *propre* au signifié (il fait partie de sa définition) ou bien *transféré* d'un autre signifié apparaissant dans le contexte local ;

— parallèlement, il est *activé* ou *désactivé*, suivant qu'il est pris en compte ou pas dans l'évaluation de la valeur sémantique du signifié ;

— enfin le trait est soit *hérité* (d'une classe de signifiés plus large à laquelle appartient le signifié), soit *ajouté* s'il contribue à distinguer des sous-classes du signifié.

Ce cadre permet de distinguer trois types de signifiés parmi les multiples interprétations d'un même signifiant. Rastier, compilant dans [99] les travaux des linguistes différentialistes, donne la typologie suivante :

— Des *emplois* sont des occurrences de signifiés qui diffèrent par au moins un trait sémantique transféré par le contexte. Ainsi, la définition de *convoi* dans le Petit Larousse renvoie à trois interprétations possibles :

  i. « Suite de véhicules transportant des personnes ou des choses vers une même destination. »
  ii. « Suite de voitures de chemin de fer entraînées par une seule machine. »
  iii. « Cortège accompagnant le corps d'un défunt à une cérémonie de funérailles. »

  L'interprétation (ii) correspond à l'ajout d'un trait /ferroviaire/ au faisceau de traits qui constituent l'interprétation (i), et l'interprétation (iii) correspond à l'ajout d'un trait /funéraire/. Bien entendu, on pourrait envisager l'ajout d'autres traits sémantiques, comme /militaire/ ou /exceptionnel/. Sachant que chacune de ces interprétations correspond à l'addition (ou à la désactivation) d'un trait transféré par le contexte, elles ne correspondent pas à des variations autour d'un même signifié (i.e. à des acceptions) ; il s'agit d'emplois différents du même signifié qui se colore différemment en fonction du contexte.

— Des *acceptions* (en termes techniques) sont des occurrences de signifiés qui diffèrent par au moins un trait sémantique transféré par une norme sociale. Pour reprendre un exemple donné par Martin [70] et repris par Rastier (*op. cit.*), on retrouve parmi les interprétations de *minute* les deux entrées suivantes dans n'importe quel dictionnaire (ici le Petit Larousse) :

  i. « Unité de mesure du temps valant soixante secondes. »
  ii. « Court espace de temps. »

  Ces deux interprétations diffèrent par la présence ou l'absence du trait /court/ qui résulte de l'application d'une norme sociale. Ces deux auteurs fournissent à titre de preuve des combinaisons lexicales telles que *cocotte minute* ou *serpent minute*, et des extraits de phraséologie orale tels que *attends une minute*, ou écrite comme les *longues minutes* qui foisonnent dans les romans de gare.

Le trait sémantique /court/ transféré par norme sociale est alors activé dans l'interprétation (ii) tandis qu'il est désactivé dans l'interprétation (i). La différence essentielle avec le cas précédent (les emplois) est qu'ici, l'addition ou la suppression du trait transféré est fixée par une norme, figé dans l'usage du langage. Les différentes interprétations du signifiant correspondent bien à des variantes différentes d'un signifié de base (qui peut très bien n'être jamais réalisé dans la langue usuelle).

— Des *sens* sont des occurrences de signifiés qui diffèrent par au moins un trait sémantique propre. Par exemple, le signifiant *blaireau* correspond dans le Petit Robert aux interprétations :

i. « Mammifère carnivore [...] bas sur pattes, plantigrade, de pelage clair sur le dos, foncé sous le ventre, qui se creuse un terrier. »

ii. « Pinceau fait de poils de blaireau dont se servent les peintres, les doreurs [...] »

Pour Martin (*op. cit.*), il s'agit de deux acceptions en relation métonymique, c'est-à-dire que le faisceau de traits sémantiques constituant l'interprétation (i) se retrouve contracté sous la forme d'un seul trait ajouté dans le faisceau de traits correspondant à l'interprétation (ii). Rastier objecte que les traits /mammifère/ et /carnivore/ qui sont présents dans la première interprétation ne le sont certainement pas dans la deuxième, même sous forme contractée. Parmi les traits propres de l'interprétation (i), seuls les traits /pelage/, /clair/ et /foncé/ sont également propres à l'interprétation (ii). On peut aussi légitimement remarquer que dans un contexte compatible avec l'interprétation (ii), il y a nécessairement une lecture sélective des traits propres, de manière à ne retenir que ceux qui figurent dans l'interprétation (ii), et à évacuer ceux qui sont présents dans l'interprétation (i) comme /plantigrade/ ou /bas sur pattes/. Il ne s'agit donc pas d'une "variation sur thème" comme avec les acceptions, mais bien d'interprétations tout à fait différentes du même signifiant : ce sont des sens différents.

Les sens tels qu'ils viennent d'être définis, correspondent aux multiples interprétations potentielles qu'on rencontre dans les cas de polysémie fonctionnelle. Le principe de ce que Rastier appelle la lecture sélective des traits propres est complètement similaire au principe de restriction de la sélection du signifié par un trait sémantique compatible avec le contexte (cf. p. 25 et §3.2.2). On voit du même coup que la polysémie fonctionnelle est une forme de polysémie relativement simple en comparaison des cas d'ambiguïté portant sur des acceptions ou des emplois, et non des sens, différents.

Figement dans la langue

Il est important de mettre l'accent sur le fait que les différentes acceptions d'un signifié sont construites autour d'un signifié "de référence" ou "thématique" par un figement dans la langue. Il peut très bien n'exister aucun moyen d'exprimer ce signifié thématique en langue. Ainsi dans le cas de *minute*, les deux acceptions qui ont été mentionnées sont des variantes d'un hypothétique signifié *espace de temps* qui n'est jamais réalisé : on rencontre soit sa variante technique (la minute de soixante secondes) soit sa variante familière (le court moment), et jamais lui-même exprimé sous une forme propre. Mais le plus crucial est le fait que la norme sociale dont parlent Martin et Rastier semble en dernier lieu échapper au domaine sociologique : il s'agit plutôt d'une trace dans la langue d'un conditionnement social, et c'est pourquoi on peut parler de figement dans la langue. Le caractère dynamique des actes sociologiques qui sont à l'origine de la différenciation entre les acceptions s'est perdu lorsqu'elles sont devenues des faits linguistiques : chaque acception d'un signifié thématique semble appartenir à un lexique particulier, et c'est ainsi que nous comprenons l'assertion de Rastier soutenant que les acceptions appartiennent à

des sociolectes[1] et relèvent, en termes de représentations, de la lexicologie plutôt que de la linguistique restreinte.

Sous-détermination de l'interprétation

Les cas d'ambiguïtés portant sur des acceptions ont fait l'objet d'une tentative de modélisation informatique par Brugère [16], que nous présenterons rapidement au §3.4 à des fins d'illustration. Le système résultant *interprète* une occurrence ambiguë en proposant une *liste ordonnée* d'acceptions possibles, réduite au nombre minimum d'éléments. On n'est donc plus dans une situation où les choses sont suffisamment claires pour éliminer toutes les mauvaises interprétations ; *on ne peut proposer, dans le cadre de la polysémie d'acception, qu'une liste ordonnée d'acceptions candidates à l'interprétation, sachant que l'une est vraisemblablement mieux adaptée que les autres.*

2.3.4 Polysémie d'usage

On a vu au paragraphe précédent que les différents emplois d'une unité lexicale diffèrent par au moins un trait sémantique transféré par le contexte (d'énonciation). On entre donc pratiquement dans le domaine de la *métaphore vive*, i.e. de la modification imprévue (voire imprévisible) d'un signifié par manipulation de son usage dans la langue. Contrairement à la polysémie fonctionnelle qui peut être décrite en restant à l'intérieur du système fonctionnel de la langue (donc relevant de la linguistique traditionnelle), et à la polysémie d'acception qui met en œuvre des traces figées dans la langue de normes sociales (relevant d'une lexicologie qui tient compte du reflet des faits de société dans la langue réellement utilisée), les cas de polysémie qui engendrent des ambiguïtés sur les emplois des unités lexicales nécessitent une description linguistique qui tienne compte des coordonnées spatiales et temporelles de l'énonciation, et de tout ce qui en découle (modes, sens locaux ou temporaires, références implicites...) On abandonne la linguistique pure pour aller à la rencontre de la sociologie. Avec la *polysémie d'usage*, il s'agit de voir comment on peut jouer de manière inventive et spontanée sur les différentes conventions sociales d'usage d'une unité lexicale pour produire une signification *ad hoc* et originale.

Considérons par exemple le titre de la nouvelle d'Edgar Poe : *La chute de la maison Usher*[2]. Cet exemple consitue un cas à la fois typique, car l'ambiguïté des emplois est nettement visible, et relativement simple car il s'agit d'alternatives binaires. Dans le contexte même de la nouvelle, il n'est pas possible de lever l'ambiguïté concernant son titre : il peut aussi bien s'agir de la ruine physique du bâtiment dans lequel habitent les Usher, que de la déchéance sociale de leur famille, comme illustré dans la figure 2.1 :

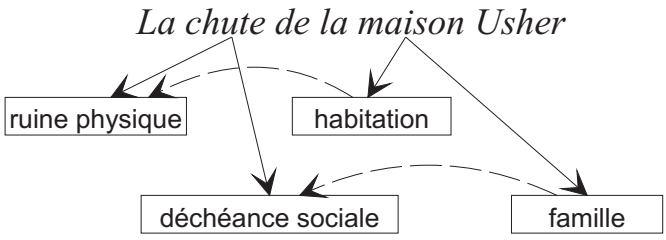


FIG. 2.1: Alternatives d'interprétation pour *La chute de la maison Usher*

Deux types de polysémie sont à l'œuvre :

[1] Les sous-langages propres à des groupes sociaux déterminés.
[2] Analysé en premier lieu par Prince [86] dans le cadre de la polysémie des mots courants.

— L'ambiguïté concernant l'interprétation de *maison* relève de la polysémie d'acception : une convention sociale (obsolète maintenant) a entériné dans la langue française la métonymie entre la famille et le bâtiment qui lui sert d'habitation. Ainsi, parmi les définitions données par le Petit Larousse, on trouve :

i. « Bâtiment construit pour abriter une famille. »

ii. « Membres d'une même famille. »

La première interprétation, par extension implicite, peut s'analyser sous forme de traits sémantiques comme /bâtiment/ + /abriter/ + /membres/ + /famille/, tandis que la seconde correspond à un faisceau de traits plus restreint : /membres/ + /famille/. Entre ces deux interprétations, il y a désactivation des traits /bâtiment/ et /abriter/. Cette désactivation résulte de la normalisation sociale de la métonymie *habitation de la famille* ⇒ *famille*. C'est un cas de polysémie d'acception (cf. §2.3.3), puisque les deux interprétations possible de *maison* diffèrent par au moins un (et en l'occurrence deux) trait(s) sémantique(s) transféré(s) par convention sociale.

— L'ambiguïté qui porte sur *chute* relève en revanche de la polysémie d'usage. Les deux interprétations possibles de *chute* correspondent à la même acception, mais avec des domaines d'application différents. Ces deux interprétations sont d'ailleurs considérées par le Petit Larousse comme les deux aspects d'une même sous-entrée :

i. « Fait de s'écrouler : ruine, effondrement. »

ii. « Le fait de commettre une faute, de tomber dans la déchéance. »

Si l'acception correspondant à cette sous-entrée se caractérise par une décomposition simple en termes de traits sémantiques /effondrement/, la première interprétation de *chute* peut être analysée comme /effondrement/ + /physique/, et la seconde comme /effondrement/ + /moral/ (ou /social/). Il s'agit bien du même phénomène (et donc du même signifié) dans les deux cas, à ceci près que le contexte d'énonciation transfère ponctuellement le trait /physique/ ou /moral/ pour indiquer le domaine d'application du phénomène. Ces deux interprétations correspondent à la définition des emplois donnée plus haut.

Définition de la polysémie d'usage

L'ambiguïté des emplois de *chute* est une conséquence de l'ambiguïté d'acception portant sur *maison*. Les deux acceptions associées à *maison* impliquent des usages diférents d'une même acception de *chute* par l'activation implicite de deux domaines d'application différents de manière simultanée. C'est ce que nous nommons la *polysémie d'usage* : l'activation simultanée de plusieurs applications pour une même acception d'un signifié, de façon à mettre en avant des variations subtiles de signification. En fait, ainsi qu'il sera exposé au §2.4.1, la polysémie d'usage est, la plupart du temps, rendue possible par la présence d'une occurrence d'un des deux autres types de polysémie (fonctionnelle ou d'acception) dans le contexte proche.

Atomicité des interprétations

Au contraire de la polysémie fonctionnelle et, dans une moindre mesure, de la polysémie d'acception, qui sont plutôt des inconvénients dont on cherche à se débarrasser, la polysémie d'usage semble avoir une

finalité en soi. En effet, l'ambiguïté qui réside dans l'interprétation de *La chute de la maison Usher* est délibérée de la part de l'auteur[1] : il est nécessaire que le lecteur ait les deux interprétations simultanément à l'esprit en lisant la nouvelle (i.e. le contexte). Une représentation du résultat du parcours interprétatif de ce titre devrait être un *ensemble* comportant les deux interprétations possibles. Il existe donc un mouvement allant de l'atomique à l'ensembliste au sein des différentes catégories de polysémie :

— dans le cas de la polysémie fonctionnelle, seul l'un des sens du signifiant polysémique est compatible avec le contexte, et il s'agit de ne sélectionner que celui-ci en éliminant les sens erronés grâce à la fonction de l'occurrence du signifiant/signifié dans l'énoncé ;

— la polysémie d'acception présente un caractère moins tranché. Le contexte d'énonciation rend plusieurs acceptions envisageables mais en les favorisant différemment, de telle sorte qu'on peut réussir à les ordonner pour ne sélectionner que la plus plausible ;

— la polysémie d'usage, quant à elle, active simultanément plusieurs emplois d'un même signifié et ce à dessein dans la majorité des cas. Que ces différents emplois soient aussi importants les uns que les autres, ou qu'on puisse leur attribuer des importances différentes, ils participent *tous* à la signification en contexte de l'occurrence polysémique.

Pour ces trois catégories de polysémie, on devrait donc envisager trois types de structures différents permettant de représenter les résultats de leur traitement : une structure atomique pour la polysémie fonctionnelle, une structure de liste ordonnée pour la polysémie d'acception, et une structure d'ensemble (dont les éléments sont éventuellement valorisés) pour la polysémie d'usage.

Polysémie des mots courants

Le traitement des ambiguïtés engendrées par la polysémie d'usage est un domaine quasiment vierge en traitement automatique des langues naturelles. Les seuls travaux allant dans ce sens sont ceux de Prince [88, 92, 93] concernant la *polysémie des mots courants*. Ceux-ci représentent, d'après l'auteur, un ensemble d'environ 1000 formes racines[2] au sein de chaque langue. Ce sont des mots employés quotidiennement qui sont susceptibles d'être énormément sollicités par les phénomènes de polysémie. Ces mots expriment généralement des notions de base telles que les relations familiales proches, les parties du corps, les processus vitaux ou les mesures du temps et de l'espace. On retrouve ces catégories dans les travaux d'anthropologie sous la désignation de "taxonomies anthropologiques" (cf. Werner [133]).

L'approche proposée par Prince se base sur les deux constatations suivantes :

— Les mots courants affectés de polysémie sont "trop polysémiques" pour trouver des critères permettant de discriminer une bonne interprétation de manière univoque tout en rejetant les autres.

— L'ambiguïté qui en résulte ne nuit pas forcément à la communication. Elle peut même constituer un outil de communication efficace.

Par conséquent, plutôt que de chercher un moyen de lever les ambiguïtés coûte que coûte, Prince propose d'*intégrer l'ambiguïté dans son modèle de traitement*, ce qui rompt totalement avec les habitudes établies

[1] Ou alors, des millions de lecteurs font preuve d'un singulier manque de perspicacité depuis des dizaines d'années.

[2] Une forme racine est le représentant canonique d'une classe d'unités lexicales permettant d'engrendrer l'ensemble de ses flexions, comme *blanc* pour *blanc*, *blancs*, *blanche*, *blanches*, *blanchir*, *blanchi*, etc.

en traitement automatique des langues naturelles. Nous reviendrons à la fois sur cette conception de l'ambiguïté au §2.4.3 et sur le modèle de traitement proposé par Prince (EDGAR) au §3.6.2.

A ces observations concernant les mots courants, nous voulons ajouter que des unités lexicales plus techniques peuvent également être affectées par la polysémie d'usage, comme nous le verrons au §2.4.1.3, mais il s'agit alors de mots ou d'expressions d'usage extrêmement répandu au sein d'un langage de spécialité. Il semble que, dans ce cas, la notion de quotidienneté ait été déplacée vers des lexiques spécialisés.

2.4 Polysémie d'usage et ambiguïté

Préalablement à la conception d'un modèle de traitement de la polysémie d'usage, il est nécessaire d'illustrer ce phénomène de façon plus détaillée. Les exemples suivants vont nous permettre de mieux cerner les caractéristiques des différentes occurrences de la polysémie d'usage. Cela nous autorisera également à étayer ce que nous avons dit précédemment du rapport particulier qu'elle entretient avec la notion d'ambiguïté, rapport radicalement différent de ceux attachés à la polysémie fonctionnelle et à la polysémie d'acception.

2.4.1 Occurrences de polysémie d'usage

Tout comme le terme *polysémie* désigne une large variété de phénomènes, la polysémie d'usage peut également se réaliser de différentes façons dans la langue quotidienne et surtout dans ses textes. Il suffit d'observer des textes d'origines diverses appartenant à une langue "vivante", i.e. n'appartenant pas au corpus classique des exemples habituellement fournis par les linguistes. Les slogans publicitaires, les articles de journaux, les revues spécialisées et les romans fourmillent d'exemples de polysémie d'usage.

De cette observation, nous avons tiré un certain nombre d'exemples qui nous ont permis de délimiter quatre grandes catégories d'occurrences de la polysémie d'usage :

— les transferts d'usage ;

— les cumuls d'emplois délibérés ;

— les "mots jokers[1]" ;

— les jeux de langage.

2.4.1.1 Transferts d'usage

Nous appelons *transfert d'usage* l'emploi d'un mot polysémique hors de son domaine d'application le plus courant, tout en conservant une acception déjà conventionnellement définie. C'est un phénomène qui se produit généralement à des fins d'illustration d'un concept technique, ou d'explicitation, en faisant référence à des connaissances supposées établies chez le récepteur de l'énoncé. L'usage habituel de cette acception du mot polysémique est ainsi enrichi par son application à un nouveau domaine de façon imprévue.

[1] Cette expression est utilisée par Prince dans [84].

— *La cellule **mère** produit deux cellules **filles** identiques.*[1]

La relation mère/fille est appliquée ici au domaine de la biologie. L'utilisation de ces deux mots appartenant au vocabulaire des relations de parenté proche permet de transmettre un contenu sémantique plus riche qu'avec un vocabulaire strictement technique, comme par exemple *la cellule originelle produit deux cellules résultantes identiques*. Pour s'en convaincre, il suffit d'appliquer les techniques structuralistes de "variation paradigmatique des segments" (cf. p. 23), à savoir remplacer un segment de l'énoncé par d'autres segments afin d'observer si l'énoncé conserve son sens ou non.

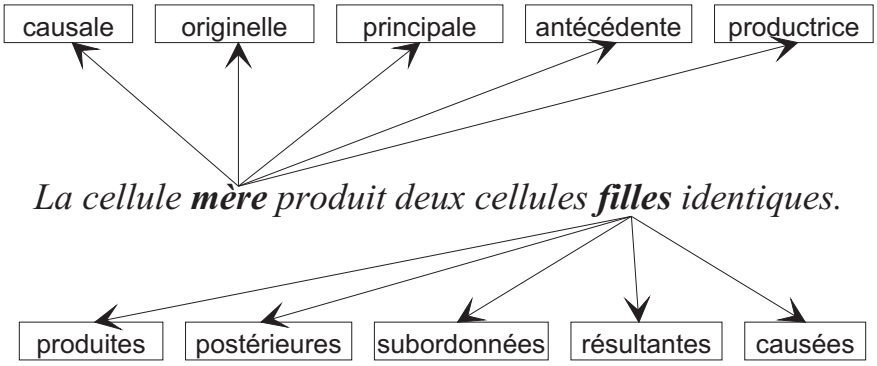


FIG. 2.2: Variations paradigmatiques pour *mère* et *fille*

Les mots qui figurent dans les cadres de la figure 2.2 correspondent à des synonymes locaux des mots *mère* et *fille* utilisés comme qualificatifs de *cellule*. Il est difficile de choisir lequel de ces mots est "le meilleur" synonyme du segment considéré. En fait, remplacer chacun des segments polysémiques par l'un de ses synonymes possibles conduit à produire un énoncé dont la signification est compatible avec celui de l'énoncé d'origine mais qui ne reflète plus qu'une partie de la signification originelle. Prise dans le contexte d'énonciation de cette phrase (description de la mitose destinée à un public non expert), la signification de chacun des synonymes que nous avons proposés est "contenue" dans la signification de l'énoncé tel qu'il est donné.

— *Le verbe est le **roi** de la phrase.*

Il s'agit d'un énoncé tiré d'un corpus oral (en fait un cours de linguistique). Si nous utilisons la même technique de variation du segment *roi* sur l'axe paradigmatique qu'avec l'exemple précédent, nous pouvons proposer une liste de synonymes candidats au remplacement :

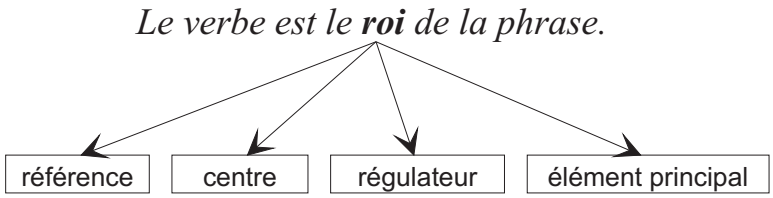


FIG. 2.3: Variations paradigmatiques pour *roi*

[1] Cet exemple, analysé par Prince dans [92], fait partie du corpus de test de son modèle EDGAR. Nous le reprenons pour pousser plus loin l'analyse des acceptions et emplois qu'il contient.

> Encore une fois, les significations de tous les énoncés produits en remplaçant le segment *roi* par les synonymes proposés constituent des aspects restreints de la signification de l'énoncé originel. Le sens grossièrement esquissé de cet énoncé est que le verbe est l'élément qui tient le rôle principal dans la phrase puisqu'il en régit le fonctionnement. Et cette charge sémantique importante se trouve concentrée dans l'utilisation du seul mot *roi* par référence implicite à tout ce que peut évoquer un personnage royal, à sa fonction dans l'état. On peut d'ailleurs remarquer que l'établissement de cette référence est vraisemblablement propre à la langue et à la culture française, car l'emploi du mot *roi* n'a certainement pas la même portée pour un locuteur anglais, par exemple, baigné dans une tradition de monarchie parlementaire, ou le souverain règne sans gouverner (ce qui semble corroboré par le fait que cet énoncé est tiré d'un cours portant sur les grammairiens de Port-Royal, donc Français vivant au XVIIe siècle).

Dans ces deux cas, la polysémie d'usage est mise en œuvre par le transfert de connaissances attachées à un domaine prototypique vers un nouveau domaine. On pourrait donc comparer ce processus à celui de l'analogie, à ceci près qu'on ne dispose pas ici de structure remarquable servant de guide au transfert des connaissances. En effet, les travaux portant sur l'analogie (cf. Robba [104]) font état de structures d'énonciation qui permettent de mettre en évidence un schéma de connaissances de référence, comportant des agents et des relations entre ces agents (comme *les planètes tournent autour du Soleil*), de nouveaux agents appartenant au domaine vers lequel on veut réaliser l'analogie (par exemple l'atome et ses électrons) et une structure comparative permettant de déclencher le transfert du schéma de référence en utilisant des traits communs entre les domaines source et cible : *les électrons tournent autour de l'atome comme les planètes autour du Soleil*. Ici, on dispose de beaucoup moins d'informations, ce qui aurait donc plutôt tendance à rapprocher le transfert d'usage des métaphores (cf. Ferrari [33]). Rappelons que la métaphore est un processus qui réalise le transfert d'au moins un trait sémantique d'un signifié source vers un signifié cible où il est absent, de façon à produire une nouvelle signification. Ainsi, dans le premier exemple, on peut considérer qu'il y a une métaphore de *mère* vers *cellule*, et dans le second de *roi* vers *verbe*. Cependant, le transfert qui est opéré entre la source et la cible concerne beaucoup plus qu'un (ou plusieurs) trait(s) sémantique(s). Il s'agit plutôt d'un vaste ensemble de références culturelles implicites tacitement partagées par l'émetteur et les récepteurs de l'énoncé. Et c'est sur ces aspects que nous voulons mettre l'accent tout particulièrement :

— les connaissances transférées sont dépendantes de la culture commune des locuteurs (c'est d'ailleurs pour cela qu'elles sont partagées) ;

— elles sont implicites, non exprimées linguistiquement.

Nous allons voir dans les paragraphes suivants que ces deux caractéristiques se retrouvent de façon homogène dans toutes les catégories de polysémie d'usage.

2.4.1.2 Cumuls d'emplois délibérés

Le premier exemple que nous avons donné pour illustrer la polysémie d'usage, *la chute de la maison Usher*, est un cas typique de cumul d'emplois. Il peut en effet arriver que l'émetteur d'un énoncé décide de jouer sur l'ambiguïté d'une expression afin de décrire une situation complexe dans un espace textuel limité. Cela arrive souvent dans les titres d'ouvrages, les slogans publicitaires et les coupures de presse.

Essayons d'évaluer les significations qui se cachent derrière ce titre, replacé dans son contexte (le contenu de la nouvelle). Cet énoncé peut être interprété ainsi :

— Comment la maison dans laquelle habitaient les Usher est tombée en ruine.

— Comment l'ensemble des membres de la famille Usher sont tombés dans la déchéance morale.

— La synchronicité / similarité / symétrie entre la chute morale des Usher et le délabrement du bâtiment qui les abrite.

Un titre de nouvelle ne peut se permettre d'énoncer toutes ces significations de façon explicite, et la solution trouvée par Edgar Poe est d'une grande élégance par sa concision et sa densité sémantique. Cette fois-ci encore, on a affaire à un transfert d'informations qui sont ajoutées principalement au signifié *chute* par le recours à des références culturelles implicites. Il s'agit notamment de références théologiques (*chute* au sens biblique) et de références historiques (comme, par exemple, *la chute de la maison des Valois*, sachant que l'éviction d'une famille régnante hors du pouvoir se faisait, il y a quelques siècles, dans des conditions suffisamment violentes pour que leurs bâtiments d'habitation, symbole de leur ancien pouvoir, soient également détruits). Ces références accompagnent implicitement l'acception de *chute* comme *dévalorisation* qui est employée ici. On peut également supposer l'établissement de la notion de synchronicité / similarité / symétrie entre ces deux événements par référence implicite au mode de raisonnement par abduction (i.e. l'instauration d'une relation de causalité entre deux objets ou événements proches dans l'espace ou dans le temps).

On trouve un autre exemple de cumul d'emplois dans cette citation d'un journal télévisé : *Les négociations ont repris entre la direction et les syndicats qui se sont réunis en table ronde*. L'expression *table ronde*, prise dans son acception de *négociation*[1] résume plusieurs significations simultanément : il s'agit à la fois de la description topologique du lieu où se passe cette réunion[2], du processus de négociation lui-même, et du protocole qui est associé à ce processus. Encore une fois, la pluralité des interprétations possibles pour cette même acception n'est explicable que par le recours implicite à des connaissances d'ordre culturel. L'acception de *table ronde* en tant que *négociation* est certes un fait établi dans la langue française, une trace figée d'une convention sociale, mais c'est une désignation sous-déterminée. On retrouve là en partie le principe d'indétermination (cf. §2.2.2) mais à un niveau qui n'est pas celui de la référence : l'acception *négociation* réfère clairement au processus de négociation. Il s'agit plutôt d'une indétermination d'usage : qu'est-ce qui est réellement pertinent, dans le cadre d'énonciation, parmi les nombreux éléments qui constituent une négociation ? Dans cet exemple, l'indétermination peut être restreinte par le contexte, notamment par les images contenues dans l'éventuel reportage associé au commentaire du présentateur. Mais s'il ne s'agit que d'un titre, ou d'une brève, toute liberté est laissée aux téléspectateurs de restreindre par eux-mêmes ou non l'indétermination d'usage, suivant leur propre culture personnelle (en fonction de ce qu'ils savent du déroulement d'une négociation entre le patronat et les syndicats).

On retiendra de ces exemples qu'il est crucial pour l'émetteur, dans le cas des cumuls d'emplois délibérés, de ne justement pas restreindre l'indétermination de l'énoncé. Cela lui permet d'émettre un maximum d'informations potentielles avec un nombre de signes linguistiques minimal. Tous les emplois possibles de l'acception considérée sont importants pour que la richesse informative du message soit préservée.

[1] Par opposition à la Table Ronde des chevaliers,ou à celle qu'on peut trouver dans un magasin d'ameublement.

[2] Même si les tables rondes ont plutôt tendance à se réaliser sous forme de plusieurs tables organisées en carré.

2.4.1.3 Mots jokers

Un autre cas de polysémie d'usage se rapproche encore plus du modèle de l'indétermination (cf. §2.2.2) : il s'agit de l'utilisation de "mots jokers". Ces mots ou expressions sont employés justement pour leur capacité à référer de manière sous-déterminée, que le mode soit transparent[1] ou opaque[2]. Les cas les plus évidents de mots jokers (tellement évidents qu'ils en deviennent caricaturaux) sont les fameux *truc*, *machin* et autres *bidules* qui sont fréquemment employés dans le langage quotidien. Nous les qualifions de caricaturaux car ils atteignent un tel degré d'indétermination qu'on ne peut même plus repérer de sens ni d'acceptions pour ces mots[3]. Ces mots peuvent référer à n'importe quel objet, ce qui les fait sortir du cadre strict de la polysémie d'usage.

Cependant, lorsqu'on ne s'intéresse plus au langage quotidien mais aux langages de spécialité, i.e. aux sous-langages propres à des communautés de locuteurs fédérés autour d'une activité commune (possédant leur propre lexique spécialisé, et parfois même une syntaxe particulière), on rencontre des mots ou des expressions fortement polysémiques (soumis à la polysémie d'usage) parce qu'ils sont utilisés pour signifier hors de leur cadre de définition originel. Il s'agit d'une "défaite" des tentatives de normalisation lexicale auxquelles tendent tous les spécialistes des divers domaines. Le propre d'un langage de spécialité est en effet de disposer d'un lexique particulier constitué en *nomenclature*. Une nomenclature est le résultat d'une construction délibérée qui a pour but de proposer un lexique univoque : les *termes*. Un terme est censé posséder une définition consensuelle très précise et rigoureuse, de telle sorte que tous les membres d'un même domaine de spécialité aient tous la même intention significatrice lorsqu'ils l'emploient. Cela implique que tous les termes d'une nomenclature sont absolument monosémiques. Cette recherche d'univocité lexicale est tout à fait légitime dans le cadre des langages de spécialité qui correspondent généralement à des domaines techniques ou scientifiques, où l'on a besoin d'une grande rigueur, et donc d'éviter les ambiguïtés. On constate cependant un échec de cette entreprise : certains termes sont si fréquemment employés au sein d'une communauté linguistique de spécialistes que leur valeur sémantique devient trop générique pour qu'on puisse déterminer clairement quel(s) usage(s) sont concrètement mis en jeu au sein de certains énoncés.

C'est notamment le cas du terme *système* en informatique. Un système, au sens large, désigne un ensemble de modules programmés dont l'interaction a pour résultat le fonctionnement dudit système. Originellement, *système* était utilisé comme élision de *système d'exploitation*[4]. Cependant, avec l'arrivée de la programmation modulaire distribuée, le terme *système* est également attribué aux logiciels composés de plusieurs programmes en interaction, ainsi souvent qu'à leurs modèles abstraits. Cette confusion est encore aggravée par la popularisation des techniques permettant d'interfacer les *systèmes* (modulaires) avec le *système* (d'exploitation), ce qui peut rendre pour le moins ardue la compréhension d'expressions telles que *les classes du système* ou *déclaré au sein du système* : le lecteur doit retrouver s'il s'agit du système modulaire, du système d'exploitation, voire des deux à la fois.

Un cas similaire est celui du terme *marché* en économie et dans les milieux financiers. Il peut désigner aussi bien l'ensemble des processus d'échanges de capitaux qu'une institution nationale (ou internationale) régissant ces processus, les résultats de ces échanges, les acteurs économiques qui y participent ou même "l'état d'esprit" de ces acteurs. Bien entendu, suivant les énoncés, il n'est pas toujours nécessaire de spécifier complètement la signification d'une occurrence donnée de *marché* :

[1] Evocation d'un objet particulier du monde de référence.

[2] Evocation d'une classe d'objets par une propriété saillante qu'ils partagent.

[3] Excepté le mot *truc* qui possède une acception non générique : savoir-faire, procédé ou astuce (d'après le Petit Larousse).

[4] L'ensemble logiciel minimal qui permet de piloter l'électronique d'un ordinateur et d'autoriser des interactions avec un utilisateur.

par exemple, dans *le marché est dynamique aujourd'hui*, le dynamisme des acteurs économiques est intimement relié au rythme des transactions de capitaux, et aux créations ou destructions de richesses qui en découlent. Toutes ces interprétations participent à la signification de l'énoncé.

Au contraire des deux catégories précédentes de polysémie d'usage (les transferts d'usage et les cumuls d'emplois), la polysémie d'usage n'est pas "nécessaire" dans le cas des mots jokers. A l'intérieur d'un lexique fortement structuré par des définitions strictes, il est toujours possible de revenir à ces définitions et de modifier le contexte d'énonciation de façon à spécifier exactement quel(s) emploi(s) est(sont) utilisé(s) dans un énoncé donné. En effet, le caractère technique ou scientifique d'un domaine de spécialité l'affranchit des contraintes esthétiques. Cependant, on constate que les émetteurs spécialistes s'abstiennent le plus souvent de procéder à cette spécification. Ils s'épargnent une fraction non négligeable du "coût cognitif" qu'elle engendrerait pour eux et la laissent à la charge des récepteurs, en contractant leurs énoncés, ce qui est source d'ambiguïté. Pour un plus long développement sur ce principe d'économie cognitive, et les perturbations des messages qu'il est susceptible de produire, on pourra consulter Prince [91, section 4.1.3]. Ces perturbations peuvent s'exprimer de façons différentes, mais dans le cas qui nous intéresse, par l'entremise de la polysémie d'usage que nous résumerons ainsi : l'attitude "égoïste[1]" des émetteurs les conduit à sous-spécifier les emplois d'un mot joker dans un énoncé, de telle sorte que tous les emplois pertinents sont évoqués simultanément sans qu'on en sache plus sur les relations qu'ils entretiennent. A charge pour les récepteurs de produire les inférences nécessaires grâce aux références culturelles (du domaine de spécialité) dont ils disposent.

2.4.1.4 Jeux de langage

Les jeux de langage sont des astuces stylistiques destinées à provoquer l'amusement de leurs récepteurs. La polysémie d'usage peut contribuer à un tel résultat, qu'il s'agisse de jeux de mots, de traits d'esprit ironiques ou d'expressions familières. L'aspect comique provient alors de la collision entre différents emplois d'une unité lexicale, ou par le contraste entre l'un des emplois évoqués et le style rhétorique de l'énoncé dans lequel l'unité lexicale s'insère.

Voyons cela sur un exemple de jeux de mots[2] : lors d'un repas, l'hôtesse amène sur la table une tarte qu'elle a confectionnée elle-même. Tandis que les convives la complimentent sur l'aspect de sa pâtisserie, l'un d'eux s'exclame ironiquement : « Tu es vraiment la reine des tartes ! » Il s'agit d'une occurrence de polysémie d'usage : l'expression polysémique qui est en cause ici est le mot *reine* dans son acception de *la plus...* Les deux interprétations possibles pour cet énoncé sont :

i. Tu es la meilleure pour ce qui est de confectionner des tartes.

ii. Tu es la plus stupide.

L'acception *la plus...* est instanciée en *la plus* + *bonne* dans le premier cas et en *la plus* + *stupide* dans le deuxième cas. L'adjectif *bonne* de la première interprétation est, si nous nous plaçons dans le cadre de la sémantique différentielle, un trait sémantique propre ajouté (nous réutilisons ici le vocabulaire que nous avons donné p. 26) du signifié *reine* : un roi (ou une reine) est, sauf mention du contraire, bon. Au contraire, l'adjectif *stupide* correspond à un des sens du mot *tarte*. Il s'agit donc bien de la même acception de *reine* dans les deux interprétations, mais avec deux emplois différents : mélioratif dans le premier cas et péjoratif dans le second. Il nous paraît en outre clair que cette opposition mélioratif/péjoratif est un

[1] Sans jugement de valeur, il s'agit du vocabulaire employé par Prince (*op. cit.*).

[2] Pas forcément subtil ni désopilant, mais extrait d'une situation réelle.

trait sémantique propagé par le contexte culturel, et non pas par un figement d'une norme sociale dans la langue, puisque l'apparition de la connotation péjorative est due à une possibilité d'emploi ironique des signifiés *roi* et *reine*. Nous sommes donc bien dans le cadre de la polysémie d'usage, et nous noterons d'ailleurs que, comme nous l'avions annoncé précédemment, la polysémie d'usage concernant le mot *reine* n'est possible que grâce à la polysémie fonctionnelle dont est affecté le mot *tarte* dans cet énoncé.

On retrouve l'ironie à l'œuvre dans cet autre exemple emprunté à Rastier [100, p. 92, note de bas de page n°2)] :

> *« Bien des auteurs se disent modernes et ne sont que tardifs. »*

Il s'agit ici d'une polysémie d'usage concernant l'adjectif *modernes* qui est affecté par la présence de l'adjectif *tardifs*. Tous deux sont à prendre dans l'acception de *récents*. L'adjectif *modernes* possède une interprétation neutre, équivalente à *récents*, et une interprétation méliorative *récents* + *rénovateurs* qui est contredite par le caractère péjoratif de *tardifs* qui s'interprète comme *récents* + *en retard*. C'est bien là un cas de polysémie d'usage étant donné le contexte dans lequel cette phrase apparaît : un ouvrage scientifique de linguistique. Dans ce cadre, la langue académique impose un usage neutre de *moderne*, i.e. sans le caractère *rénovateur* que lui confère la langue quotidienne. Il ne s'agit donc pas d'un cas de polysémie par présence de deux acceptions *récents* vs *rénovateurs*, mais bien par présence de deux emplois *récents* vs *récents* + *rénovateurs* en raison de l'emprunt que fait l'auteur du deuxième emploi à un registre de langue qui n'est pas celui du reste de son discours.

Dans ces deux exemples, l'amusement visé par le jeu de langage est atteint par la présence simultanée de deux emplois d'une même acception de l'expression polysémique. Cette pluralité des emplois n'est pas nécessaire à la construction de la signification de l'énoncé, mais à son aspect ludique. Alors que dans les trois précédentes catégories de polysémie d'usage, la présence simultanée de plusieurs emplois contribue à la formation d'une signification plus riche, ici elle conduit à construire une signification globale incohérente, éclatée, et c'est ce qui provoque l'amusement. Parallèlement, l'aspect culturel des connaissances nécessaires pour détecter et interpréter la polysémie d'usage se fait nettement sentir ici aussi. Dans les deux exemples que nous venons de donner, elles dépendent effectivement des tics de langage en usage à un moment donné dans la société, de la façon dont les locuteurs "déforment" inconsciemment la définition originelle de certains mots pour refléter l'esprit du moment.

Enfin, une dernière catégorie de jeux de langage se prête également à la polysémie d'usage : il s'agit d'un grand nombre d'expressions familières puisant dans le vocabulaire des sens. Techniquement, on peut considérer que ces expressions correspondent à des transferts d'usages particuliers entre le domaine sensoriel et des domaines émotionnels plus abstraits, sauf que c'est ici, comme dans les autres jeux de langage, la discontinuité entre les emplois qui est significative, au contraire des transferts d'usages proprement dits qui fonctionnent à partir de la continuité des différents emplois. On prendra à titre d'exemples des expressions telles que *je ne peux pas le sentir* ou *j'en ai plein le dos*. Tout comme pour les transferts d'usage (cf. §2.4.1.1), on se situe à la limite de la métaphore : il y a création d'un sens nouveau par transfert d'usage d'un domaine vers un autre domaine imprévu. Et encore une fois, il s'agit en fait du transfert d'un vaste ensemble de connaissances, certaines "encyclopédiques" (communes) et d'autres propres à chaque locuteur. Dans l'exemple *j'en ai plein le dos*, il y a un résumé d'emplois tels que *fardeau*, *douleur dans le dos* et *fatigue* qui sont transférés sur une description de l'état émotionnel du locuteur par une comparaison implicite esprit/corps. C'est cette opposition tranchée (clairement voulue ?) entre le domaine source et le domaine cible qui caractérise selon nous ce genre d'expressions et les distingue des autres formes de transfert d'usage. Il y a discontinuité nécessaire entre les différents emplois en présence.

<table>
<tr><th></th><th>Aspects communs</th><th>Spécificités</th><th>Continuité sémantique</th></tr>
<tr><td>Transfert d'usage</td><td rowspan="4">Il y plusieurs emplois présents simultanément.

Les emplois sont définis par de nombreuses références dépendantes de la culture des locuteurs.

Ces références sont implicites, non exprimées.</td><td>Les emplois sont transférés d'un domaine de référence vers un nouveau domaine d'application imprévu.</td><td rowspan="3">Il y a une continuité sémantique entre ces emplois.</td></tr>
<tr><td>Cumul d'emplois</td><td>La multiplicité des emplois est nécessaire.

Il est important de ne pas restreindre l'indétermination.</td></tr>
<tr><td>Mot joker</td><td>Une acception très générique correspond à plusieurs emplois particuliers.

Il est possible de spécifier le(s) emploi(s) visé(s).</td></tr>
<tr><td>Jeu de langage</td><td>La multiplicité des emplois est nécessaire.</td><td>Il y a discontinuité sémantique, voire franche opposition, entre les différents emplois.</td></tr>
</table>

TAB. 2.1: Caractérisation des diverses catégories de polysémie d'usage

2.4.2 Caractérisation

Pour tenter de synthétiser les différents exemples de polysémie d'usage que nous avons présentés dans cette section, nous proposons de récapituler les éléments communs et les différences entre les différentes catégories de polysémie d'usage.

Le tableau 2.1 (p.38) permet de consulter cette synthèse.

2.4.3 Faut-il résoudre l'ambiguïté ?

Résolution des ambiguïtés

Une caractéristique commune des différentes formes de polysémie d'usage qui ressort du tableau 2.1 est la présence *simultanée*, parfois obligatoire, de plusieurs emplois dans l'interprétation d'une occurrence polysémique en contexte. C'est une propriété qui est particulièrement importante, car elle va à l'encontre des principes les plus fondamentaux du traitement des ambiguïtés lexicales en intelligence artificielle. A titre d'illustration, il suffit de considérer le titre de l'ouvrage de référence *Lexical Ambiguity Resolution* (Small *et al.* [115]). Ce titre exprime bien l'attitude qui est universellement adoptée : les ambiguïtés lexicales constituent un *problème* qu'il faut *résoudre*. Sous-entendu que comprendre un texte, c'est arriver à attribuer un unique concept ou rôle sémantique à chaque segment autonome du texte,

de manière à *composer* ces concepts et ces rôles afin de produire *la* représentation conceptuelle du texte analysé.

On retrouve là les hypothèses implicites qui sont à la base du modèle "homonymie" (cf. §2.2.1) : le langage a pour vocation d'établir des références entre ce qui est communiqué et des choses réelles ou des états du monde, et pour mener cette tâche à bien, il se doit d'être univoque dans son essence la plus profonde, c'est-à-dire que par-delà les accidents du discours, on a bien une relation univoque des mots vers les éléments du monde. Avec une telle conception du langage, les ambiguïtés lexicales posent un réel problème : ce sont des phénomènes perturbateurs qui obscurcissent le rapport du mot à la chose. Cet obscurcissement prend la forme d'une *sous-détermination* de la relation mot $\rightarrow$ chose ou mot $\rightarrow$ concept en une relation mot $\rightarrow$ chose$_1$, ..., chose$_n$ ou mot $\rightarrow$ concept$_1$, ..., concept$_n$, c'est-à-dire que la relation dont on dispose n'est pas suffisante pour déterminer exactement quel mot ou quel concept est corrélé au mot. On comprend alors que le seul traitement des ambiguïtés lexicales qui soit envisagé soit leur *résolution*, i.e. la recherche d'une détermination complémentaire des relations lexicales afin de connaître précisément la chose ou le concept qui correspond au mot polysémique.

Cette stratégie est tout à fait adaptée au traitement de la polysémie fonctionnelle, la seule forme de polysémie envisagée classiquement en traitement automatique des langues naturelles. Car en fait, la polysémie fonctionnelle est plus une polysémie "de principe" et s'apparente plutôt à l'homonymie. Les exemples les plus flagrants sont ceux des unités lexicales qui correspondent à des signifiés ayant des natures paradigmatiques différentes, comme *couvent*, qui est soit un nom commun, soit une forme fléchie du verbe *couver* (d'où l'inénarrable *les poules du couvent couvent*). Il est impératif, dans ce cas, de trouver l'unique signifié acceptable pour une occurrence donnée, et ceci en suivant tout simplement des règles grammaticales. Ainsi, dans notre exemple, la phrase commence par un syntagme nominal suivi de *du*, ce qui implique qu'on attend après *du* un nom au génitif ; par conséquent, la forme *couvent* correspond forcément au nom commun. De plus, *les poules du couvent* forme un groupe nominal autonome, donc on attend ensuite un verbe plutôt qu'un autre nom commun, ce qui permet d'identifier immédiatement la deuxième forme de *couvent* avec le verbe *couver*. Des règles semblables sont applicables aux mots dont plusieurs formes fléchies ont la même orthographe (par exemple, *je fais, tu fais*).

Pour ce qui est des occurrences qui correspondent à des signifiés de même nature paradigmatique, comme *plateau* (lieu géographique ou ustensile), il est également indispensable de retrouver le bon signifié. Les règles qui permettent de lever l'ambiguïté sont cette fois plus élaborées, puisqu'elles requièrent de pouvoir manipuler des cas profonds (ici *locatif* et *ergatif*, par exemple) ainsi que des traits sémantiques. Il est même envisageable d'avoir recours à des informations plus pragmatiques (lorsque le texte analysé dépasse la simple phrase prise hors contexte), par exemple par l'intermédiaire d'un gestionnaire de thèmes ou d'un gestionnaire de personnages, comme dans le système CARAMEL-1 [107, 108]. Dans tous les cas, grâce à des règles de filtrage plus ou moins complexes, utilisant des informations plus ou moins variées, il s'agit de retrouver l'unique signifié qui est compatible avec le contexte linguistique de l'occurrence.

Au contraire, cette stratégie nous paraît inadaptée au traitement de la polysémie d'usage. Dans tous les exemples de polysémie d'usage que nous avons vus, il n'est en effet pas possible de supprimer un des usages attachés à une occurrence sans perdre de l'information pertinente. Ainsi dans l'exemple que nous avons cité p. 32 : *le verbe est le roi de la phrase*, tous les emplois de *roi* (*référence, centre, régulateur, élément principal*) constituent autant d'aspects significatifs de la *signification* du mot *roi* dans cette phrase. Tous sont pertinents simultanément, et tous sont subtilement différents les uns des autres. En supprimer un revient à ne plus considérer une des facettes du rôle du verbe, alors qu'elle était pourtant (vraisemblablement) présente dans l'intention de l'émetteur. Une stratégie visant à ne sélectionner qu'un

seul emploi et à éliminer les autres nuit à l'exhaustivité de l'interprétation : on ne retient plus qu'un seul aspect pour décrire une situation bien plus complexe. De même, en ce qui concerne l'exemple de la p. 33 : *la chute de la maison Usher*, choisir une seule des deux interprétations possibles supprime tout un pan de l'éclairage que ce titre peut apporter sur la nouvelle à laquelle il se rapporte. C'est encore plus flagrant dans le cas des jeux de langage : une interprétation univoque d'un jeu de mots lui enlève tout son sel.

Or, les modèles et les systèmes généralement proposés en traitement automatique des langues naturelles ne mettent en œuvre que la stratégie de sélection d'un unique signifié candidat. Le critère qu'ils préconisent dans les cas que nous venons de rappeler consiste immanquablement à effectuer un *choix arbitraire* entre les différentes propositions. En effet, comment choisir autrement qu'arbitrairement entre plusieurs interprétations qui sont toutes valides ? On pourrait se demander si les applications actuelles nécessitent réellement un tel niveau d'analyse sémantique. Cependant, minimiser la représentation sémantique des signifiés jusqu'à la perte d'information est envisageable à condition que le texte dans lequel s'insère le segment qui a été ainsi "tronqué" ne fasse pas de référence à ses différents usages. Dans le cas contraire, on se retrouve face à des inférences impossibles à effectuer, ce qui conduit le système à rejeter des phrases ultérieures (qui contiennent notamment des anaphores) alors qu'elles sont tout à fait valides. Par exemple, dans le cas du mot joker *marché*, la suite de phrases suivante sera rejetée : *Je vais au marché. Une fois que je l'aurai fini, j'irai en ville*. Dans la première phrase, le contexte locatif implique de sélectionner un signifié *marché*-lieu (dans le meilleur des cas), alors que le contexte perfectif de la seconde phrase implique que l'anaphore *l'* doit se rattacher à un antécédent qui est un processus, introuvable ici.

Conservation de l'ambiguïté dans les traitements

Cela dit, l'approche proposée par Prince [92], préconisant l'emploi de structures de représentations et d'algorithmes qui *ne choisissent pas* parmi les différentes variantes sémantiques des signifiés, nous semble incontournable pour arriver à créer des systèmes informatiques permettant de réaliser un traitement de la polysémie d'usage qui ne soit ni artificiel, ni réducteur.

Dollan [28] et Vanderwende [128], qui travaillent tous deux sur l'acquisition d'information lexicale à partir de dictionnaires électroniques, adoptent la même position. Nous retiendrons tout particulièrement l'argumentation de Dollan lorsqu'il considère l'emploi des définitions des dictionnaires comme source de représentation des signifiés dans le cadre de la désambiguïsation lexicale (*word sense disambiguation*) en partant, tout comme lui, de l'exemple du mot *stalk*[1] dont la définition dans le LDOCE[2] comprend deux entrées :

i. « The main upright part of a plant (not a tree) (Ex : *"a beanstalk"*) »

ii. « A long narrow part of a plant, supporting one or more leaves, fruits, or flowers ; stem. »

Les différences qui existent entre ces deux définitions sont suffisamment subtiles pour paraître superflues dans bon nombre de tâches, y compris des tâches de désambiguïsation lexicale. Ainsi, il n'est pas nécessaire de distinguer les deux interprétations de *stalk* dans une phrase comme *The stalks remained in the farmer's field long after summer*[3].

[1] En français *tige*.

[2] Longman's Dictionary of Contemporary English

[3] (Litt.) *Les tiges sont restées dans le champ du fermier longtemps après l'été.* En français, on dira plus volontiers *les chaumes* que *les tiges* dans un tel contexte agricole.

L'approche classique (basée sur une présomption de polysémie fonctionnelle) de désambiguïsation du mot *stalks* consistera à déterminer laquelle de ces deux définitions s'applique pour cette occurrence. Il y aura forcément un choix arbitraire dans ce cas, puisqu'aucune des deux n'est meilleure que l'autre. Cela impliquera une perte d'information puisqu'avec une telle stratégie, la quantité d'information sémantique disponible pour ce mot sera limitée à ce qui est présent dans la signification sélectionnée, et du coup, certaines informations ou traits sémantiques "accompagnants" seront perdus alors qu'ils pourraient être utiles pour un traitement moins local. Ici, choisir la première définition entraîne la perte de la relation de synonymie entre *stalk* et *stem*, ainsi que des traits sémantiques /narrow/ et /supports leaves, fruits, or flowers/. Sélectionner la seconde définition revient à perdre le trait sémantique /upright/, l'instanciation de *stalk* en *beanstalk* ainsi que l'exception annonçant que la partie principale dressée d'un arbre ne peut pas être étiquetée par *stalk*. Dollan précise que les lecteurs humains ne font pas face à ce genre de problème car ils ne traiteraient pas la définition de *stalk* comme une paire d'entrée discrètes. Un humain aura plutôt tendance à se constituer une représentation de la signification de *stalk* comme une seule entité, soit plus abstraite que les deux entrées, soit qui reprend les informations figurant dans les deux définitions.

A partir de ces remarques, Dollan propose d'utiliser des représentations sémantiques, pour ce genre d'éléments du lexique, qui regroupent les différentes sous-entrées figurant dans le dictionnaire. Il appelle ces représentations des *clusters* (ou groupes). Il ne s'agit pas de linéariser brutalement toutes les définitions du dictionnaire qui se décomposent en sous-entrées : les clusters regroupent, pour une même entrée lexicale, les sous-entrées qui donnent lieu à un choix arbitraire, i.e. celles qui sont très intimement reliées les unes aux autres (ce qui correspond en fait à ce que nous avons appelé les *emplois* d'une même acception). Il propose un système automatique de création des clusters par exploration itérative d'un dictionnaire, grâce à la méthode d'auto-amorçage (*boot-strapping*) décrite dans Vanderwende (*op. cit.*). Pour résumer grossièrement cette méthode, il s'agit de partir d'une base de données syntaxico-sémantique extrêmement réduite qui, appliquée au contenu du dictionnaire, permet d'acquérir des informations sémantiques supplémentaires par filtrage des définitions. Puis, lors des multiples passes successives, les informations sémantiques disponibles à l'étape n permettent d'acquérir de nouvelles informations à l'étape $n + 1$.

L'approche de Dollan est donc similaire à celle de Prince dans le sens où il considère comme inacceptable le fait d'avoir à choisir entre différents emplois d'une acception, et donc de perdre des informations sémantiques éventuellement utiles. Cependant, elle présente une différence majeure : il vise en fait à éliminer le processus de désambiguïsation lexicale en replaçant le traitement de l'ambiguïté dans une phase antérieure : lors de la constitution du lexique. Il s'agit de réunifier les acceptions d'un mot lorsque celles-ci ont été déclinées en emplois par les définitions du dictionnaire. Nous pensons cependant que la distinction entre les différents emplois d'une acception ne doit pas être ainsi atténuée car ils correspondent à des réalités de la langue. Nous souhaitons plutôt constituer des structures de représentation des signifiés lexicaux qui permettent à la fois de *regrouper* les différents emplois (afin de préserver tous ceux qui sont présents dans l'interprétation d'une occurrence du mot polysémique), et de les *distinguer* en fonction de l'éclairage particulier que chacun d'entre eux apporte sur la signification de cette occurrence, à quel point ils sont individuellement pertinents dans ce contexte, dans quelle mesure ils participent à la construction de la signification de l'occurrence.

2.5 Cadres sémantiques théoriques

La recherche en traitement automatique des langues naturelles fait partie du projet qu'a attribué le cognitivisme orthodoxe à l'intelligence artificielle : reproduire sur machine les processus mentaux des humains. Le but du traitement automatique des langues naturelles est donc à l'origine de reproduire la compétence linguistique, en se basant toujours sur le postulat énoncé par le cognitivisme orthodoxe fonctionnaliste, à savoir que les processus sont indépendants de leurs substrats, et par conséquent que les processus mentaux sont computationnels. Même si ce postulat est aujourd'hui attaqué par les mêmes qui l'avaient autrefois formulé, comme par exemple Putnam [96], il n'en a pas moins imprimé aux conceptions utilisées en traitement automatique des langues naturelles une inertie qui rend difficiles les changements de directions théoriques.

Or, dans notre activité d'étude et de modélisation de la polysémie d'usage, nous avons besoin de rompre avec ce cadre traditionnel. Afin de prendre la mesure de cette nécessité, ainsi que de la nature des changements dont nous avons besoin, nous allons réexaminer rapidement, mais clairement, les théories classiques de la sémantique en linguistique.

2.5.1 Paradigmes sémantiques classiques

Nous aborderons tout d'abord le paradigme référentiel et le paradigme inférentiel, qui sont les deux familles théoriques dans lesquelles on peut ranger pratiquement tous les travaux qui ont été menés en traitement automatique des langues naturelles (et dans les recherches cognitives en général) ces cinquante dernières années. Cela nous permettra en outre d'exposer certains problèmes inhérents à ces deux paradigmes, problèmes qui peuvent être rédhibitoires lorsqu'il s'agit de modéliser la polysémie d'usage.

La définition traditionnelle de la signification se base sur celle du *signe* linguistique. Or, il ne faut pas oublier que les signes linguistiques sont définis arbitrairement, alors que la signification semble être tout sauf arbitraire. Rappelons à cet égard deux propriétés distinctives des signes :

— Il existe différentes catégories de signes linguistiques, qui semblent difficilement interchangeables. Par exemple, le point d'interrogation dans *Tu viens ?* a la même fonction que la particule interrogative dans *Est-ce que tu viens ?*, mais ces deux signes ne sont pas équivalents.

— Un signe en lui-même est un outil construit par l'homme, sans valeur propre. De plus, il ne fonctionne pas indépendamment des autres signes, au contraire des symboles qui symbolisent, eux, dans l'absolu. Le signe, au contraire, ne dénote pas hors contexte.

Rastier [101], dans une démarche de récapitulation des théories linguistiques, dénombre quatre conceptions du signe qui correspondent à autant d'approches de la signification. Nous ne citerons cependant ici les deux premières qu'à titre d'illustration, et nous nous attarderons plus longuement sur les deux dernières, qui sont celles dominantes en traitement automatique des langues naturelles.

— *L'icône* est un signe qui représente analogiquement son référent de manière "directe et naturelle". Cependant, on constate généralement qu'il s'agit d'une analogie conventionnelle : une main, une tête, etc. sont toujours représentées de la même façon. C'est un type de représentation qu'on retrouve en sémantique cognitive (par exemple chez Lakoff [61] ou chez Langacker [62]) sous forme de métalangage graphique. Elle repose sur l'hypothèse que la cognition est une sorte d'extension

abstraite des facultés visuelles. Cependant, les langues telles qu'elles sont parlées et écrites ne contiennent pas de signes iconiques (même les langues à écriture idéographique ont tellement chargé les signes de conventions que leur aspect analogique est très largement discutable, cf. le chinois).

— *L'index* est un signe qui n'a de sens que relativement à une situation de communication réelle ou évoquée par un texte. D'après l'ethnométhodologie (cf. Garfinkel [41]), toutes les langues, mais aussi les comportements extralinguistiques sont pour une part dans ce cas : tout acte, langagier ou comportemental, ne prend pleinement son sens qu'en référence à une situation. C'est le principe de l'indexicalité. Pourtant, les langues ne contiennent pas d'index purs, seulement des signes indexicaux (comme les démonstratifs ou les déictiques) qui sont, selon la sémantique interprétative, arbitrairement séparés du régime commun des autres signes puisque, de toute façon, le sens d'un signe lui est conféré par le contexte, linguistique et extralinguistique.

— *Le symbole*, qui relève de la "triade Aristotélicienne" ;

— *L'indice*, qui est le support d'une inférence.

Nous allons aborder plus en détail ces deux dernières catégories de signes, et les paradigmes sémantiques qui en découlent.

2.5.1.1 La sémantique référentielle

Il s'agit du modèle de la signification qui est certainement le plus important historiquement, et *a fortiori* le plus répandu. Il remonte à Aristote [2] et se perpétue de nos jours sous le nom de *triangle sémiotique* : ce nom lui est dû à sa représentation graphique la plus commune, qui tente d'expliciter les relations existant entre les mots, les concepts et les choses.

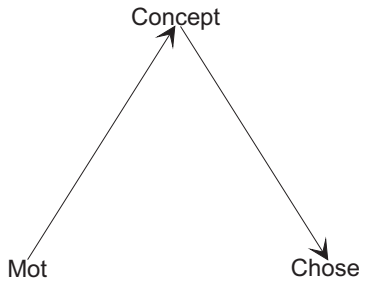


FIG. 2.4: Triangle sémiotique

Cette conception se base sur l'invariabilité des choses décrites opposée à la variabilité du discours : même si les paroles changent, nous décrivons tous la même chose, mais par des moyens d'expression différents. Cette triade, qui fonde l'universalisme du langage, a été reprise par les scolastiques au Moyen-Âge, par les grammaires générales du siècle des Lumières, et on la retrouve dans le monde anglo-saxon avec Ogden et Richards [79] et Lyons [68].

Cette conception a été récusée par la linguistique saussurienne depuis le début du siècle par l'introduction de la notion de signifié, mais elle reste extrêmement vivace en philosophie du langage, et par voie de conséquence, elle règne presque intégralement sur les programmes de recherche sur la cognition, et plus particulièrement sur le traitement automatique des langues naturelles (cf. Johnson-Laird [54]) : l'objectif est de « montrer comment le langage se rapporte au monde par l'intermédiaire de l'esprit. »

Remise en cause du triangle sémiotique

Parmi les chercheurs concernés par la cognition, rares sont ceux à avoir contesté le triangle sémiotique. On pourra cependant noter Putnam [96] qui réfute les trois thèses suivantes :

1. « Chaque mot employé est associé dans l'esprit du locuteur avec une certaine représentation mentale.
2. Deux mots sont synonymes (ont le même sens) seulement dans le cas où ils sont associés avec la *même* représentation mentale par les locuteurs qui les emploient.
3. La représentation mentale détermine ce à quoi le mot réfère, s'il réfère à quelque chose. »

où il donne le sens de concept à *représentation mentale*. L'insurrection de Putnam concerne principalement le postulat de l'existence d'un langage de la pensée universel et indépendant des langues : le *mentalais* de Fodor [35]. Ce dernier suppose en effet qu'il existe une forme d'expression interne à l'esprit humain, innée et commune à tous les humains. Comprendre un texte reviendrait alors à le traduire dans ce langage mental, et à faire des inférences sur ces représentations internes pour déduire de nouvelles propriétés. Deux mots dans deux langues différentes qui se traduisent par le même symbole mental (le même concept) ont le même sens. Putnam conteste ce langage mental, et propose des « représentations mentales de surface », les seules dont nous puissions avoir conscience. Du coup, ces représentations mentales ne sont plus indépendantes des langues : il cite comme exemple la distinction entre le français *orme* et l'anglais *elm* qui correspondent respectivement aux représentations de surface *arbre qu'on appelle "orme"* et *tree that one calls an "elm"*.

Sémantique véric onditionnelle et sémantique extensionnelle

Ce paradigme fonde la *sémantique référentielle* qui se décline sous deux aspects, que nous appellerons respectivement la sémantique *véric onditionnelle* et la sémantique *extensionnelle*.

— La sémantique véric onditionnelle s'attache à étudier comment le langage exprime la vérité, conçue comme une adéquation du monde et de l'intellect. L'activité principale de la compréhension peut se résumer à l'attribution de *valeurs de vérité* aux différents constituants des énoncés, de façon à accepter ceux qui s'évaluent à VRAI et à rejeter ceux qui s'évaluent à FAUX. Cette attribution de valeurs booléennes est régulée par des *conditions de vérité* qui représentent plus ou moins des connaissances sur le monde.

  Tout cela repose sur l'hypothèse que le sens d'un texte peut être représenté par un langage formel, du style du calcul des prédicats du premier ordre, comportant au moins quatre types de signes :

  1. des *symboles terminaux* qui désignent directement un objet du monde réel ;
  2. des *variables* qui n'ont pas référence (la "compréhension" du texte a donc pour objectif d'unifier les variables avec des symboles terminaux, seule façon de pouvoir évaluer la véracité des énoncés) ;
  3. des *prédicats* permettant de représenter les interactions entre les choses (i.e. les actions, les procédures et les états de choses) ;
  4. et des *quantificateurs* qui servent à modifier les relations syntaxiques entre les prédicats et les terminaux/variables, du genre *pour un/pour tous*, *nécessairement/éventuellement*, etc.

Il s'agit d'une manifestation particulière du fameux langage mental de Fodor, sous la forme universellement répandue de l'écriture logique. Il est amusant de constater que cette habitude d'écrire le sens des énoncés sous forme logique s'enracine dans une très longue tradition... métaphysique ! On retrouve en effet cette idée de langage intérieur chez Platon (dans *Le Sophiste*), et deux courants d'idée s'affrontent historiquement à son propos : soit il varie avec les langues, soit il en est indépendant. Saint Augustin, puis Thomas d'Aquin, tentent de concilier ces deux conceptions en scindant le langage intérieur en deux : un verbe du cœur qui n'appartient à aucune langue (c'est le langage de l'âme, issu du divin), et un verbe intérieur qui varie suivant les langues. Fodor, avec son mentalais, refuse la synthèse augustinienne et se place en droite ligne de l'héritage platonicien et néoplatonicien. En effet, la variabilité des signifiés viendrait remettre en cause le postulat d'universalité des concepts (qui sont, dans cette tradition, des formes pures, libérées des astreintes matérielles). Au contraire, Putnam se place plutôt dans une optique stoïcienne avec ses représentations de surface qui sont une reformulation moderne du verbe intérieur.

A la base de cette dialectique, il y a l'idée que "le langage doit dire le vrai", et l'on cherche donc de tous bords à trier les énoncés faux des énoncés vrais. Outre qu'il nous semble surprenant d'imposer ainsi une finalité éthique au langage, cette recherche de la vérité implique qu'on doive se référer à une description du vrai enregistrée quelque part dans l'univers. C'est ce que fait la sémantique vériconditionnelle en recourant aux concepts comme reflets universels de la réalité. Cela implique que nous advenons dans un monde de sens déjà donné : il existe "quelque part" des objets non matériels qui sont la référence ultime en matière de réalité.

Ce postulat nous semble trop fort, et trop sujet à la subjectivité, pour constituer la base de notre propre étude systématique de la signification dans ses aspects observables de polysémie d'usage et d'acception.

— Ce que nous nommons "sémantique extensionnelle" est généralement dénommé de façon générique comme "sémantique référentielle" ou "sémantique cognitive". C'est une sémantique qui cherche avant tout à assigner une *extension* aux signes linguistiques. Rappelons tout d'abord ce que sont l'extension et l'intension d'une classe d'objets :

  – l'*extension* d'une classe est l'ensemble des éléments qu'elle contient, d'où l'extension d'un concept est l'ensemble des objets du monde auxquels ce concept peut référer (l'extension du concept *arbre* est l'ensemble des arbres qui existent) ;
  – l'*intension* d'une classe est l'ensemble minimal des propriétés caractéristiques de ses éléments, et l'intension d'un concept est donc la liste des conditions que des objets doivent remplir pour pouvoir servir de référents à ce concept (pour *arbre*, il s'agit de quelque chose comme : avoir un tronc, avoir des feuilles qui tombent éventuellement en hiver, etc.)

  La sémantique extensionnelle se présente comme une avancée par rapport à la sémantique vériconditionnelle car elle se débarrasse de la téléologie qui fixe au langage le but de "dire le vrai". Elle accorde une place au contrefactuel[1], ce qui étend le champ des textes qui peuvent être étudiés par la sémantique extensionnelle. On évacue ainsi apparemment le recours aux valeurs de vérité comme interprétation du sens des énoncés. Cependant, on peut retrouver des valeurs de vérité implicites portant sur les rapports entre les concepts et leurs référents. Cette constatation découle

[1] Les choses inventées, fausses ou chimériques.

logiquement de la théorie fregéenne [37] selon laquelle l'intension détermine l'extension[1]. Toute définition basée sur une fonction caractéristique se ramène en fait à une acceptation ou un rejet des éléments susceptibles de faire partie de l'extension du concept, ce qu'on peut aisément représenter par une fonction à valeurs booléennes : VRAI lorsque l'objet considéré fait partie de l'extension du concept, et FAUX lorsqu'il n'en fait pas partie (i.e. lorsqu'il ne correspond pas à sa définition intensionnelle). Ce qui amène à un paradoxe amusant : la sémantique extensionnelle se réduit forcément à une théorie intensionnelle ; on ne peut pas échapper aux valeurs de vérité.

Ces considérations ne laissent aucune place à une quelconque idée de locuteur. C'est pourquoi certains chercheurs on fait en sorte d'intégrer la sémantique extensionnelle aux recherches cognitives, en ne définissant plus le sens comme rapport entre un signe linguistique et un référent objectif (dans la réalité ou dans une fiction), mais comme relation avec un référent "subjectif". Les référents subjectifs sont envisagés soit au niveau perceptif (par Woods [139] qui ramène les conditions de vérité à des opérations élémentaires de perception sensorielle), soit au niveau mental (par Winograd [137] qui voit la sémantique comme étude des rapports entre « les objets linguistiques et les états mentaux. »)

La sémantique extensionnelle pose le rapport entre la langue et le monde comme s'il y avait un contact direct et déterminé entre les concepts et les choses (ou bien les corrélats mentaux) auxquelles ils réfèrent. Toute la variabilité du langage se trouve confinée dans la partie gauche du triangle sémiotique (dans le rapport des mots aux concepts), alors que les flèches concepts $\longrightarrow$ choses sont fixes. La version objective de cette sémantique (avant son intégration aux recherches cognitives) se ramène, tout comme la sémantique vériconditionnelle, à des postulats épistémologiques forts : on ne dispose plus d'une table de vérité ultime, mais d'un reflet conceptuel universel du monde extrêmement semblable au ciel des Idées de Platon. La version subjective de la sémantique extensionnelle (celle qui a recours aux corrélats mentaux ou perceptifs) ne peut, quant à elle, s'expliquer que si l'on suppose l'existence de tels corrélats qui sont des universaux, comme par exemple les stéréotypes de Rosch [105]. On notera cependant qu'il existe des arguments linguistiques et sociologiques qui vont à l'encontre de ces stéréotypes, ou "catégories psychologiques universelles". Ces arguments s'appuient sur des observations sociolinguistiques montrant que certains stéréotypes utilisés par Rosch pour illustrer sa thèse ne se retrouvent pas dans certaines cultures (comme 'siège' ou 'meuble', par exemple). Ces arguments jouent en faveur d'une hypothèse de relativisme culturel, que l'on connaît en linguistique sous le nom d'*hypothèse de Sapir-Whorf*.

Motivations du rejet de la sémantique référentielle

Cette brève présentation permet d'expliquer pourquoi nous ne souhaitons pas nous placer dans le cadre de la sémantique référentielle pour concevoir un modèle de la polysémie d'usage. Nos arguments tiennent en trois points :

1. La première objection est d'ordre épistémologique : la sémantique référentielle repose sur des postulats forts (même s'ils sont implicites ou occultés dans la plupart des travaux) qu'on pourrait résumer ainsi : il existe un moyen de se référer à une compilation de ce qui est vrai (ou réel) de manière univoque. Cette hypothèse est fort tentante pour un informaticien (il existe quelque part une base de faits universelle), mais est de plus en plus contestée par les structuralistes (cf. Sapir [112],

[1] Mathématiquement, la *fonction caractéristique* d'un ensemble détermine ses éléments, ou, pour parler plus simplement, si l'on dispose d'une définition très précise et complète d'un concept, on connaît automatiquement tous les objets qui sont des cas particuliers de ce concept.

Benveniste [4, 5] ou Rastier [100]) et les constructivistes (cf. Piaget [80] ou Vygotsky [132]). La version psychologique moins ambitieuse, qui consiste à postuler des catégories universelles de la psychologie humaine, est empreinte d'un ethnocentrisme qui lui confère une dangereuse fragilité en tant que théorie globalisante.

2. D'autre part, la sémantique référentielle est imprégnée de logique aristotélicienne, et a donc recours aux trois principes argumentatifs qui y sont attachés : le principe d'identité[1], le principe de non-contradiction[2] et le principe du tiers exclus[3]. Or, par leur nature ensembliste, les interprétations d'occurrences de polysémie d'usage défient le principe de non-contradiction et le principe du tiers exclus : de façon générale, on ne peut pas dire qu'un usage exclut les autres usages. Ils sont présents simultanément, avec éventuellement plus ou moins de pertinence. Il n'en reste pas moins que des usages présentant des attributs contradictoires (comme *maison* dans le sens de famille qui possède un attribut 'animé', tandis que *maison* dans le sens de bâtiment possède un attribut 'inanimé'[4]) peuvent très bien coexister dans une même interprétation.

3. Et finalement, la définition des emplois, ou usages, donnée p. 26 implique que les signifiés qui servent à l'interprétation des occurrences de polysémie d'usage appellent des ensembles ouverts d'emplois : ainsi, nous avions donné l'exemple de *convoi*, qui peut se décliner en *convoi ferroviaire*, *convoi militaire*, *convoi funéraire*, etc. On peut donc rencontrer de façon inopinée de nouveaux emplois d'un signifié, créés par l'adjonction spontanée d'un trait sémantique imprévu. Cela met l'accent sur une caractéristique importante de la polysémie d'usage, et sur laquelle nous aurons l'occasion de revenir au §2.5.2 : on ne peut identifier ces signifiés avec des concepts. En effet, un concept est traditionnellement défini par son intension/extension qui sont fixes et le déterminent entièrement. Or, dans le cas de la polysémie d'usage, on a affaire à des signifiés dont l'extension (et donc l'intension) peut-être modifiée par rapport à leur définition originelle de façon imprévisible, d'où la prolifération de leurs emplois. Cette sous-détermination du signifié par rapport à ses emplois a conduit Prince [88] à proposer de le considérer comme un point de vue (ou *vue conceptuelle*), i.e. un "concept partiel" doté d'une étiquette, mais pas d'une définition exhaustive. Auquel cas, un tel signifié *dénote* (par son étiquette) mais ne *réfère* pas. Ce qui nous place résolument en-dehors du champ de la sémantique référentielle.

2.5.1.2 La sémantique inférentielle

La conception du signe en tant qu'*indice* est tout aussi ancienne que celle du signe-symbole au sein du triangle sémiotique. On la retrouve elle aussi chez Aristote puis chez Saint Augustin. Pour paraphraser Aristote, un signe est une chose dont l'existence entraîne l'existence (antérieure ou postérieure) d'une autre chose. Pour remonter du signe à la chose dont il est l'indice, il faut effectuer une *inférence*, i.e. établir une liaison logique *nécessaire*, dans le meilleur des cas une implication.

D'après Sperber et Wilson [117], toutes les théories de la communication depuis Aristote (sous-entendu la sémantique référentielle) sont basées sur un *modèle du code* : communiquer c'est coder et décoder des messages. A ce modèle, ils opposent le *modèle de l'inférence* dans lequel « communiquer, c'est produire et interpréter des indices. »

[1] $\forall a, a = a$

[2] $\forall a, (a \wedge \neg a)$ est faux (on ne peut avoir simultanément une chose et son contraire)

[3] De deux assertions incompatibles sur un même sujet, l'une est nécessairement vraie et l'autre fausse.

[4] Analysé par Prince dans [86].

L'opposition qui est faite entre le modèle référentiel (celui du triangle sémiotique) et le modèle inférentiel reflète la distinction traditionnelle entre la sémantique (qui établit des références) et la pragmatique, qui traite des "sens dérivés" et des "effets de sens" à travers les inférences. Il semble alors naturel de considérer, comme Sperber et Wilson, que ces deux modèles se complètent parfaitement au sein du découpage habituel : syntaxe / sémantique / pragmatique.

L'inférence est une opération mentale (tout comme la référence) qui consiste à mettre en rapport le signe et ce dont il est un indice. La différence fondamentale qui sépare la référence de l'inférence est que la référence établit un rapport entre des objets de natures différentes : un concept d'un côté, une "chose" de l'autre, alors que l'inférence met en relation des objets de même nature : deux choses ou deux concepts. Cependant, ces deux éléments mis en relation ne possèdent pas le même statut, car l'inférence est orientée : l'un est antécédent, l'autre est conséquent. On dira alors que le premier est le *signe* de l'autre.

Le terme d'*inférence* est large et recouvre plusieurs opérations mentales différentes. La plus connue d'entre elles, est la *déduction* (ou implication stricte) $a \Rightarrow b$ ou "si a alors b" en informatique. Mais il en existe bien d'autres, très largement utilisées par l'esprit humain, et posant des difficultés de modélisation informatique, telles que l'induction (si p est vraie pour certains, alors p est probablement vraie pour tous), l'abduction (si p et q sont proches dans l'espace ou dans le temps, alors $p \Rightarrow q$), la généralisation, la spécialisation, etc.

L'inférence en intelligence artificielle

En intelligence artificielle, les *schémas*, *scénarios* et *plans* (cf. §3.2.2) servent principalement de supports pour les inférences. Le principe général est que des inférences diverses servent à déduire la valeur des attributs non encore instanciés à partir des attributs qui ont déjà reçu une valeur. On cherche à compléter des syllogismes (ou raisonnements) incomplets[1]. Or, pour ce faire, il faut nécessairement disposer de syllogismes complets servant de norme à la complétion des enthymèmes. De tels syllogismes normés constituent ce que les anciens (par exemple Aristote [2]) appelaient des *topoï* (avant qu'ils ne se mettent à dénoter plus récemment les lieux communs et autres phrases toutes faites). Les topoï servent à expliciter les relations stéréotypiques qu'entretiennent deux objets du monde, en partant du principe que les relations entre les choses conditionnent les relations entre les signes linguistiques qui y réfèrent.

Causalité et nécessité

On retrouve l'hypothèse implicite d'un monde de sens pré-donné. Les topoï jouent un rôle équivalent dans la sémantique inférentielle à celui que joue la "base de faits universelle" dans la sémantique référentielle. On pourra cependant arguer que les topoï peuvent très bien représenter des phénomènes physiques probablement universels (comme des lois physiques) puisque nous sommes ici dans une théorie (anonyme) de la causalité : en effet, $a \xrightarrow{f} b$ où f est une inférence quelconque s'interprète comme "a donc b" (quelques soient les raisonnement qui se cachent sous ce *donc*). Ce qui est discutable ici, c'est l'hypothèse selon laquelle les relations entre les objets physiques correspondent de manière quasi-isomorphique à des inférences entre les significations de signes linguistiques. Formellement, cela revient à supposer que l'on peut définir un morphisme μ entre le plan physique et le plan sémantique tel que

$$\mu\big(A \xrightarrow{R} B\big) = \mu(A) \xrightarrow{f} \mu(B) = a \xrightarrow{f} b$$

[1] Ce que la rhétorique classique nomme *enthymèmes*.

où A et B sont deux objets du monde reliés par la relation R, a et b sont deux significations de signes linguistiques reliés par l'inférence f.

Cette hypothèse, ainsi que la conception causale qui s'attache au paradigme de l'inférence, entraînent une double exigence de *nécessité* pour la sémantique :

— toute relation causale entre deux objets du monde se traduit *nécessairement* par une inférence entre les signifiés qui leur correspondent ;

— dans le cadre d'une seule inférence, le signifié du signe aboutit *nécessairement* au signifié dont il est l'indice.

Suffisance

L'idée que l'indice *entraîne nécessairement* son signifié correspondant ne concorde pas avec la sous-détermination des inférences qui apparaît avec la polysémie d'usage : non seulement le signe polysémique entraîne plusieurs possibilités d'interprétation, mais en plus, comme nous avons pu le voir dans les exemples du §2.4.1, ces différentes possibilités peuvent se combiner entre elles non pas pour s'éliminer mutuellement, mais pour se renforcer les unes les autres. En somme, avec la polysémie d'usage, on a plutôt affaire à des mécanismes de *suffisance* qu'à des mécanismes de *nécessité*, et c'est pourquoi la sémantique inférentielle ne peut, selon nous, expliquer entièrement ce phénomène.

2.5.2 La sémantique différentielle

Les deux paradigmes sémantiques principalement utilisés en traitement automatique des langues naturelles s'avèrent insuffisants pour décrire la polysémie d'usage. Ces deux cadres de description ne nous satisfaisant pas, nous nous tournons vers le troisième paradigme de la linguistique moderne : le paradigme *différentiel*.

2.5.2.1 La valeur

Pour mieux comprendre la sémantique différentielle, il faut partir d'une réflexion sur les synonymes. Deux synonymes diffèrent par leurs idées accessoires (périphériques), que l'on nomme aussi *traits connotatifs* (cf. §2.7.1) en linguistique. Dans le modèle de la triade aristotélicienne, deux synonymes sont associés à un même concept, ou idée principale. Or, à strictement parler, ils devraient renvoyer à deux concepts différents de par leurs idées accessoires, et par là, devraient référer à deux objets différents (et ainsi, une *bicyclette* n'est pas un *vélo*).

Ce genre de paradoxe a conduit Saussure [26] a former la notion de *valeur* :

— La valeur est la véritable réalité des unités linguistiques ;

— elle est déterminée par les positions relatives des unités dans le système (i.e. par leurs différences mutuelles) ;

— il n'y a pas de quantité ou de qualité linguistique mesurable avant la détermination de la valeur par le système.

La valeur est ce qui fait que les signes linguistiques “veulent dire quelque chose” *lorsqu'ils sont dans leur contexte naturel* (celui de leur utilisation dans un discours). C'est une fonction exprimant les relations que peut entretenir une unité linguistique avec toutes les autres, et qui se ramènent à des oppositions entre unités au sein de classes fermées. Cela signifie qu'on ne peut jamais donner explicitement ce que l'unité X veut dire, mais plutôt ce qu'elle ne veut pas dire parce qu'elle diffère de telles unités dans tel contexte, et de telles autres dans un autre contexte encore. Il s'agit donc de définitions *apophantiques*[1] .

Ordre herméneutique

Cette théorie linguistique permet de décrire des phénomènes difficilement explicables par les sémantiques référentielle et inférentielle, grâce à la réhabilitation de l'ordre herméneutique[2] puisque c'est l'ensemble de la langue qui détermine la valeur de chacune de ses unités. Mais on commence également à voir se profiler des difficultés pour l'informaticien cherchant à mettre en œuvre ce genre de modèle : le codage informatique reste basé sur la logique, qui est par essence compositionnelle. Or, même si la linguistique différentielle peut se décrire dans un cadre componentiel (par des jeux de traits), il s'agit maintenant de circonscrire la signification en opposant ou en enlevant des traits à une “masse” sémantique (les classes) et non plus de la construire en ajoutant des traits. De simples considérations de complexité laissent entendre la quasi impossibilité de coder *in extenso* un modèle différentiel de la langue en raison de l'explosion combinatoire des traits.

Il n'en reste pas moins important que la sémantique différentielle introduit une nette séparation entre le signifié et le concept. Le signifié est défini opératoirement au sein d'une langue par ses oppositions avec les autres signifiés. En somme, un signifié se définit relativement à une langue et à l'intérieur de celle-ci, tandis qu'un concept est absolu et extérieur à toute langue. De ce fait, un concept peut être considéré comme composé de représentations à vocation universaliste (et il devient tout à fait légitime de chercher à définir un concept en-dehors de l'expression linguistique). Au contraire, un signifié appartient à l'espace de significations d'une langue et est tissé de relations “X est comme Y mais...” où Y est lui-même un signifié. Par conséquent, les signifiés ne peuvent être bien représentés qu'au sein de leur langue, par des processus (infinis) de définition circulaire : c'est ce qu'on appelle aussi le *cercle herméneutique*.

2.5.2.2 Les éléments constitutifs de la sémantique différentielle

Signes et signifiés

Là où la linguistique traditionnelle ne connaît comme signe que le mot et comme signification que le concept auquel il est associé, la linguistique différentielle voit deux niveaux de complexités différentes :

— Les signes linguistiques minimaux (les plus petits) sont appelés *morphèmes* qui possèdent un signifiant (ne formant pas un mot autonome) et un signifié : le *sémème*. De ce fait, les morphèmes, sans pouvoir constituer des unités lexicales à eux seuls, sont porteurs de suffisamment de sens pour pouvoir constituer de nouvelles unités lexicales. Pour emprunter une expression de Ecco (dans *Le pendule de Foucault*), le mot *tétracapillotomiste*[3] contient quatre morphèmes : *tétra-* (quatre), *capillo-* (cheveu), *tom-* (couper) et *-iste* (agent).

[1] *Apoha* signifie *répudiation* en sanskrit.

[2] Il s'agit d'un des quatre ordres de description linguistique (syntagmatique, paradigmatique, herméneutique et référentiel). Cet ordre permet de décrire les conditions de production et d'interprétation des énoncés.

[3] Ou “coupeur de cheveux en quatre”, d'après l'auteur.

— Une *lexie* est un groupe de morphèmes intégré qui forme une unité de signification autonome, dont le signifié est la *sémie*. Les lexies sont plus ou moins intégrées, suivant qu'on peut rajouter ou pas des morphèmes à l'intérieur. Une lexie constituée d'une seule unité graphique (ou mot) maximise son intégration : on ne peut rien y rajouter. Les lexies composées de plusieurs unités graphiques peuvent être soit figées (comme *à la queue leu leu*) et n'admettent pas d'insertion, soit elles admettent des insertions limitées (dans *il prit aussitôt ses jambes à son cou*, l'adverbe *aussitôt* a été inséré dans la lexie *prendre ses jambes à son cou*).

Unités microsémantiques

En partant de la distinction entre signifié et concept qu'opère la sémantique différentielle, on peut se servir des systèmes de notation par traits pour décrire la signification d'unités plus petites que le "sens du mot". C'est le domaine de la microsémantique.

La sémantique différentielle nomme *sèmes* les traits qu'elle utilise dans ses descriptions microsémantiques. Les sèmes s'organisent en ensembles structurés appelés *sémèmes* (les signifiés sont constitués d'un ou plusieurs sémèmes). En fait, ces deux définitions se font plutôt dans l'autre sens : les sèmes se définissent comme des relations d'opposition ou d'équivalence au sein de classes de sémèmes : ainsi le sème /masculin/ ou /féminin/ différencie 'fille' de 'garçon', tandis que le sème /mature/ ou /immature/ différencie 'fille' de 'femme'.

Des auteurs comme Rastier [101], estiment que l'utilisation d'un sème est dépendante de la langue, ainsi que de critères herméneutiques de pertinence : les sèmes ne sont pas des universaux, ni logiques, ni psychologiques. Ils sont constitués et utilisés en fonction des besoins d'analyse d'un texte particulier dans une optique particulière (et du coup, ils sont en nombre infini).

On sépare les sèmes en deux sortes :

— Les *sèmes génériques* sont hérités depuis une classe sémantique par les sémèmes appartenant à ses sous-classes sémantiques. Ils dénotent une relation d'équivalence : les sémèmes de la super-classe et des sous-classes sont équivalents du point de vue des sèmes génériques.

— Les *sèmes spécifiques* permettent de différencier les sémèmes au sein d'une classe sémantique.

Cette séparation entre sèmes génériques et spécifiques n'a rien d'absolu : un même sème peut être générique dans une classe et spécifique dans une autre, et puis cela dépend aussi de la définition des classes sémantiques, qui est bien entendu fonction de l'objectif d'analyse.

De plus, un sème, qu'il soit générique ou spécifique, peut posséder deux statuts :

— Un *sème inhérent* appartient au type du signifié dont on observe une occurrence. En l'absence d'indication contraire, un sème inhérent est hérité par défaut du type dans l'occurrence. Par exemple, une occurrence de 'éléphant' hérite par défaut de /gris/ sauf dans une expression comme *un éléphant blanc*.

— S'il n'est pas inhérent, on parle de *sème afférent*. L'afférence peut être codifiée par des normes socioculturelles (et ainsi la classe //'homme', 'femme'// reçoit des afférences issues de la classe //'force', 'faiblesse'// sous forme des sèmes /fort/ et /faible/). Dans ce cas, ces sèmes sont associés aux types *homme* et *femme*, mais sans être définitoires. Ce sont des traits qui deviennent éventuellement des valeurs pour des attributs facultatifs des occurrences de ces types.

Mais l'afférence peut aussi se résumer à une propagation de traits en contexte, entre occurrences, sans que leurs types entrent en jeu. Il s'agit alors de sèmes propagés par le contexte, comme dans *le loup apprivoisé*, où le sème /apprivoisé/ doit pouvoir être pris en compte dans l'occurrence de 'loup', même s'il s'agit d'une caractéristique purement locale.

Finalement, il existe quatre niveaux de pertinence d'un sème au sein d'un sémème pour une occurrence donnée :

— *neutralisé* : le sème est exclus du sémème ;

— *virtualisé* : le sème est momentanément désactivé, mais il demeure réactualisable par la suite ;

— *actualisé* : le sème est présent dans le sémème ;

— *saillant* : le sème est prééminent.

Classes sémantiques

1. Les classes sémantiques minimales sont appelées *taxèmes*. Elles définissent le sème le moins générique de leurs sémèmes (ce qu'ils ont en commun), ainsi que les sèmes spécifiques différenciant les sémèmes contenus dans un taxème. Les taxèmes sont les plus petites classes de sémèmes ayant un sème générique commun, et ils sont généralement définis opératoirement par l'observation de phrases contenant des alternatives. Par exemple (pour reprendre Rastier, *op. cit.*), on peut rencontrer *Tu prends le bus ou le métro ?*, mais pas *Tu prends l'autobus ou l'autocar ?*, ce qui implique en français l'existence d'un taxème des moyens de transports en commun intra-urbains //'métro', 'bus' ou 'autobus'//, et un pour les moyens de transports en commun extra-urbains //'train', 'autocar'//, mais pas de taxème mixte contenant à la fois 'autobus' et 'autocar'. La définition des taxèmes dépend donc de pratiques culturelles variant avec les langues.

2. Au cran de généralité supérieur, on trouve les *domaines*. Un domaine est lié à un type de pratique sociale donné. On trouve généralement des indications de domaines dans les dictionnaires comme *chim.* (chimie) ou *cuis.* (cuisine). On pourra noter qu'un taxème n'appartient pas forcément à un domaine (par exemple les taxèmes grammaticaux). Les domaines présentent les propriétés suivantes, qui peuvent servir à les identifier :

   — Il n'existe généralement pas de polysémie lexicale au sein d'un domaine.

   — On n'établit pas de métaphore entre éléments d'un même domaine.

3. Les taxèmes se structurent en *champs*. Ainsi, un champ //moyen de transport en commun// peut regrouper les deux taxèmes //'métro', 'bus'// et //'train', 'autocar'//.

4. Finalement, les classes de plus grande généralité sont les *dimensions*. Ces classes dénotent des grandes oppositions dans l'univers sémantique comme /végétal/ vs. /animal/ ou /animé/ vs. /inanimé/. Les sémèmes d'un même taxème appartiennent généralement à la même dimension, sauf dans le cas des taxèmes graduels : ainsi dans //'glacial', 'froid', 'chaud', 'brûlant'//, 'froid' et 'chaud' appartiennent à la dimension /neutre/, et 'glacial' et 'brûlant' à la dimension /extrême/.

Ces classes ne s'organisent pas en degrés d'abstraction croissants, mais plutôt en degrés de figement : les taxèmes évoluent relativement vite et peuvent être remaniés par un discours, les domaines évoluent à l'échelle du siècle, et les dimensions à l'échelle du millénaire.

2.5.2.3 Signification et interprétation

La microsémantique différentielle s'appuie sur une théorie de l'interprétation, plutôt que de la compréhension. Elle entend rester à l'intérieur d'une pratique de la langue : avant de comprendre un texte, il faut l'avoir interprété (i.e. être capable de le paraphraser ou de l'expliciter), et c'est cette étape préalable qui appartient à la linguistique, les étapes ultérieures sortant de son cadre.

Il existe d'innombrables façon d'interpréter un texte. Pour la sémantique différentielle, avec les définitions données ci-dessus, interpréter un texte, c'est pouvoir donner l'ensemble de ses sémèmes, avec leurs sèmes indexés dans leurs différentes classes sémantiques. Et la signification d'une occurrence s'explicite alors par l'attribution de valeurs de pertinence à chacun de ces sèmes.

De ce fait, on reste à l'intérieur de la langue pour en décrire la sémantique, ce qui a nettement tendance à réduire les inadéquations entre systèmes de notation linguistique et logique.

Le *sens* (d'un sémème-type) est considéré comme une abstraction artificielle des *significations* (de ses sémèmes-occurrences). Et celles-ci sont *construites en contexte* par les influences mutuelles des différents sèmes. Le texte est conçu comme contenant des instructions de lecture permettant de mettre en œuvre ces influences. Le cheminement qui part des instructions de lecture à une distribution donnée de valeurs de pertinences aux sèmes est appelé *parcours interprétatif*.

Opérations interprétatives

Deux micro-opérations sont à l'œuvre dans les parcours interprétatifs :

— la *dissimilation* qui opère sur de faibles contrastes sémantiques et vient augmenter la différence existant entre deux sémèmes (c'est souvent ce qui se passe avec les tautologies comme *Paris sera toujours Paris* où l'on affecte une acception différente aux deux occurrences de *Paris*) ;

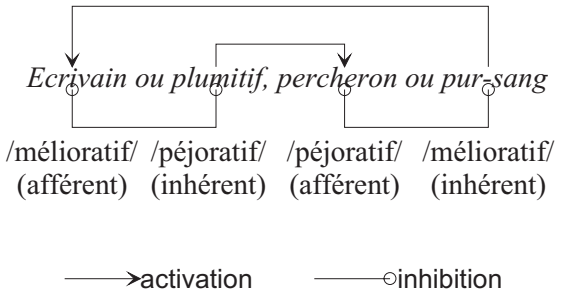
— l'*assimilation* qui fonctionne à l'inverse sur de forts contrastes sémantiques et gomme les différences entre sémèmes (Rastier [101] cite *des fous, des femmes et des fainéants* où *femme* reçoit un sème /péjoratif/ par assimilation à *fou* et *fainéant*).

A un niveau supérieur, les sens des lexies sont déterminés par trois opérations fondées sur l'assimilation et la dissimilation :

— L'*inhibition* interdit à un sème inhérent de s'actualiser et va le virtualiser. Ainsi, le sème /mort/ est inhérent à *assassiner*. Il est présent dans *Ravaillac a assassiné Henri IV*, mais il est virtualisé dans *McEnroe a assassiné Connors sur le court*.

— L'*actualisation* opère sur les sèmes afférents. Ainsi, le sème /debout/ n'appartient pas à la définition du sémème 'chevalier' mais peut être inféré du sème inhérent /humain/. Il est actualisé dans *le chevalier faisait rempart* par le sème /verticalité/ présent dans 'rempart'.

— Et finalement la *propagation de sèmes* concerne les sèmes afférents en contexte (entre occurrences). Cette opération greffe un nouveau sème sur un sémème qui ne le possède pas originellement (même à l'état virtuel) pour des raisons le plus souvent de cohérence textuelle. Rastier cite cet exemple emprunté à Julien Gracq :



Les sèmes /mélioratif/ inhérent à ‘pur-sang’ et /péjoratif/ inhérent à ‘plumitif’ propagent /mélioratif/ à ‘écrivain’.

2.5.2.4 Isotopie

Nous ne rentrerons pas ici dans les détails de la théorie des isotopies, concept introduit par Greimas [46]. Cette théorie est en effet dense, complexe, subtile et ouverte à des débats de spécialistes.

Nous allons nous baser sur la définition proposée par Rastier [99] :

> *« On appelle isotopie toute itération d'une unité linguistique. L'isotopie élémentaire comprend donc deux unités de manifestation. Cela dit, le nombre des unités constitutives d'une isotopie est théoriquement illimité.»*

Une isotopie est l'apparition récurrente d'une unité linguistique, quelle qu'elle soit, au long d'un texte. Ce qui nous intéresse plus particulièrement ici, ce sont les *isotopies sémantiques*, c'est-à-dire la récurrence d'un ou plusieurs sème(s).

Il faut tout de suite noter une propriété importante de l'isotopie : *les isotopies sont des suites non ordonnées*. Outre les implications qui en découlent sur les aspects syntaxico-syntagmatiques des isotopies, on retiendra surtout qu'une isotopie ne se “propage” pas du début vers la fin d'une portion de texte ; elle n'est pas orientée. Nous considérerons plutôt une isotopie comme un “maillage” d'une portion de texte : à différents endroits réapparaissent des éléments équivalents (pour nous, ces éléments seront plutôt des sèmes).

Le phénomène d'isotopie sémantique a comme effet le plus marquant *d'assurer la cohérence d'un texte*. A ce titre, certaines isotopies sont induites par la récurrence d'un sème spécifique (par exemple, le sème /inchoactif/[1] dans *L'aube allume la source* (Eluard), qui est présent dans ‘aube’, ‘allume’ et ‘source’), ou d'un sème générique (comme /navigation/ dans *L'amiral ordonna de carguer les voiles*). On fait également la distinction entre les *isosémies*, qui sont des isotopies prescrites par le système fonctionnel de la langue, et les isotopies au sens large, qui relèvent d'autres types de normes. On les appelle encore isotopies facultatives.

Les isotopies et les isosémies structurent l'interprétation. Cela peut aller des énoncés ne contenant ni isosémie, ni isotopie facultative (ce qu'on appelle plus généralement du charabia, comme *suite amiral A les contenant*) à ceux contenant une isotopie facultative et des isosémies : dans ce cas, plusieurs sémèmes appartiennent à un seul domaine, sans contradiction avec un autre domaine, et se suivent de façon fonctionnellement correcte (ce sont les énoncés “clairement compréhensibles”). En fait, il est plus important de disposer d'une isotopie facultative que d'isosémies, car les énoncés agrammaticaux mais sémantiquement cohérents sont aussi bien compris que ceux qui sont grammaticalement corrects.

[1] Ou *commencement* pour parler plus simplement...

Nous retiendrons que la cohérence d'un texte est garantie par l'existence d'un fonds sémantique constitué par des isotopies.

— Les isotopies de traits génériques ont des effets qui dépendent du type de classes manifestées. Les isotopies sur les dimensions créent les tons, ou niveaux de langues, et structurent les points de vue globaux abordés dans le texte. Les isotopies sur les domaines créent les *impressions référentielles*[1] globales, et les isotopies de taxèmes les impressions référentielles locales.

— Les isotopies de traits spécifiques assurent, quant à elles, la conservation des formes sémantiques, et provoquent un effet de cohérence textuelle.

Pour finir, notons que la propagation des traits par les isotopies dépend du type de ces traits. Les traits génériques se propagent plus généralement que les traits spécifiques dont la propagation nécessite la constitution de contextes particuliers comme les métaphores ou les comparaisons. Parmi les traits génériques, les sèmes dimensionnels se propagent facilement (c'est ce que nous avions noté *restrictions de sélection* à la page 25). Comparativement, les sèmes domaniaux se propagent moins facilement, mais engendrent des isotopies participant à la définition de ce que l'on appelle les discours. Et finalement, les sèmes taxémiques ne se propagent que dans des contextes fortement assimilateurs.

Nous retiendrons surtout que la propagation des sèmes (en général) peut être très intense au sein d'un même syntagme, mais est bloquée par les frontières syntaxiques. Les isotopies sont relativement fréquentes au sein d'une même phrase, mais on tendance à s'amenuiser lors du passage aux phrases contigües, et sont encore nettement diminuées par les frontières de paragraphes.

2.6 Les limites de la dénotation

Nous voulons montrer ici en quoi la sémantique différentielle nous semble plus prometteuse pour une modélisation de la polysémie d'usage (et de la polysémie d'acception par ailleurs) que les théories référentielle (extensionnelle) et inférentielle. Cette constatation sera néanmoins modérée par des observations sur la difficulté d'appliquer la sémantique différentielle directement par des moyens informatiques.

Il est primordial de rappeler tout d'abord que nous nous situons ici dans un cadre de modélisation *informatique* de la langue. Cela implique que l'on a forcément recours à une sémantique *dénotationnelle*. Par dénotationnelle, on entend communément l'utilisation de *dénotations*, qui sont des représentations (symboliques ou numériques) des objets que l'on cherche à étudier.

Codage des concepts

Les modèles référentiel et inférentiel présupposent que l'on peut, d'une certaine façon, identifier dénotation et référence, par le truchement des concepts. Leur fondement commun peut s'exprimer ainsi[2] : le sens se trouve dans la mise en référence des mots et des objets qu'ils évoquent. Or il n'est pas possible de "mettre" les référents dans une machine. Mais tout objet est une instance d'un concept qui est universel et entièrement cernable par sa description. Par conséquent, on peut "mettre" les concepts dans une machine (ou plutôt coder leur description, ce qui revient au même), et du coup il devient

[1] L'impression qu'on est en train de parler de quelque chose.

[2] Bien sûr, nous exagérons ici les positions afin de mieux illustrer notre propos.

possible de créer une référence directe entre les mots et les concepts (ce qui est peut-être encore plus intéressant que de référer aux objets eux-mêmes).

Il s'agit là d'une conception omniprésente, qui se ramène à la philosophie du langage, s'enracinant elle-même dans une longue et puissante tradition hellénistique. Il est vrai que l'on conçoit mieux *a priori* l'existence des concepts (universels et absolus) que celle des signifiés qui sont des abstractions dépendantes des langues et des cultures qui s'y rattachent. Cependant, outre les discussions de plus en plus pressantes sur le fait que les concepts seraient construits à l'intérieur des langues, on peut s'étonner du postulat que les concepts seraient entièrement compris dans leur description. Nous voudrions attirer l'attention sur le fait que dans toutes les bases de connaissances conceptuelles que nous avons observées (et le lecteur pourra certainement se ramener à sa propre expérience), il existe en définitive des concepts minimaux, qui ne sont décrits que par leur étiquette. Celle-ci est invariablement une étiquette lexicale. Nous interprétons cela de la façon suivante : puisque ces concepts ne possèdent pas d'autre description, c'est que leur sens est contenu dans les mots qui les étiquettent, et on est donc obligé de faire appel à leur signifié. C'est-à-dire que même en soutenant le contraire, on suppose implicitement l'existence d'un plan sémantique autonome au sein de la langue fournissant ces étiquettes lexicales, et auquel on se ramène en dernier lieu.

De plus, les modèles utilisés en traitement automatique des langues naturelles se fondent presque intégralement sur l'hypothèse de compositionalité du sens, c'est-à-dire ramènent l'acte de signifier à une logique ou une algèbre. Il est vrai qu'en partant des étiquettes lexicales (i.e. des concepts minimaux) comme axiomes, des règles logiques et/ou algébriques peuvent construire tout un univers de sens fonction de ces axiomes. Mais la question qu'on occulte en faisant cela est : dans quoi s'enracinent les axiomes pour signifier ? Par exemple, pourquoi ne pas avoir pris des étiquettes absurdes, sans signifié ? Le problème majeur de cette approche se trouve là, selons nous : en faisant cela, on confond la notation avec le sens. Nous sommes d'avis que les applications des modèles référentiel et inférentiel "fonctionnent" bien dans un grand nombre de cas parce que :

1. les axiomes sémantiques/conceptuels sont en fait des signifiés ;
2. on applique ces modèles surtout au sous-ensemble des énoncés d'une langue qui se construisent de façon quasi-algébrique à partir des signifiés des mots qui les composent.

Tout cela est, bien entendu, en majeure partie affaire d'opinion. Mais la sémantique computationnelle est un domaine jeune, reposant sur des théories incomplètes, où l'on est bien obligé de faire des choix devant des alternatives non encore tranchées.

Avantages de la sémantique différentielle

Nous résumerons ainsi notre prise de position : puisque même les modèles référentiel/inférentiel donnent lieu à des applications qui se fondent sur une utilisation des signifiés, et que la nature algébrique de la construction du sens n'est toujours pas évidente après deux mille ans de tentatives pour le montrer, nous choisissons de considérer que le langage est un bon métalangage pour le langage. Ou plutôt *une langue est un très bon outil pour se décrire elle-même.*

C'est pour cette raison que nous sommes enclin à utiliser plutôt la sémantique différentielle pour modéliser le problème qui nous occuppe. En effet, la sémantique différentielle :

— reconnaît un univers sémantique autonome au sein d'une langue (les signifiés) ;

— caractérise les signifiés par des traits (les sèmes) qui sont :

  – des unités de la langue elle-même (mots ou paraphrases) ;
  – choisis de manière opportuniste ;

— adopte une vision herméneutique de la construction du sens : c'est le global qui influence le local ;

— ne se fonde pas sur une prééminence de la syntaxe (grâce aux isotopies).

Inconvénients de la sémantique différentielle

Bien entendu, même si la sémantique différentielle semble résoudre un problème épistémologique d'adéquation entre les modèles informatiques et la réalité de la langue observée, elle soulève de nombreux problèmes quant à la réalisation d'applications :

1. comment déterminer les sèmes pertinents à un moment donné ? (il y en a potentiellement une infinité)
2. comment gérer dans les faits l'approche descendante du global vers le local alors que l'on travaille sur des architectures qui sont matériellement compositionnelles (donc ascendantes) ?
3. comment gérer les boucles de rétroaction infinies induites par les cercles herméneutiques ?

La réponse globale que nous apportons est celle-ci : comme toujours en informatique, en faisant des compromis. De façon plus détaillée, et respectivement aux questions ci-dessus :

1. En choisissant de modéliser une *expérience de la langue* plutôt que la langue elle-même : on répertories des cas particuliers qui vont servir de points de repères (mais ne pourront jamais être exhaustifs).
2. En abaissant nos ambitions : de la même façon que ci-dessus, on choisira expérimentalement des caractéristiques mesurables et généralement pertinentes pour simuler la notion de contexte global.
3. En séparant les statuts des éléments décrits et des éléments descripteurs : dans une même itération, un même élément ne peut avoir simultanément les deux statuts. Le problème des cercles herméneutiques apparaît lorsque les mots se comportent simultanément comme des objets linguistiques *et* des désignations symboliques (cf. Brachman [10]).

Styles de codage

Finalement, il nous reste un dernier choix à effectuer. Rastier [101] estime que le meilleur paradigme de codage informatique pour la sémantique différentielle est celui des réseaux connexionistes. C'est également l'avis de Fuchs et Victorri [130]. Effectivement, les mécanismes d'isotopie, d'assimilation et de dissimilation sont similaires aux mécanismes de propagation d'activité des réseaux connexionistes. Cependant, l'emploi de ces réseaux soulève selon nous deux problèmes :

1. L'arbitraire du codage aux interfaces : l'activation de la couche d'entrée résulte d'une transformation arbitraire des données réelles, et les activités sur la couche de sortie doivent faire l'objet d'une interprétation avant d'être utilisées. En fait, il n'y a pas de sens dans le réseau. Ce sont ses concepteurs qui transforment les données entrantes et sortantes suivant des règles qu'ils ont eux-mêmes édictées.

2. L'information codée à l'intérieur du réseau n'est pas directement accessible, ni souvent interprétable. On ne comprend pas bien ce qui s'y passe.

L'arbitraire du codage en entrée et en sortie nous semble découler naturellement du caractère dénotationel inhérent à l'informatique. En cela, les réseaux connexionistes n'offrent donc aucun avantage par rapport aux approches symboliques. De plus, il est bien plus facile de comprendre ce qui se passe à l'intérieur d'un système symbolique que dans un réseau connexioniste. Le seul avantage que nous concédons donc à ces réseaux est leur mode de fonctionnement naturel par propagation de l'activité.

C'est pourquoi, nous sommes d'avis qu'un bon modèle informatique de la polysémie d'usage devrait :

1. s'appuyer sur la sémantique différentielle (entièrement décrite au sein d'une langue) ;
2. adopter une démarche hybride entre les systèmes symboliques (pour la clarté et la transparence des processus) et les réseaux connexionistes (pour la propagation d'activité).

2.7 Connotation, profondeur et continuité

Nous nous sommes donnés le cadre théorique de la sémantique différentielle pour l'étude de la polysémie d'usage. A l'intérieur de ce cadre, nous souhaitons aborder maintenant trois notions qui nous semblent se trouver au cœur du phénomène de polysémie d'usage : la connotation, la profondeur sémantique, et la continuité sémantique.

2.7.1 Le rôle de la connotation

La notion de connotation n'est pas des plus clairement définies en linguistique, soit parce qu'extrêmement restrictive selon certains auteurs, soit trop large pour d'autres (cf. la discussion de Rastier [99]).

Résumons les différentes positions ainsi : est dit *connotatif* tout trait sémantique *en plus* du sens strict d'une unité lexicale. En fait, toutes les définitions de la connotation opposent, dans la signification d'un mot, la partie connotative (supplémentaire ou ajointe) à la partie dénotative (ou distinctive). Parler de connotation nous ramène à opposer un sens propre à d'autres sens (figurés ou secondaires). Bien entendu, pour accepter cette opposition, il faut préalablement accepter que les mots puissent avoir un sens premier ou littéral, c'est-à-dire qu'il existe un concept correspondant à ce mot, ou une réseau de conditions nécessaires et suffisantes pour déterminer ce concept.

Cependant, notre présentation de la polysémie d'usage (cf. §2.3.4) nous conduit aux deux remarques suivantes :

1. il n'y a pas d'usage qui soit "le meilleur" (ou le premier, ou littéral) ;
2. il n'est pas aisé (nous dirons même le plus souvent impossible) de déduire un usage d'un autre, car les opérations rhétoriques qui ont présidé à leur création (métonymies, synechdoques, hypallages...) se sont majoritairement perdues au cours du temps.

Rastier (*op. cit.*) rejette la mise en opposition connotation *vs* dénotation et fait éclater la notion de connotation en sèmes génériques virtuels (non actualisés), sèmes afférents socialement normés et sèmes afférents localement normés.

Loin d'entrer dans de tels débats, nous voudrions utiliser une version plus "sociologique" de la connotation, que nous définissons en deux étapes :

1. Pour nous, connotation ne s'oppose pas à dénotation, et nous ne reconnaissons pas le statut de "sens premier" ou "littéral" (au mieux reconnaissons-nous celui de "le plus fréquent").

2. Nous appellerons connotation une référence *implicite* à une connaissance propre à la culture du groupe social (réel ou simulé) de l'auteur.

En faisant appel à des connaissances liées à une culture, nous souhaitons évacuer les connaissances encyclopédiques, qui ne peuvent rendre compte des emplois et usages. Par exemple, les connaissances encylopédiques sur *flic* et *policier* concernent la description du même concept de fonctionnaire de police. La nuance péjorative attachée à *flic* appartient en propre à la culture du français et n'entre pas dans la description encyclopédique des attributs et fonctions du policier. C'est pourtant elle qui fait de la déclaration de Clémenceau « Je suis le premier flic de France » un bon mot. On pourra certes faire remarquer que toutes les langues possèdent vraisemblablement un terme argotique péjoratif pour désigner les policiers, et qu'en tant que constante, le rejet généralisé du policier par ses concitoyens pourrait être inclus parmi les connaissances encyclopédiques. Prenons alors un autre exemple moins contemporain, et donc moins risqué. Au siècle de Périclès, les Grecs donnaient le nom argotique d'"archer" à la monnaie perse, pièce d'or montrant l'empereur Darius couronné tenant un arc à la main. Il en résulta une plaisanterie perse classique : « Aucun Grec ne reste insensible à un archer perse. » reposant sur plusieurs caractéristiques supposées des Grecs dans l'imaginaire perse, y compris l'avidité d'or. Ce n'est certes pas la définition encyclopédique (conceptuelle) de *Grec* qui donne son sel à cette plaisanterie.

On a vu au §2.3.4 que la caractéristique la plus saillante de la polysémie d'usage est qu'elle engendre des phrases "à tiroirs", que ce soit dans un but d'amusement ou de condensation des informations. Ce que nous voulons faire remarquer ici, c'est que l'impression d'ambiguïté qui naît de la lecture de ces phrases repose en fait sur la notion de connotation que nous venons de définir : les différents usages saillants simultanément sont identifiés grâce à des références implicites à la culture des locuteurs. Cela réapparaîtra d'ailleurs de façon opératoire au §4.2.2.

En partant de cette observation, nous déduisons que les phrases présentant une polysémie d'usage tombent majoritairement dans la catégorie des *idiomatismes*, et rendent donc difficile leur traduction, à moins que les cultures attachées aux langues source et cible ne partagent un bon nombre des références implicites mises en jeu. En particulier, les jeux de langage reposant sur la polysémie d'usage devraient se révéler généralement intraduisibles sans perdre leur caractère ludique.

2.7.2 La profondeur sémantique

Parallèlement au paradigme componentiel dans lequel s'inscrit la sémantique différentielle, on rencontre en linguistique (Martin [70]) et en informatique (Kayser [57]) (représentation des connaissances), une tendance à décomposer le sens selon une "profondeur variable". Cette approche, qui reste parfaitement cohérente avec les linguistiques componentielles, se base sur l'idée que le sens est fonction du niveau de généralité et de la catégorie fonctionnelle des signifiants. Nous verrons au paragraphe suivant que la notion de profondeur sémantique n'est pas sans importance dans celle, plus générale, de continuité sémantique.

L'approche prônant une interprétation à profondeur variable apporte une notion de "dynamique" par rapport à l'analyse componentielle, essentiellement statique (même si l'idée que les sèmes doivent

être choisis de façon "opportuniste" ouvre la voie à une conception dynamique de la représentation des signifiés lexicaux). Martin proposait l'emploi d'opérateurs (d'addition et de soustraction) sur les sèmes associés à une unité lexicale qui permettraient, en itérant les applications, d'obtenir un niveau d'interprétation d'indice i (en fonction des itérations des opérateurs). Du côté informatique, la théorie développée par Kayser se base sur le principe suivant : fournir un système formel à fonctionnement par défaut dans lequel le cas de base (adopté par défaut) correspond à un niveau "racine" d'interprétation. Tout ceci en postulant que le sens évolue de façon quasi continue en profondeur (i.e. en généralité) à l'intérieur d'une représentation hiérarchique des composants conceptuels pertinents.

La notion de *niveau d'interprétation* se formalise alors de la façon suivante : un mot m d'indice i est interprété à la fois par sa *profondeur* (le chemin conduisant de l'interprétation racine jusqu'au composant i) et par sa *largeur* (les composants adjacents au sein du niveau i de la représentation conceptuelle arborescente).

On trouve chez Kayser [57] un exemple illustrant ceci :

Le livre est sur l'étagère.

Le mot *livre* peut être qualifié de polysémique, au moins au niveau fonctionnel. Il couvre aussi bien un objet physique (*un gros livre en cuir*), une référence (*le dernier livre de* X), un contenu (*c'est un livre sur les métaphores*) que la qualité d'une production (*c'est un livre extraordinaire*).

Une interprétation au niveau physique de l'exemple ci-dessus produirait l'interprétation : *l'objet livre est situé sur l'étagère*, tandis qu'une interprétation au niveau de la référence donnerait : *la référence (que vous recherchez) est matérialisée dans l'objet qui est situé sur l'étagère*, etc.

La notion de profondeur variable admet le principe qu'un niveau d'interprétation peut être plus ou moins pertinent suivant les moments, en particulier lorsque ce niveau est indiqué par le contexte.

Cette théorie n'est pas incompatible avec la possibilité d'avoir plusieurs interprétations valides simultanément (caractéristique essentielle de la polysémie d'usage et, dans une moindre mesure, de la polysémie d'acception), mais elle part également du principe que, pragmatiquement, une seule interprétation va émerger pour une occurrence donnée.

Remarquons pour finir qu'en partant du principe qu'on peut interpréter un mot à une profondeur arbitraire, les représentations informatiques des connaissances en profondeur variable ne peuvent se permettre d'être extensives sans engendrer des structures d'une taille prohibitive. C'est pourquoi les travaux dans ce domaine s'appuient plutôt sur des connaissances dérivées de règles d'inférence interprétative et sur des raisonnements non monotones que sur des représentations statiques. Ce dernier point jouera un rôle important dans la constitution de notre modèle, comme nous le verrons au §3.6.

2.7.3 La continuité sémantique

Nous avions déjà abordé le problème de la continuité sémantique avec les différents exemples de polysémie d'usage que nous avions présentés (cf. §2.4.1).

Nous avions établi comme caractéristique commune à tous les cas de polysémie d'usage une "continuité sémantique" entre les différents emplois mis en jeu simultanément par la polysémie d'usage. La seule exception à cette régularité étant observée pour les jeux de langage, où l'on a affaire plutôt à une "discontinuité sémantique", le contraste entre les différents emplois étant nécessaire à l'aspect ludique du bon mot.

Nous sommes cependant d'avis qu'il ne s'agit pas là d'une véritable opposition avec d'un côté des cas de polysémie d'usage reposant sur une continuité sémantique, et d'autre part des cas fonctionnant sur un principe inverse. Si l'on y regarde de plus près, et à la lumière du formalisme de la sémantique différentielle, il semblerait plutôt que la particularité des jeux de langage soit l'adjonction d'un processus de *dissimilation* beaucoup plus marqué que dans les autres cas. Pour reprendre l'exemple de « Tu es la reine des tartes », on a, pour les deux interprétations de *reine* un sème commun /superlatif/, accompagné dans un cas de l'actualisation d'un sème virtuel /mélioratif/ et dans l'autre de l'actualisation d'un sème afférent localement normé (ou connotatif pour nous) /péjoratif/. On reste ainsi à l'intérieur d'un taxème de superlativité, où les deux sèmes sélectionnés sont localement mis en opposition par dissimilation afin de respecter un schéma oppositionnel caractéristique de l'ironie.

La continuité sémantique est l'objet d'un intérêt croissant de la part de chercheurs en linguistique et/ou en informatique, comme par exemple Fuchs et Victorri [40]. Nous voudrions nous référer ici plus particulièrement à un article de Prince [88]. Il s'agit de savoir comment simuler la continuité sémantique dans un cadre informatique, discret par essence. L'auteur y aborde le problème de la polysémie (en général) pour souligner les problèmes que soulève une conception de la continuité comme fonction complexe appliquée à une grande quantité de particules de sens minimales (c'est-à-dire le principe de compositionnalité en soi).

Nous nous attarderons avec l'auteur sur un exemple de semi-treillis conceptuel (représentation du sens la plus répandue en traitement automatique des langues naturelles) et son utilisation pour interpréter une phrase simple. Ce graphe est schématisé à la figure 2.5 (tous les arcs figurant sur ce schéma peuvent être orientés dans les deux sens, et les "concepts" qui le constituent sont des descriptions relatives au contexte *to eat*).

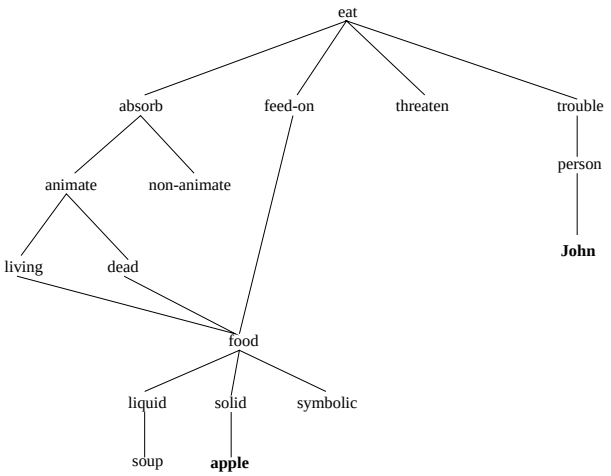


FIG. 2.5: Semi-treillis conceptuel pour *to eat*

On peut tenter d'avoir une idée de continuité sémantique avec un tel semi-treillis en considérant l'ensemble des chemins du graphe.

Par exemple, pour la phrase *John eats an apple*, on peut obtenir les chemins suivants, correspondant à autant d'interprétations valides :

1. eat – absorb – animate – dead – food – solid – apple
2. eat – absorb – animate – living – food – solid – apple

3. eat – feed-on – food – solid – apple

Au-delà du problème d'ambiguïté clairement apparent ici (à condition que les inférences portant sur la pertinence de *feed-on* comme concept valide dans cette interprétation ne soient pas appliquées), l'auteur considère que l'existence de *plusieurs* chemins d'une part, et la proximité entre les composants de ces différents chemins d'autre part, montrent qu'une interprétation sémantique complète de cette phrase semble requérir une certaine "épaisseur". L'auteur plaide ensuite en faveur d'une modélisation qui gère des représentations *ensemblistes* des interprétations plutôt qu'atomiques, sous la contrainte que les éléments appartenant à ces ensembles soient "sémantiquement proches". Grâce à cette proximité sémantique, la continuité est simulée par la prolifération des nœuds sur les chemins, et par des chemins qui constituent un "écheveau sémantique" (pour citer l'auteur).

Nous reviendrons plus longuement sur les propositions de cet auteur au §3.6, et plus particulièrement sur son choix d'utiliser une analyse componentielle en "moyenne résolution" plutôt qu'en "haute résolution".

2.8 Récapitulation

Ce chapitre avait pour but de poser un certain nombre de problèmes, ainsi que de définir les cadres à l'intérieurs desquels nous allons évoluer dans le reste de cet ouvrage. Il s'agissait en fin de compte de découvrir les contraintes linguistiques que nous devrons respecter dans notre travail de modélisation si nous voulons aboutir à un formalisme qui ne soit ni artificiel, ni réducteur.

Nous pouvons récapituler le contenu de ce chapitre par les points suivants :

1. Nos discours sont émaillés d'ambiguïtés de toutes sortes qui peuvent affecter n'importe quel élément de discours, quel que soit son niveau de décomposition : depuis les éléments sublexicaux (comme les désinences) jusqu'aux parties du discours (les paragraphes ou même le texte entier).

2. Les linguistes proposent trois approches différentes des ambiguïtés : le modèle homonymie, le modèle indétermination et le modèle polysémie. Seule cette dernière approche postule qu'un segment de discours ambigu puisse être une entité unique du point de vue de la signification (en ayant une unique "valeur fondamentale") tout en arborant des déclinaisons sémantiques en contexte (des "effets de sens").

3. Modéliser la polysémie (i.e. la coexistence de plusieurs interprétations pour une occurrence d'un segment de discours ambigu) requière de formaliser les notions d'énoncé et de sens des énoncés. Au terme de cette formalisation, on dira qu'un segment est polysémique lorsque son sens n'est ni entièrement contenu dans sa forme (son "apport propre"), ni entièrement déduit du contexte dans lequel il s'insère.

4. On peut distinguer trois familles de phénomènes correspondant à cette définition, soit trois catégories de polysémie pour les unités lexicales :

   (i) la *polysémie fonctionnelle* lorsqu'un même mot peut potentiellement assumer plusieurs fonctions dans l'énoncé, et il faut alors retrouver laquelle s'applique à cette occurrence particulière ;

   (ii) la *polysémie d'acception* lorsque les différentes interprétations de l'élément polysémique correspondent à des distinctions de sens issues d'habitudes (de normes sociales) solidement ancrées dans la langue ;

(iii) la *polysémie d'usage* lorsque les différentes interprétations naissent de la superpositions de plusieurs usages linguistiques qu'on peut faire du mot polysémique.

C'est cette dernière catégorie qui fait l'objet de notre étude, car elle n'a été jusqu'ici que très peu étudiée.

5. Nous avons dénombré par observation quatre sous-catégories de polysémie d'usage : les transferts d'usages, les cumuls d'emplois, les mots jokers et les jeux de langages. Quantité d'énoncés extrêmement banals peuvent être rangés dans ces sous-catégories, prouvant que la polysémie d'usage est un phénomène extrêmement répandu. Après observation de ces sous-catégories, nous pouvons en fournir une caractérisation permettant d'appréhender leurs points communs ainsi que leurs différences.

6. Le principe commun à toutes les formes de polysémie d'usage est la présence simultanée de différents emplois (variantes sémantiques) pour une même occurrence. Le sens de cette occurrence en contexte ne peut être envisagé que comme l'amalgame de ces multiples emplois, avec éventuellement des pertinences locales relatives.

7. Aux trois catégories de polysémie que nous avons relevées, correspondent trois stratégies de traitement de l'ambiguïté, et trois structures de représentation du signifié :

   (a) *Polysémie fonctionnelle* : il faut résoudre l'ambiguïté par une stratégie *winner takes all* afin de trouver "le" bon sens du mot polysémique, compatible avec le contexte de l'énoncé. On a donc affaire à une structure de représentation atomique :

   $$S = \text{sens}_n$$

   (b) *Polysémie d'acception* : il faut restreindre (et non plus résoudre) l'ambiguïté en déterminant les acceptions du mot polysémique qui sont valides dans le contexte d'énonciation, et en les ordonnant, de la plus adéquate (la plus précise) à la moins adéquate (la plus générale). La structure de représentation appropriée est alors une liste ordonnée :

   $$S = (\text{acception}_1, ..., \text{acception}_n)$$

   (c) *Polysémie d'usage* : il ne faut surtout pas résoudre l'ambiguïté : elle n'est pas un problème venant perturber la compréhension, mais un outil d'accumulation sémantique. On cherchera alors plutôt à regrouper tous les emplois du mot polysémique qui sont pertinents dans le contexte d'énonciation. On pourra éventuellement les classer en fonction de leur degré de pertinence, tout en gardant à l'esprit que plusieurs emplois peuvent être également pertinents, et que de toute façon, aucune information n'est à rejeter. On a alors une structure de représentation ensembliste : $S = \{\text{emploi}_1, ..., \text{emploi}_n\}$. Si l'on veut prendre les degrés de pertinence en compte, il s'agira d'une liste ordonnée d'ensembles :

   $$S = (\{\textit{emplois très pertinents}\}, \{\textit{emplois pertinents}\}, \{\textit{emplois valides}\}, ...)$$

8. Les deux cadres linguistiques traditionnels issus de la philosophie du langage, à savoir la *sémantique référentielle* et la *sémantique inférentielle*, reposent tous deux sur des postulats épistémologiques forts, non prouvables, et qui cadrent difficilement avec l'observation de la polysémie d'usage. Ces postulats se ramènent en fait à l'existence de *concepts* qui sont des abstractions absolues, universelles, intemporelles et parfaitement circonscrites par leur description. Ces postulats reposent

sur des positions épistémologiques complexes portant sur la nature du langage et de la réalité qui semblent difficilement prouvables par l'observation. De plus, ils entraînent des jugements *a priori* sur la nature des signifiés lexicaux qui peuvent entraîner une impossibilité de traiter efficacement les phénomènes de polysémie complexes comme la polysémie d'acception et la polysémie d'usage.

9. Le cadre linguistique plus récent de la *sémantique différentielle* offre de travailler sur des objets linguistiques plus facilement confrontables à l'observation (notamment les signifiés et les classèmes) grâce à leur lexicalisation dans la langue.

10. Néanmoins, la sémantique différentielle repose sur une formalisation des opérations sémantiques qui s'avère difficile à transposer dans un cadre pratique de traitement automatique. Il va donc nous falloir chercher des informations plus pratiques pour produire un système opérationnel de traitement des formes de polysémie complexes.

Chapitre 3

Modèles computationnels

> «Le maître Zen nous dit qu'une idée omniprésente peut être cachée par son omniprésence même. Dans notre comportement quotidien ordinaire, nous sommes totalement égarés, en proie à une idée illusoire du moi. Chaque inclination enchanteresse de l'orgueil et de son ego, de la convention et de son maître – le dressage social – conspire à maintenir l'illusion. Le sémanticien l'appelle l'inertie des vieilles prémisses. Et c'est ce qui maintient nos analyses de la conscience à l'intérieur de limites fixes.»
>
> **Frank Herbert**, *Destination Vide*

3.1 Introduction

Au chapitre précédent, on a abordé l'aspect linguistique des ambiguïtés lexicales résultant de formes de polysémie complexes (polysémies d'acception et d'usage), que nous avons cherché à formaliser. Cette première étape descriptive n'est pas suffisante pour réaliser un système automatique de traitement de ces ambiguïtés. C'est pourquoi nous allons nous pencher dans ce chapitre sur des modèles computationnels de traitement des ambiguïtés lexicales.

Nous nous intéresserons tout d'abord aux rapports qu'entretiennent le traitement automatique des langues naturelles et la notion d'ambiguïté, afin de déterminer dans quelle mesure les ambiguïtés lexicales qui nous occupent sont ou non des artefacts des modèles de traitement qui leur sont appliqués.

Puis, afin d'illustrer la discussion qui a été développée au chapitre précédent à propos des différentes théories sémantiques, nous passerons en revue quelques systèmes classiques fondés sur les théories de la référence et de l'inférence, ainsi que des systèmes plus marginaux basés sur la sémantique différentielle.

Pour finir, nous étudierons plus en détails trois modèles computationnels de traitement des ambiguïtés engendrées par la polysémie : un modèle connexioniste, et deux *lexiques dynamiques* : le GENERATIVE LEXICON et EDGAR.

3.2 Le traitement automatique des langues naturelles et l'ambiguïté

Les exemples du §2.4.1 montrent des cas pour lesquels l'ambiguïté constitue une source de richesse sémantique. Cette situation à conduit certains chercheurs (comme Dollan et Prince) à préconiser de la conserver dans les représentations sémantiques (cf. §2.4.3). Cela va à l'encontre des stratégies habituellement appliquées en traitement automatique des langues naturelles, où l'on cherche plutôt à se débarrasser des ambiguïtés.

La stratégie qui consiste à résoudre toutes les ambiguïtés est bien adaptée à certaines catégories d'ambiguïtés, telles que celles de l'homonymie ou de la polysémie fonctionnelle. Mais son application sur des formes d'ambiguïté plus complexes n'est pas souhaitable, à moins d'être prêt à perdre des informations potentiellement importantes.

Nous pensons qu'il existe un problème d'*adéquation* entre les formalismes utilisés en informatique, ceux issus de la linguistique et les données observables. Ce problème tient d'une part aux hypothèses linguistiques simplificatrices et aux options de mise en œuvre choisies, et d'autre part au statut accordé en traitement automatique des langues naturelles aux représentations des signifiés.

3.2.1 Hypothèses et architectures

Il y a lieu en traitement automatique des langues naturelles de se méfier des *ambiguïtés artificielles*, i.e. des ambiguïtés rencontrées par les systèmes de traitement et pas par les locuteurs humains. Elles sont une conséquence des choix de mise en œuvre qui ont été pris par les concepteurs de ces systèmes.

En quoi les ambiguïtés engendrées par les formes complexes de polysémie comme la polysémie d'acception et la polysémie d'usages relèvent ou pas de la catégorie des ambiguïtés artificielles ? Pour le savoir, il faut évaluer l'impact des stratégies de conception des systèmes de traitement de la langue sur la nature des ambiguïtés auxquelles ils sont confrontés.

Modularité

Depuis la sémiotique de Morris et Carnap [77], l'étude du sens est traditionnellement séparé entre morphosyntaxe, sémantique (traitant du sens hors contexte) et pragmatique (s'occupant des effets du contexte sur le sens).

Une hypothèse largement utilisée est qu'il est possible de réaliser un traitement sériel des textes par enchaînement de modules s'étageant le long d'un axe d'abstraction : l'analyse morphosyntaxique préalable fournit les entrées du module d'analyse sémantique, dont les sorties sont enfin reprises par un module d'analyse pragmatique. Cette modularité suppose que les différentes analyses sont étanches et se suivent dans cet ordre.

Représentations sémantiques

Les représentations sémantiques sont calculées pour des unités linguistiques supposées objectives : les mots, les phrases et le texte. Le calcul est réalisé par un processus d'interprétation fondé sur deux principes essentiels :

1. *Les représentations de base sont immuables* : les significations des unités lexicales (en nombre fini) sont des constantes du système.

2. *La signification est compositionnelle* : la signification d'une unité englobante reflète la compositionnalité de sa structure syntaxique. Ainsi, en introduisant un isomorphisme entre les connexions syntaxiques et les connexions conceptuelles, on calcule le sens d'une phrase grâce à une algèbre opérant sur le sens des mots qui composent cette phrase.

Déterminisme du traitement

Originellement, les systèmes de traitement automatique des langues naturelles s'articulaient autour de traitements non-déterministes : à chaque étape de l'analyse peut apparaître un point d'indécision engendrant de multiples solutions locales. Les stratégies non-déterministes consistent à explorer exhaustivement toutes ces solutions (et les solutions globales auxquelles elles mènent) soit en largeur d'abord (traitement en parallèle), soit en profondeur d'abord (par retour-arrière).

Ces stratégies étaient considérées comme nécessaires en raison de l'ambiguïté des langues naturelles, mais des études, comme celle de Rady [98], ont montré que le non-déterminisme est en fait lié à la séparation des connaissances appliquées lors du traitement sériel : morphologie, puis syntaxe, puis sémantique et enfin pragmatique.

En fusionnant les étapes de traitement et en se munissant d'un regard en avant, il est possible de réduire le non-déterminisme : à chaque étape, l'analyseur ne conserve qu'une seule solution en éliminant toutes les impasses. C'est, par exemple, le cas du premier système CARAMEL (cf. Sabah [107, 108]) qui fusionne les traitements syntaxique et sémantique.

Cependant, cette stratégie conduit à adopter une approche de "résolution des ambiguïtés" : face à une ambiguïté, l'analyseur doit choisir quel est *le* bon sens à conserver avant de continuer. Si cette approche est opérationnelle dans la plupart des cas d'ambiguïté, elle n'est clairement pas adaptée à la polysémie d'usage.

Conclusion sur les hypothèses architecturelles

En guise de conclusion pour ces paragraphes, nous renvoyons à l'étude de Bassi Acuña [3] qui propose d'adopter les nouvelles hypothèses suivantes :

1. La séparation entre morphosyntaxe, sémantique et pragmatique n'est pas fondée objectivement, ni psychologiquement.

2. La séparation des connaissances appliquées pendant le traitement engendre des ambiguïtés artificielles (et nous rajoutons : empêche de se concentrer sur les ambiguïtés réelles).

3. Il n'est pas possible de définir de critères de filtrage sémantique applicables à des textes sur des mondes ouverts (i.e. non restreints à un domaine dont les formes d'expression sont toutes connues).

4. La phrase n'est pas une unité réelle de découpage des textes (cf. Ghiglione [43]), il n'est donc pas souhaitable de construire de représentations du sens des phrases.

5. Le sens des phrases n'est pas compositionnel par rapport aux sens des mots qui les composent (on peut aussi se reporter à Rastier [99]).

6. Les représentations sémantiques ne sont pas délimitées : le sens d'un énoncé n'est pas entièrement compris en lui-même (cf. §2.3.1.3).

7. Il n'existe pas d'isomorphisme entre l'expression sémantique d'un énoncé et son contenu sémantique.

3.2.2 Statut du lexique en traitement automatique des langues naturelles

Le problème des ambiguïtés artificielles ne tient pas uniquement à l'architecture des systèmes de traitement automatique des langues naturelles. Nous pensons que le problème se situe en grande partie dans le statut des représentations lexicales.

Les ambiguïtés lexicales sont généralement considérées comme un "faux problème", car si les locuteurs humains sont censés ne pas être confrontés aux alternatives rencontrés par les systèmes automatisés face à une ambiguïté lexicale. Cependant, la linguistique fait état d'un "sentiment d'alternative" (cf. §2.2) qui remet cette croyance en question. Nous pensons que ces deux opinions sont néanmoins compatibles. Le tort, que nous attribuons surtout aux informaticiens, est de ne considérer que les ambiguïtés lexicales qui sont produites par la polysémie fonctionnelle, qui constituent effectivement des "faux problèmes" dans la plupart des cas. Comme nous l'avons dit plus haut, la polysémie fonctionnelle se situe à mi-chemin entre l'homonymie et la stricte polysémie telle qu'on la définit en linguistique structuraliste : elle concerne des unités lexicales pour lesquelles une même forme peut correspondre à des interprétations qui n'ont que peu de choses (ou pas du tout) en commun. Or, les humains possèdent apparemment un système de représentation en mémoire qui établit comme un fait psychologique la différence entre les différentes significations de ces unités lexicales (cf. Bassi Acuña [3]). Ce système de représentation est donc tout à fait adapté à un traitement efficace et inconscient de l'homonymie et de la polysémie fonctionnelle. Si les systèmes automatiques rencontrent des alternatives pour ce genre de cas, c'est donc bien les formalismes de représentation du sens en machine qui sont en cause, et le lourd traitement des alternatives est un artefact qui n'a pas lieu d'être du point de vue de la stricte efficacité.

Le problème d'adéquation des systèmes informatiques avec l'ambiguïté lexicale tient, selon nous, aux formalismes de représentation des signifiés lexicaux en machine, notamment parce que ces formalismes se sont développés dans l'optique généralisée d'une prédominance de la syntaxe et qu'ils ont été *adaptés* à la représentation du signifié lexical, mais qu'ils n'ont pas été créés *pour lui*.

Analyse componentielle : sémasiologique ou onomasiologique

Les systèmes informatiques les plus répandus à l'heure actuelle se fondent sur une représentation analytique du signifié lexical, mais par des techniques apparemment inconciliables avec ce qui est préconisé en linguistique moderne. Dans le cas de l'intelligence artificielle comme dans le cas de la linguistique, la représentation du signifié lexical résulte d'une *analyse componentielle* qui consiste à énumérer les éléments distinctifs des signifiés, que ce soient sous forme de traits spécifiques / génériques chez Pottier [82], ou bien de *distinguishers* ou *semantic markers* chez Katz et Fodor [56]. Cependant, sous ce principe commun, les méthodes diffèrent : les linguistes adoptent une approche *différentielle* par la méthode *onomasiologique* (dirigée par les significations) : les signifiés sont analysés à travers leurs différences au sein d'une classe sémantique minimale qui n'est définie que par des critères sémantiques, sans se soucier de la forme, des signifiants. Au contraire, les informaticiens adoptent plutôt une approche *référentielle* par la méthode *sémasiologique* (dirigée par les mots), comme en lexicographie : les signifiés sont analysés par leurs différences au sein d'une classe définie par des critères d'expression. Par exemple, dans le cas du mot *assiette*, l'approche onomasiologique considérera, entre autres, deux classes de signifiés : l'une pour les couverts et l'autre pour les phénomènes d'équilibre, au sein de chacune il existe un signifié dont l'expression est *assiette* (mais qui n'ont donc aucun rapport entre eux). L'approche sémasiologique

considérera au contraire la classe des signifiés qui s'expriment tous par *assiette* et en déduira qu'elle contient un signifié *assiette [de table]* et un signifié *assiette [d'un vaisseau, d'un cavalier]* qui sont reliés par leur réalisation commune en langue. Un conflit inconciliable apparaît immédiatement entre ces deux approches : l'approche différentielle est relative, tandis que l'approche référentielle est absolue. Dans le premier cas, le nombre et la nature des composants d'un signifié correspondent aux traits nécessaires pour le distinguer clairement des autres signifiés de sa classe. Au contraire, aucun critère linguistique ne peut venir limiter le nombre de traits distinctifs d'un signifié dans l'approche référentielle, et on cherche à en épuiser les propriétés définitoires.

Les primitives sémantiques

Une confusion s'est largement répandue à propos de l'analyse en composants : beaucoup d'auteurs ont adjoint à l'analyse componentielle le critère d'utiliser des composants minimaux, universels et en nombre fini, appelés *primitives*, comme chez Schank [113], Wilks [135] et Sowa [116]. Les composants sont des éléments linguistiques, des descripteurs des signifiés relativement à des lexiques donnés, et sont donc dépendant d'une langue, d'une culture, d'un domaine d'activité. A l'opposé, les primitives sont des universaux métalinguistiques, ou qui n'ont même plus de rapport avec le langage. Il s'agit alors de catégories conceptuelles pures. Nous voulons ici mettre l'accent sur le fait que pratiquer l'analyse componentielle d'un signifié n'implique pas du tout qu'on doive recourir à des primitives (dont l'existence reste encore à établir) ; différencier un signifié de tous les autres au sein d'une classe sémantique n'a absolument pas pour conséquence de cerner son intension, c'est-à-dire de l'enfermer en tant que concept au centre d'un réseau de conditions logiques nécessaires et suffisantes.

Représentations sémantiques en intelligence artificielle

Pour représenter les signifiés lexicaux, l'intelligence artificielle propose généralement d'employer des *réseaux sémantiques* ou leurs variantes améliorées. Il s'agit de graphes dont les nœuds contiennent les composants du signifié, reliés par des arcs qui sont étiquetés par des primitives (vues comme des relations conceptuelles indépendantes de la langue). De plus, en convenant que les nœuds sont étiquetés par des signifiants non primitifs (i.e. par les mots qui servent à désigner les composants dans la langue), ils sont eux-mêmes représentables par des réseaux autonomes. L'exemple présenté sur la figure 3.1 est le réseau représentant *buy* (d'après Sowa [116, p. 110]) : les liens conceptuels sont étiquetés AGNT (agent), INIT (initiateur), INST (instrument), RCPT (récepteur), OBJ (objet), SRCE (source) et PART (partie de).

On pourra trouver une discussion détaillée sur la comparaison entre l'analyse componentielle différentielle et les réseaux sémantiques chez Rastier [100]. Nous nous contenterons de mentionner quelques caractéristiques des réseaux sémantiques qui nous semblent importantes si l'on veut expliquer les problèmes que rencontre ce type de formalisme face à la polysémie.

Problèmes posés par les réseaux sémantiques

Tout d'abord, les composants représentés dans les réseaux sémantiques semblent le plus souvent déterminés par une analyse référentielle : il s'agit de propriétés et de qualités des référents des signifiés représentés. Or, s'agissant de qualités intrinsèques, il n'existe aucun critère linguistique pour les déterminer ni en fixer le nombre. On a donc un problème de *pertinence* sémantique : pourquoi choisir telle liste de composants plutôt que telle autre ? Ce qui revient à se demander pourquoi tel réseau plutôt qu'un

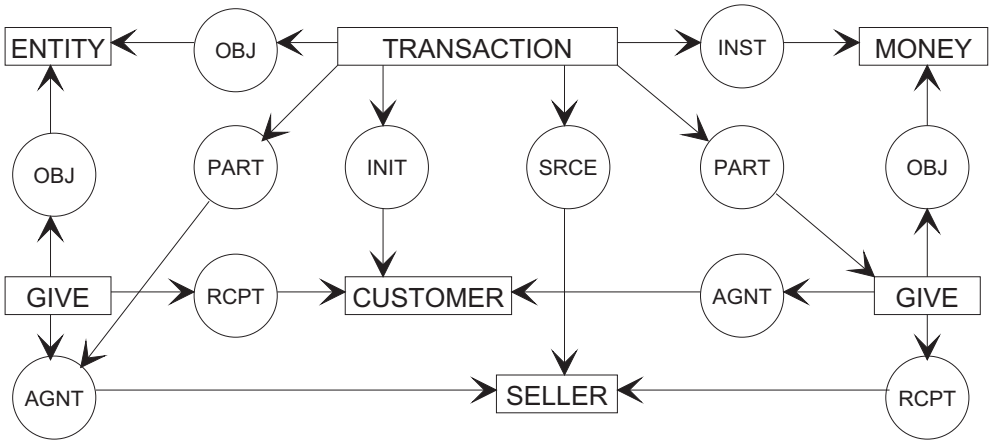


FIG. 3.1: Réseau sémantique pour *buy*

autre ? On voudrait pouvoir disposer de familles de graphes apparentés entre lesquels il serait possible de choisir le plus pertinent pour un objectif donné.

D'autre part, les réseaux sémantiques sont bien adaptés à la représentation des relations hiérarchiques entre signifiés, grâce aux relations *sorte-de* et *est-un*. Cependant, la lexicologie et la terminologie montrent depuis longtemps que les lexiques ne sont pas organisés en arborescences hiérarchiques. Il s'agirait plutôt, pour filer la métaphore, de "steppes d'arbustes entremêlés". On pourrait donc envisager de créer des arborescences limitées, elles-mêmes regroupées dans d'autres arborescences d'un ordre supérieur, à la manière des hypergraphes.

Pour améliorer les capacités des réseaux sémantiques à représenter les signifiés, certains dispositifs généralement appelés *frames* ou *schémas* leur ont été ajoutés[1]. Un frame, ou schéma, possède des *attributs* à qui on peut affecter des valeurs grâce à leurs *facettes*. Les attributs sont des traits, caractéristiques ou propriétés descriptives (et souvent envisagées comme définitoires) du schéma. Les facettes régulent l'affectation de valeurs aux attributs. Il peut s'agir de facettes déclaratives spécifiant les types de valeurs acceptables pour un attribut, parmi lesquelles on trouve également des facettes d'héritage fixant une valeur donnée pour un schéma et tous ses descendants, et les facettes de limitation qui restreignent les valeurs possibles. Mais il existe aussi des facettes procédurales qui décrivent comment calculer la nouvelle valeur de l'attribut (en la faisant dépendre d'autres attributs ou de variables), avec le cas particulier des *démons* qui sont des procédures à effectuer automatiquement lorsque l'attribut change de valeur.

Revenons maintenant au problème qui nous intéresse plus particulièrement ici, à savoir en quoi ces formalismes sont mal adaptés à la gestion de l'équivocité du lexique, et comment ils permettent de résoudre la classe des ambiguïtés lexicales produites par la polysémie fonctionnelle.

Stratification des analyses

Un premier problème important des modèles informatiques ne tient en fait pas tant aux formalismes de représentation des signifiés lexicaux qu'à leur séparation radicale des autres paliers de l'analyse linguistique. Ainsi, en séparant nettement l'analyse lexicale, l'analyse syntaxique et l'analyse sémantique, comme il va pratiquement de soi en traitement automatique des langues naturelles, on introduit une

[1]Nous utilisons ici le terme *frame* comme un générique applicable également aux variantes telles que les scripts, les prototypes et les objets des langages orientés objets.

distinction discutable entre les ambiguïtés lexicales et les ambiguïtés syntaxiques ou structurales. Or, nous avons vu (cf. §2.2 et 2.3) que les linguistes estiment plutôt faire face à un continuum d'ambiguïtés aussi bien sublexicales que lexicales, syntaxiques, structurales et rhétoriques présentant les mêmes caractéristiques (bien que nous n'ayons envisagé ici que des problèmes d'homonymie et de polysémie). La conception est pourtant omniprésente parmi les informaticiens que toute langue est en fait un empilement de niveaux différents qui seront traités par autant de modules spécialisés, chacun mettant en œuvre une composante d'une grammaire particulière. Les systèmes de traitement automatique de la langue comportent donc avec une étonnante régularité des modules spécialisés pour les composantes phonologique, syntaxique, sémantique, ainsi qu'une sorte de "fourre-tout", boîte à outils hétéroclite se réclamant de la pragmatique. Or, ce n'est pas parce qu'elle est largement acceptée que cette partition acquière automatiquement un caractère de validité. Par exemple, sur quoi se fonde la démarcation entre sémantique et pragmatique ? Outre la critique de cette organisation donnée par Bassi Acuña [3], il semble qu'est déclaré comme relevant de la sémantique tout ce qu'on peut représenter facilement avec les réseaux sémantiques ou leurs variantes. Le reste est relégué dans la pragmatique.

On voit apparaître un début d'explication au côté artificieux des ambiguïtés lexicales en traitement automatique des langues naturelles. En effet, avec une telle partition de la langue en couches, fondée sur le postulat du rôle central de la syntaxe, on voit mal comment traiter le lexique : il ne se laisse réduire ni à la syntaxe pure, ni à la sémantique pure. Et dire qu'il est à cheval sur les deux ne nous explique pas comment régler les problèmes dans les faits. Cependant, depuis les années 1970, l'effort a été mis dans tous les systèmes informatiques sur le développement d'une interaction entre les différents modules (qui correspondent aux différents niveaux linguistiques). Les systèmes les plus récents se fondent sur des modèles multi-agents "d'experts" coopérant de manière opportune pour apporter leurs connaissances aux autres modules et requérir leur aide pour tout ce qu'ils ne savent pas traiter (cf. le système CARAMEL de Sabah [107, 108, 111, 12], par exemple). Il nous semble que c'est dans cette direction qu'il faut chercher à développer une représentation fine des contenus lexicaux : tous les "niveaux" participent de l'interprétation des signifiés et c'est leurs interactions, plutôt que des connaissances figées en mémoire qui permettront d'approcher le "sens" des énoncés.

Statut du lexique

Une des caractéristiques propres aux phénomènes de polysémie est qu'une partie du sens d'un énoncé polysémique est porté par le contexte dans lequel l'énoncé s'insère (cf. §2.3.1). Or la modélisation des problèmes concernant le contexte lexical n'a pas été très poussée en intelligence artificielle, étant donné que le lexique n'est généralement pas étudié pour lui-même, mais comme moyen de représenter des connaissances. Le formalisme le plus répandu pour représenter le lexique est celui des réseaux sémantiques et de leurs variantes, et dans tous les ouvrages traitant des réseaux sémantiques, ceux-ci sont indissociables des bases de connaissances dans lesquelles sont puisées leurs étiquettes. Depuis Quillian [97], ils ont été conçus en tant que support de mémoires sémantiques. Ils sont donc faits pour stocker des connaissances figées, et leur aspect lexical est simplement dû au fait qu'il est plus pratique d'énoncer des connaissances à travers les mots de la langue. De ce fait, le lexique est perçu par les informaticiens comme un réservoir d'unités et de relations qui sont codées une fois pour toute dans un système informatique : c'est une "base de faits", la référence ultime du système pour décider de ce qui est factuel ou contrefactuel à propos du contenu des mots. Le traitement qui est appliqué à cette base de connaissances consiste alors à rechercher les données adéquates pour le traitement en cours (comme, par exemple, apparier une unité lexicale dans une phrase à analyser avec la représentation en machine d'un concept) et de simuler des inférences sur ces connaissances en parcourant le réseau : on produit des

"raisonnements" en propageant "l'attention" du système à partir des connaissances trouvées initialement, de connexité en connexité. Si l'on cherche au contraire à modéliser le lexique pour lui-même, il ne s'agit plus de rechercher des connaissances, mais de les construire à travers des procédures parfois complexes (surtout lorsqu'il faut retrouver le contenu lexical d'une occurrence particulière alors que son type n'a pas encore été identifié).

Traiter une ambiguïté (même si elle est artificielle) laisse entendre qu'il faut quelque part effectuer un choix entre différentes alternatives. Or l'intelligence artificielle, comme toute la branche référentialiste de la linguistique, postule *a priori* qu'il faut choisir *la* bonne interprétation et se débarrasser des autres, qui sont mauvaises et perturbent la compréhension, puisque le texte en cours d'analyse est considéré comme forcément univoque, au moins dans les intentions de l'émetteur. Cependant, cette dichotomie entre la bonne interprétation d'un côté et les mauvaises de l'autre est erronée pour un grand nombre de cas dans la réalité de la langue, notamment en ce qui concerne la polysémie d'usage. Dans le cas optimal, on peut déterminer une interprétation significativement plus plausible que les autres. Dans le cas général, plutôt que de se débarrasser de $n-1$ interprétations jugées irrecevables, il est plus fondé psychologiquement de chercher à hiérarchiser les n interprétations en fonction de leurs degrés de plausibilité respectifs.

Adéquation des formalismes informatiques à la polysémie fonctionnelle

Nous entrevoyons maintenant pourquoi les formalismes proposés par l'intelligence artificielle pour la représentation des signifiés sont à même de résoudre les ambiguïtés dans le cas de la polysémie fonctionnelle (au moins pour la majorité d'entre elles). Il s'agit en effet de cas où face à un ensemble d'interprétations possibles d'une unité lexicale ambiguë, une seule d'entre elle correspond effectivement à la *fonction* de l'unité dans l'énoncé où elle s'insère. Il y a adéquation entre le formalisme et les faits.

Méthodes de résolution des ambiguïtés

Deux méthodes sont traditionnellement utilisées pour "lever" l'ambiguïté dans ce cas :

— On peut utiliser des *traits de sélection*, qui sont des traits génériques du signifié marquant son appartenance à des classes sémantiques très générales (comme *animé, inanimé, humain* ou *animal*). Ces classes correspondent aux dimensions de la sémantique différentielle (cf. §2.5.2.2). Elles permettent de sélectionner la bonne acception du signifié en vertu de ce que Rastier [99] nomme la *loi d'isosémie*, i.e. la récurrence de traits sémantiques prescrite par le système fonctionnel de la langue. On cherche à repérer dans le contexte proche la présence (ou "actualisation") d'un trait générique donné sur un segment d'un énoncé. La loi d'isosémie prescrit qu'afin de conserver la cohérence sémantique de l'énoncé, ce trait générique doit être également actualisé dans les segments suivants de l'énoncé. Ainsi, dans l'énoncé *le laboratoire se trouve sur le plateau de Gif*[1], le segment *le laboratoire*, qui est une sous-catégorie de *bâtiment*, porte un trait /élément de paysage/, et le segment *se trouve* porte un trait /locatif/. En vertu de la loi d'isosémie, ces traits sont également présents dans le segment *le plateau*, et seule l'acception de *plateau*-lieu géographique est sélectionnée. Dans un énoncé comme *la tasse se trouve sur le plateau*, le segment *la tasse* porte un trait /vaisselle/ qui n'est compatible qu'avec l'acception de *plateau*-élément de vaisselle. Le trait /locatif/ n'a plus la même importance que précédemment (il est afférent, et non plus inhérent) et

[1] Exemple inspiré de Rastier [99].

n'est pas bloquant pour la sélection du bon signifié. Ce mécanisme de préservation de la cohérence des énoncés se retrouve dans la notion de *préférence* chez Wilks [135].

— On peut aussi spécifier l'appartenance des signifiés à des *lexiques scripturaux*[1]. Pour un script donné, on définit un lexique particulier dans lequel les unités lexicales possèdent des sens particuliers relatifs à ce script. Partant de ce principe, lorsqu'un script est reconnu, tous les mots de son lexique associé seront interprétés dans ces sens spécifiques. Il s'agit de quelque chose se rapprochant de la notion de *domaine* en sémantique différentielle [83] : un domaine est un groupe de *taxèmes* lié à une pratique sociale donnée, comme dans les dictionnaires lorsqu'un indicateur comme *litt.* (littérature) ou *cuis.* (cuisine) est spécifié dans une entrée. Cette définition veut qu'au sein d'un domaine donné, les unités lexicales deviennent des termes, i.e. il n'existe plus ni homonymie, ni polysémie. Cependant, le problème qu'on rencontre généralement avec de vrais énoncés est la collision ou surimposition de plusieurs domaines. Rastier (*op. cit.*) cite l'exemple hypothétique d'un texte se passant lors d'un cocktail, et dans lequel *canapé* pourrait relever aussi bien du domaine *alimentation* que du domaine *ameublement*. Une solution radicale à ce problème, et bien souvent envisagée, est de décider qu'à un texte donné correspond un unique domaine. Mais cela revient à régler le problème du choix en faisant un unique choix arbitraire global au lieu de faire une suite de choix arbitraires locaux pour chacune des ambiguïtés rencontrés. Dans le fond, rien n'a changé.

Les deux méthodes reposent sur le même principe : sélectionner des signifiés "compatibles avec le contexte" en supposant la récurrence de traits génériques à travers les énoncés. Ces traits localisent le signifié sur deux axes sémantiques différents : les dimensions ou les domaines, mais le processus est le même, et il est donc très facile de combiner les deux approches en omettant simplement de distinguer les dimensions des domaines, ainsi que le fait Dyer [30].

Enfin, l'approche connexioniste fournit également sa propre solution au problème de la polysémie fonctionnelle, au travers d'un type particulier de modèles : les modèles connexionistes étiquetés (pour une présentation détaillée, cf. Jodouin [53]). Si le traitement est particulier au connexionisme (affectation de poids aux connexions, calcul itératif, rétropropagation de l'erreur, renforcement des connexions positives et élagage des connexions négatives), les concepts de base n'en demeurent pas moins semblables à ce qui vient d'être dit. En effet, les réseaux utilisés permettent de discriminer entre les différentes acceptions d'un signifié par la sélection progressive d'un neurone formel appartenant à une couche cachée spécialisée et jouant le rôle de "contexte" de désambiguïsation. Or, dans les exemples de ce type, ces contextes correspondent à la définition des domaines sémantiques.

Une autre approche

En fin de compte, il nous semble important de prendre en compte les propositions de Prince [86, 88] :

— La polysémie ne se réduit pas à la polysémie fonctionnelle. Les mots courants constituent l'exemple le plus flagrant de cette assertion, ce à quoi nous rajoutons que même des mots d'origine plus techniques peuvent également être affectés (cf. les transferts d'usage et les mots jokers, §2.4.1).

— La polysémie (au sens large) semble être une caractéristique fondamentale du lexique et les ambiguïtés qui en résultent sont plus souvent pertinentes que gênantes. Il est donc souhaitable de représenter en machine de véritables lexiques de signifiés intégrant cette polysémie naturelle dans

[1] Les scripts sont une variante des schémas (cf. p 70).

leur formalisme. Le modèle de lexique EDGAR de Prince (*op. cit.*) est une preuve de la faisabilité informatique d'un tel lexique.

3.3 Quelques modèles computationnels classiques

Afin d'illustrer l'utilisation des paradigmes sémantiques classiques en traitement automatique des langues naturelles (cf. §2.5.1.1 et 2.5.1.2), nous présentons quelques systèmes connus de traitement des ambiguïtés lexicales. Il ne s'agit pas à proprement parler de systèmes de traitement de la polysémie, ni *a fortiori* de la polysémie d'usage, mais ce sont des systèmes qui font référence dans le domaine du traitement automatique des langues naturelles.

3.3.1 Le Word Expert Parser

Le *Word Expert Parser* (WEP) de Small, Rieger et Adriaens [1] est basé sur les études de Small concernant l'ambiguïté [114]. L'idée fondamentale développée dans ce système est que si l'on veut pouvoir interpréter un mot dans un nombre de contextes aussi grand que possible (sous-entendu : plus le nombre de contextes autorisé est grand, plus il y aura d'ambiguïtés sur l'interprétation de ce mot), il faut disposer d'informations morphologiques, syntaxiques, sémantiques et pragmatiques le concernant.

De ce fait, le système WEP fonctionne comme un système expert (d'où son nom) pour l'interprétation lexicale. Il dispose d'une base de connaissances qui sont des règles d'interprétation pour chaque mot du lexique susceptible d'ambiguïté. L'ensemble des connaissances portant sur un mot se présente fonctionnellement sous forme arborescente, où les noeuds intérieurs correspondent à des règles (des demandes d'information à des modules externes) qui permettent d'effectuer un choix dans le parcours interprétatif de l'arbre. Chaque feuille de l'arbre représente une signification en contexte du mot, et seule une des feuilles peut être atteinte au cours d'un parcours interprétatif de l'arbre, étant donné que les embranchements intérieurs correspondant aux prises de décision (i.e. à l'application des règles) sont mutuellement exclusifs. Cette représentation des connaissances est illustrée sur la figure 3.2.

Ce système relève à la fois de la sémantique référentielle, étant donné que les feuilles des réseaux de règles ont un statut de concepts, et de la sémantique inférentielle à travers les processus inductifs qui permettent de prendre des décisions lors du parcours interprétatif des arbres. Cela le conduit à devoir trouver *la meilleure interprétation* pour un mot ambigu donné (puisque les inférences utilisées engendrent des alternatives mutuellement exclusives), et le disqualifie pour le traitement de la polysémie d'usage (nous avons vu au §2.4.2 que la polysémie d'usage se caractérise par une présence générale de *plusieurs* interprétations simultanément). D'autre part, le statut conceptuel des interprétations du WEP rend difficile la représentation des emplois des signifiés, car le recours implicite à une définition intensionnelle de ces signifiés cadre mal avec la sous-détermination irréductible des emplois.

3.3.2 Les Polaroid Words

Les *Polaroid Words* (PW) de Hirst [50] permettent, au sein d'un système de compréhension du langage, à un module d'analyse syntaxique et à un interpréteur sémantique de travailler de concert même lorsque des ambiguïtés lexicales obligent le système à repousser l'analyse syntaxico-sémantique d'un mot jusqu'à l'arrivée d'informations supplémentaires. En régime normal, l'interpréteur sémantique construit des représentations sémantiques des mots aussitôt que leur analyse syntaxique est complète. Dans le cas d'une ambiguïté lexicale, cette analyse ne peut être menée à terme immédiatement. L'analyseur

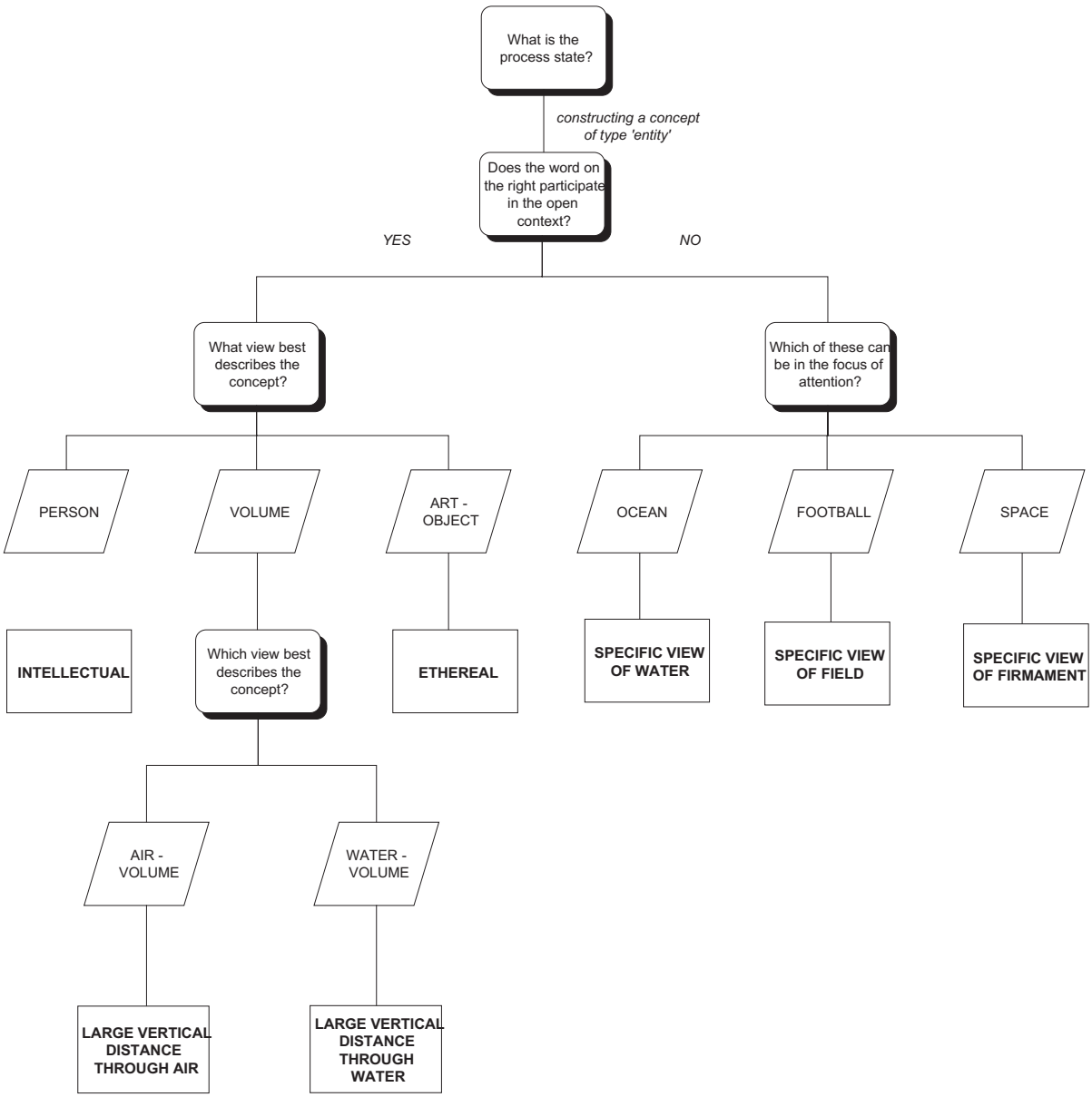


FIG. 3.2: Réseau de règles WEP pour *deep* (tiré de [1])

syntaxique soumet donc à l'interpréteur sémantique un ersatz d'analyse (un *Polaroid Word*) contenant les informations qu'il a pu rassembler jusque là. Lorsque de nouvelles informations sont disponibles et peuvent compléter l'analyse du mot en question, elles sont ajoutées dans le PW. D'où le terme *polaroid* : ces structures de données se "révèlent" petit à petit, comme des photos polaroïds, et le système peut commencer à faire des inférences avant que le "développement" soit terminé.

Le système utilise deux mécanismes simultanément pour "développer" les PW : la communication entre PW au sein d'une zone commune d'échange (une ébauche de tableau noir), et un mécanisme de propagation d'activation par transmission de marqueurs entre mots associés.

Les PW sont des structures de données semblables aux *frames* (cf. p 70). Leurs attributs correspondent aux propriétés définitoires du concept qu'ils représentent. On se retrouve dans le cadre de la sémantique référentielle, avec la formalisation classique schéma/attributs de l'intension des concepts. Les PW représentant les noms sont assez simples : ils ne contiennent que la liste de leurs interprétations possibles. Les PW pour les prépositions et les verbes sont plus complexes : ils sont composés de couples attribut/prédicat, où les prédicats opèrent comme des conditions de *restriction de la sélection* (cf. p 25) des interprétations compatibles avec le contexte[1]. La communication entre les PW sert à supprimer des possibilités d'interprétation au sein des PW non encore entièrement développés, suite à une contradiction entre une possibilité d'interprétation d'un PW et une information contenue dans un autre PW dans le cas de relations particulières verbe/noms-prépositions, préposition/noms-prépositions, modificateur nominal/noms, etc. (en fait, toutes les relations syntaxiques où l'un des PW régit d'autres PW).

Le mécanisme de transmission de marqueurs est conduit parallèlement à la communication entre PW. Il s'agit d'une tentative de formalisation des phénomènes d'amorçage sémantique observés en psychologie cognitive (cf. Swinney [123]) : lorsqu'un mot est présent dans le contexte, les mots qui lui sont directement associés dans le réseau sémantique du système sont activés (imitant par là la communication entre mémoire de travail et mémoire à long terme chez l'humain). La transmission est reconduite sur une profondeur de n associations dans le réseau, avec n petit par rapport à la taille du réseau sémantique. Si une interprétation possible d'un PW est ainsi activée par un marqueur, elle est choisie préférentiellement aux autres, ce qui court-circuite le processus de communication entre PW.

Ce type de fonctionnement est surtout adapté à la polysémie fonctionnelle, mais pas à la polysémie d'usage. Cela est encore une fois dû à la conception atomiste de l'interprétation qui est employée : un PW pleinement développé consiste en une seule interprétation sélectionnée.

3.3.3 Le système CARAMEL

Le système CARAMEL développé au LIMSI est une architecture multi-agents pour la compréhension du langage naturel. Cette architecture est composée de modules experts dans des domaines particuliers partageant leurs informations. La première version de ce système, CARAMEL-1 (cf. Sabah [107, 108]) comportait un analyseur syntaxique déterministe relié à une base de graphes conceptuels (cf. Sowa [116]), un dictionnaire de 15.000 formes, un module "gestionnaire de thèmes" (qui permet de résoudre les problèmes relevant de la pure homonymie), un module "gestionnaire de références" (qui permet de traiter les anaphores et les ellipses) et un correcteur orthographique. Cette architecture a évolué vers CARAMEL-2 (cf. Sabah *et al.* [109, 111, 12, 110]) comportant un analyseur par *charts* basé sur la grammaire LFG (cf. Kaplan [55]) et une adaptation des autres modules de l'architecture précédente.

[1] Ici *contexte* est à prendre dans un sens très restreint, puisqu'il s'agit uniquement des informations sur le PW présentes dans la base de connaissances sémantique du système, ainsi que du contenu des PW précédents dans la phrase en cours.

De par sa composition, CARAMEL est un exemple typique de la complémentarité des paradigmes référentiel et inférentiel proclamée par Sperber et Wilson [117] : la modélisation des informations sémantiques par les graphes conceptuels se situe nettement dans le paradigme référentiel (même si la mécanique des graphes conceptuels est munie d'opérations qu'on peut assimiler à des inférences), tandis que la grammaire LFG fait très largement appel aux principes du paradigme inférentiel.

L'originalité marquante de CARAMEL-2 est que ces différents modules coopèrent à l'aide d'un "carnet d'esquisses" (*sketchboard*, cf. Sabah [111]), qui est un type sophistiqué de tableau noir. Grâce au carnet d'esquisses, il est possible de simuler des processus mentaux à la fois au niveau réflexe (inconscient) et au niveau contrôlé (conscient). Ce type d'architecture est prometteur pour le traitement automatique des langues naturelles car il autorise d'une part des inférences non déductives (des associations non rationnelles), et particulièrement parce qu'il laisse un degré de liberté suffisant pour se passer d'une sémantique référentielle absolue (tout comme le préconisait Putnam, cf. p. 44, il s'agirait alors d'une sémantique dénotationnelle qui s'assume en tant que telle). Bien entendu, cela ne suffit pas pour s'affranchir de la tripartition syntaxe / sémantique / pragmatique, mais cela permettrait à des modules qui se placent en dehors de ce cadre de coopérer entre eux, et éventuellement d'arriver à une représentation (au moins indirecte) de ce que sont les signifiés lexicaux.

3.4 Un système de traitement de la polysémie d'acception

Les cas d'ambiguïtés mettant en jeu des acceptions ont fait l'objet d'une ébauche de modélisation informatique par Brugère [16] en se plaçant entièrement dans le cadre de la sémantique componentielle dynamique de Rastier [99]. Le but de ce travail de modélisation était de construire un système automatique capable de fournir une décomposition en traits sémantiques pour une occurrence d'un mot polysémique de façon à privilégier clairement une acception-type par rapport aux autres. Ce but devait être atteint sur un corpus ouvert, et pas seulement sur les exemples ayant servi à construire la base de connaissances du système, à savoir les dénominations de relations de parenté proche (*père, mère, fille, garçon, ...*) observés sur un corpus contenant *Les Mots* de J.-P. Sartre et *Introduction à la psychanalyse* de S. Freud.

Le traitement effectué par le système résultant comporte trois étapes :

1. *Sélection d'un signifié-type.* Il s'agit d'identifier l'acception-type de l'occurrence en se basant sur le contexte linguistique proche, représenté par les résultats fournis par un analyseur morpho-syntaxique (signes typographiques, formes canoniques des mots et traits syntaxiques). Cette étape se décompose en trois phases correspondant à des degrés de confiance sur les informations fournies par le contexte :

   (a) Les indications les plus sûres qu'on peut trouver dans le contexte sont des syntagmes lexicaux décisifs. Ce sont des formes lexicales figées qui permettent d'identifier de manière certaine une acception particulière. Par exemple, pour le mot *garçon*, la forme lexicale *petit garçon* identifie immédiatement l'acception 'enfant', tandis que la forme *rester garçon* identifie l'acception 'célibataire'.

   (b) En l'absence de telles formes remarquables, le système cherche à reconnaître des formes phraséologiques autour du mot à analyser, qui vont permettre de favoriser certaines acceptions ou d'en écarter d'autres improbables en fonction de la structure syntaxique reconnue. Ainsi, parmi les règles du système figure le fait qu'une structure *garçon Adj* ou *Adj garçon*

favorise les acceptions appartenant à la catégorie connue par le système sous le nom ‘évaluation’. Ce genre de règles suppose que la signification d’un syntagme dépend en partie de sa morphologie, et correspond à des observations sur corpus. En cas d’ambiguïté, i.e. lorsque plusieurs formes syntagmatiques sont reconnues, le système privilégie la forme la plus précise. Par exemple, il privilégiera la forme *garçon de N Adj* (comme *garçon de taille moyenne*) correspondant à une acception de type ‘évaluation’ à la forme *garçon de N* (comme *garçon de café*) qui correspond plutôt à une profession ou une fonction.

(c) Lorsque les deux précédentes phases ne fournissent pas d’élément décisif, le système cherche à identifier la “classe contextuelle” du signifié-type en s’appuyant sur les signifiés qui l’entourent. Ainsi, rencontrer un antonyme d’un signifié-type donné permet de privilégier ce dernier pour une réponse ultérieure : *fille* pour *femme*, *fille* pour *garçon* ou *homme* pour *garçon*. De même, la présence d’un signifié qui appartient à un même script qu’un signifié-type donné permet de favoriser ce dernier, comme la présence de *patron* et *café* indiquera que l’occurrence de *garçon* correspond à ‘garçon de café’, alors que la présence de *père* ou *mère* favorisera plutôt l’acception ‘enfant’.

2. *Interprétation d’acception.* La seconde étape du traitement consiste à activer ou à désactiver certains traits sémantiques en s’appuyant sur le contexte. Il est ainsi possible de modifier lors de cette étape la décomposition en traits sémantiques obtenue au sortir de l’étape précédente pour tenir compte des modifications apportées par le contexte. Cela permet de traiter des exemples tels que ce vers de Racine *Un père en punissant, Madame, est toujours père* où l’activation du trait /éducateur/ dans la première occurrence de *père* entraîne l’activation du trait associé /bienveillant/ dans la seconde. Pour faciliter cette tâche, l’auteur du système a choisi une représentation des signifiés sous forme d’arborescences d’acceptions plutôt que sous forme de faisceaux de traits sémantiques. De cette façon, une acception correspondra à un ensemble de traits aux valeurs prédéfinies. Cette étape de modification des faisceaux de traits en fonction du contexte est ainsi rendue indissociable de la phase d’ordonnancement des acceptions selon leur plausibilité : les règles d’attribution d’une plausibilité ne sont pas séparées des règles de modification *a posteriori* des traits sémantiques. Pour ce faire, le système se base sur un calcul probabiliste inspiré du système MYCIN pour attribuer des coefficients de vraisemblance aux différentes acceptions à partir des indices recueillis précédemment.

3. *Génération de la réponse.* Cette étape consiste à fournir le résultat de l’analyse des traits des acceptions en langage naturel.

La figure 3.3 présente une arborescence d’acceptions citée en exemple par l’auteur pour *garçon*. Dans chaque boîte, la première ligne correspond à l’étiquette de l’acception, tandis que les lignes suivantes représentent les règles de filtrage utilisées comme indice.

Ce système est le seul, à notre connaissance, à proposer une tentative de traitement de la polysémie d’acception considérée comme un phénomène en tant que tel, et non pas à essayer de la faire rentrer de force dans le cadre de la polysémie fonctionnelle par le truchement de représentations syntaxiques des métonymies, hypallages et autres figures de rhétorique.

L’esprit dans lequel il a été conçu est clairement celui d’un modèle visant à une solution unique, même si ses sorties sont le plus souvent des listes de propositions ordonnées. Au final, il reste quand même à choisir *la* bonne acception pour une occurrence donnée. Il ne s’agit pas, selon nous, d’une “faille” du modèle, mais d’une caractéristique inhérente au phénomène modélisé lui-même (cf. §2.3.3).

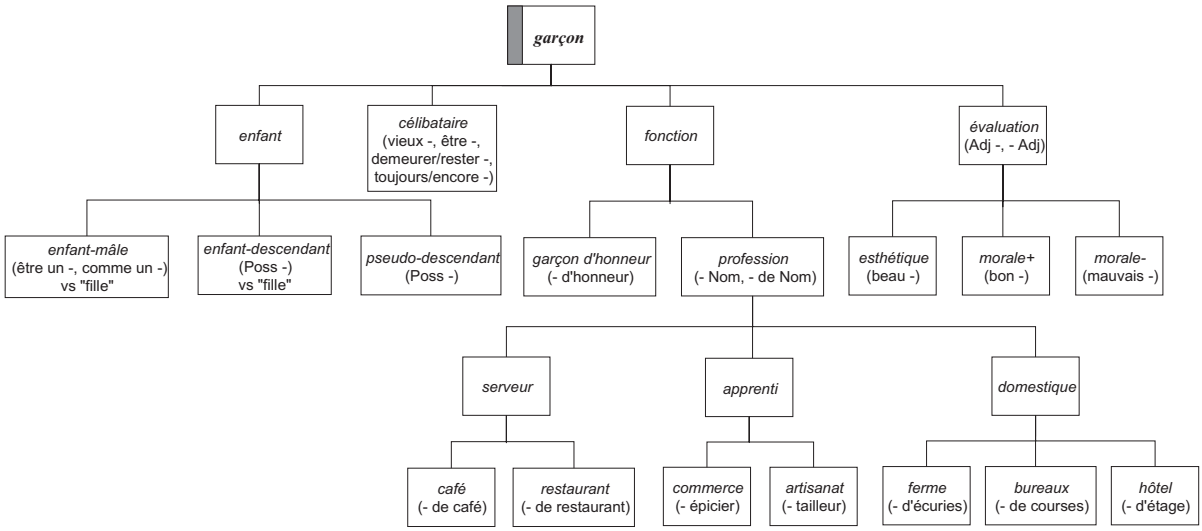


FIG. 3.3: Arborescence d'acceptions pour *garçon* (tiré de [16])

Alors que dans le cas de la polysémie fonctionnelle, le processus de traitement consiste clairement à sélectionner un unique sens compatible avec le contexte dans une approche de type *winner takes all*, la polysémie d'acception engendre des situations où les choses ne sont plus si tranchées, et l'on ne peut plus proposer qu'une liste ordonnée d'acceptions candidates à l'interprétation, sachant que l'une est vraisemblablement mieux adaptée que les autres.

3.5 Un système connexioniste de traitement de la polysémie

Les travaux de Victorri et Fuchs sur la polysémie les ont conduit à proposer un modèle morphodynamique du traitement de la polysémie, qui a été mis en œuvre par un réseau connexioniste (Fuchs et Victorri [40], Victorri [129]). Ce modèle s'inspire des travaux du mathématicien Thom [125, 126, 127] sur la morphogénèse. Il se place dans un cadre différentiel étendu grâce au recours aux mathématiques du continu. Cela permet de représenter des différences graduelles (de type gradient) tout en conservant la possibilité de coder certaines caractéristiques discrètes (en utilisant des équations aux différences finies).

3.5.1 Le modèle morphodynamique

A toute expression polysémique sont attachés un *espace sémantique* (l'espace des sens potentiels de l'expression) et un *espace co-textuel* (les contraintes imposées sur l'interprétation de l'expression par les éléments textuels environnants). Ces deux espaces sont modélisés par des *variétés différentiables*, qui sont une généralisation de la notion de surface à n dimensions (cf. Morlet [76] et Chenciner [17]).

L'interprétation de l'expression polysémique est réalisée grâce à une *dynamique* induite sur l'espace sémantique S. A tout point p de l'espace co-textuel C correspond une fonction réelle U_p définie sur S : le *potentiel sémantique* induit par C sur S. Cette fonction potentielle induit un champ vectoriel sur S (une dynamique) qui est en tout point défini par l'opposé du gradient de U_p. Cette dynamique généralise la notion de "paysage" à n dimensions : elle contient des pics et des bassins. Le minimum de chaque bassin est un *attracteur*. Il s'agit d'une application de la théorie des catastrophes de Thom (*op. cit.*) : S

joue le rôle d'espace d'état et C celui d'espace de contrôle. Les frontières entre les différents bassins de la dynamique constituent l'ensemble de catastrophe associé au modèle.

Victorri appelle *modèle sémantique* de l'expression considérée les deux espaces S et C ainsi que l'application dynamique F, suffisamment régulière, de C dans l'espace fibré tangent des champs vectoriels sur S, telle que tout point p de C définisse une dynamique F_p dérivant du potentiel U_p (avec F_p de classe C^∞ sur S). Afin d'éviter d'obtenir une situation "chaotique" (présentant des catastrophes généralisées), on adjoint à F les conditions de régularité suivantes :

— Il existe un ouvert V partout dense dans C tel qu'en chaque point $p \in V$, F_p ne possède qu'un nombre fini d'attracteurs et que la réunion de leurs bassins soit partout dense dans S.

— Chaque attracteur est structurellement stable (au sens de Thom).

— Le complémentaire de V dans C est un ensemble stratifié (au sens de Thom).

La *configuration sémantique* d'une expression polysémique est une forme de dynamique (nombre et topologie des bassins d'attraction). Une expression est polysémique s'il y a plusieurs configurations sémantiques suivant la région de C dans laquelle se situe l'énoncé.

Le *comportement sémantique* d'une expression est caractérisé par les différentes régions de C qui conduisent à des configurations sémantiques différentes. Chaque attracteur de la dynamique constitue un sens possible de l'expression.

Le sens d'une expression en contexte est une région de l'espace sémantique. Pour trouver cette région, il faut fixer un *seuil d'admissibilité* qui est une altitude limite du potentiel au-dessus de laquelle les attracteurs ne sont pas pris en compte. Le seuil d'admissibilité est fixé empiriquement par référence à des *valeurs typiques* d'interprétation de l'expression attestées sur des corpus de référence.

Avec ce modèle, un cas d'homonymie franche correspond à une région d'interprétation formée de sous-espaces disjoints de S, un cas de polysémie équivalente à une homonymie (la polysémie fonctionnelle en somme) à une région d'interprétation formée de sous-espaces contigus mais ne partageant aucun bassin d'attraction, et un cas de polysémie complexe (polysémie d'acception ou d'usage) à une région d'interprétation formée de sous-espaces contigus partageant des bassins d'attraction.

3.5.2 La mise en œuvre connexioniste

Ce modèle morphodynamique est en fait un système dynamique qui peut être mis en œuvre (pour une expression donnée) à l'aide d'un réseau connexioniste constitué d'une couche d'entrée unidirectionnelle (de type perceptron) et d'une couche de sortie récurrente (de type réseau de Hopfield [51, 52]).

Ce réseau a été mis en œuvre pour le mot *encore*. Il est contrôlé par une loi de fonctionnement hybride constituée pour une part d'une équation aux différences finies renseignée par une analyse statistique d'un échantillon de phrases représentatives, et pour l'autre part par un processus d'apprentissage par essai-erreur sur un corpus d'apprentissage.

Le système résultant est muni d'un mini-analyseur afin de coder une phrase entrée par l'utilisateur sur la couche d'entrée du réseau. En sortie, un mini-générateur fournit un ensemble d'interprétations possibles composées chacune d'un certain nombre de paraphrases acceptables (choisies parmi une vingtaine de paraphrases de *encore*).

3.5.3 Critique

Ce modèle morphodynamique est particulièrement puissant puisqu'il permet de traiter tout le continuum des ambiguïtés lexicales s'étendant de la franche homonymie jusqu'aux formes de polysémie complexes. De plus, il permet de s'abstraire dans une certaine mesure des formalismes de description linguistique en gérant aussi bien les différences continues que les différences finies.

Nous ne souhaitons pas ici nous attarder sur les performances interprétatives de ce modèle et de sa mise en œuvre mais sur des problèmes plus pratiques qu'ils soulèvent.

— Tout d'abord, le modèle morphodynamique repose sur des mathématiques assez complexes, qui peuvent poser des problèmes non négligeables à un non spécialiste (par exemple un linguiste) lors de la constitution des espaces sémantiques et co-textuels.

— D'autre part, la mise en œuvre connexioniste implique une opacité structurelle non justifiée (cf. p. 58) : il est difficile de "voir" ce qui se passe à l'intérieur du réseau. Victorri et Fuchs (*op. cit.*) évoquent d'ailleurs la difficulté de la méthode d'*extraction de règles* (cf. Gosselin *et al.* [45], et Konfé *et al.* [60]) qui permet de retrouver la loi de fonctionnement du réseau.

— Et finalement, ce modèle donne lieu à une approche *Word Expert* : il faut construire un réseau pour chaque unité du lexique. Sachant la taille que peuvent atteindre les réseaux connexionistes (souvent de l'ordre de n^2 cellules pour coder n informations) et leur temps d'apprentissage, cette méthode s'annonce coûteuse à la fois en temps et en mémoire.

3.6 Lexiques dynamiques

Les lexiques statiques ont une taille prohibitive

De nombreux systèmes de traitement automatique des langues naturelles s'appuient sur des lexiques uniquement statiques. Les plus connus d'entre eux sont le WEP (Adriaens et Small [1]) et le WORDNET (Miller *et al.* [75]). Ces lexiques abordent le problème de la multiplicité des sens d'une unité lexicale en enrichissant sa description, ainsi que les descriptions des autres unités qui peuvent lui être reliées. Par exemple, le WORDNET part d'une relation fondamentale de *synonymie contextuelle* pour définir le(s) sens d'une unité lexicale comme un ensemble de synonymes relatifs (un *synset*). Les différents synsets sont ensuite reliés les uns aux autres par des relations d'hyponymie, d'antonymie, de méronymie, d'implication et de dérivation morphologique. En outre, le WORDNET se constitue en thésaurus, i.e. il structure son contenu selon une arborescence conceptuelle. Il existe donc une nomenclature qui fait office de "squelette sémantique" à l'intérieur de la base de connaissances lexicales.

Cette approche est susceptible d'engendrer une base de connaissances énorme. Il semble que cela soit le prix à payer pour une conception statique du lexique : la richesse sémantique ne peut être prise en compte que par la prolifération des données, représentations et relations. Or, de nombreux travaux ont montré que la richesse et la complexité de la sémantique lexicale n'impliquait pas obligatoirement d'employer des représentations aussi extensives ; celles-ci pouvaient être remplacées de manière judicieuse par des mécanismes de raisonnement. A cet égard, on pourra citer Stallard [118], un des pionniers en la matière. Il fut en effet un des premiers à suggérer d'employer, pour représenter les signifiés lexicaux, un noyau sémantique minimal associé à des traits dynamiques permettant d'inférer les différentes relations lexicales, plutôt que de les représenter.

Principes des lexiques dynamiques

Il s'agit du principe de base des *lexiques dynamiques* : ne représenter de manière statique qu'une fraction minimale de sens, et laisser le soin à des mécanismes d'inférence de construire en temps voulu les significations en contexte.

Outre cet aspect structurel, les lexiques dynamiques induisent d'autres changements par rapport aux lexiques statiques :

— Les lexiques statiques sont un "cliché" de la langue à un moment donné, et ne peuvent rendre compte de la *créativité* du lexique : un emploi non répertorié dans le lexique n'est tout bonnement pas interprété.

— Le traitement des ambiguïtés lexicales est ramené à l'interprétation dynamique en contexte d'une entrée lexicale. De ce fait, il n'y a plus de traitement des ambiguïtés lexicales à proprement parler.

— Les lexiques dynamiques sont généralement construits dans une perspective onomasiologique, i.e. les données sont structurées en fonction des entrées lexicales. Cela permet notamment d'éviter les découpages conceptuels *a priori* qui engendrent la rigidité des lexiques statiques.

Différentes catégories de lexiques dynamiques

Deux familles de travaux récents reprennent les principes des lexiques dynamiques :

1. La première se caractérise par l'emploi d'un lexique complet et multidimensionnel (à plusieurs "facettes"), au sein duquel la partie statique se compose de structures descriptives minimales, et la partie dynamique de règles d'inférence réalisant des "coercions" de types lexicaux. C'est le cas du GENERATIVE LEXICON de Pustejovsky [95] sur lequel nous allons revenir plus longuement (cf. §3.6.1), ainsi que du système de Briscoe et Copestake [14] utilisant des jeux de traits et muni de règles d'interprétation par défaut.

2. La seconde se compose de lexiques différenciés en fonction de leur utilisation finale dans le système de traitement automatique des langues naturelles qui les supervise. On se référera par exemple au système de McRoy [73] qui fournit, dans une démarche client/serveur, des données lexicales spécialisées à des modules experts au sein d'une architecture de désambiguisation du sens.

Enfin, nous voudrions citer un dernier travail, assez atypique au sein des lexiques dynamiques. Il s'agit du modèle EDGAR de Prince [88, 86, 92] qui emprunte un certain nombre de caractéristiques aux travaux cités précédemment mais qui possède en outre deux caractéristiques originales :

1. Bien qu'utilisant en interne des représentations ensemblistes, il est facilement interfaçable avec des systèmes réclamant des représentations atomiques (en grande partie compatible avec un formalisme à base de graphes conceptuels).

2. La métaphore ne dispose pas d'un statut à part, elle est traitée de façon aussi naturelle que le "sens littéral" (ce qui corrobore implicitement notre décision de ne pas différencier un "sens littéral" des "sens dérivés").

Nous reviendrons plus en détails sur ce modèle au §3.6.2.

3.6.1 Le GENERATIVE LEXICON

Le principe de base de ce modèle est la *construction* des expressions sémantiques correspondant à la signification des mots *en contexte* grâce à un nombre fixé de mécanismes *génératifs*. Ces mécanismes opèrent sur un ensemble de noyaux sémantiques minimaux (dotés d'une structure interne plus riche que dans les lexiques traditionnels). En utilisant certains processus de composition, on obtient un ensemble étendu de significations lorsque les unités lexicales individuelles sont considérées relativement aux autres unités dans les phrases.

Chaque entrée lexicale permet d'encoder un certain nombre d'aspects représentatifs de ses signifiés lexicaux. Ces composants sémantiques définissent simplement les limites sémantiques appropriées aux usages de cette entrée.

D'après une argumentation de Pustejovsky [94], ces aspects représentatifs doivent s'étager sur quatre niveaux de représentation :

1. La *structure argumentative* qui fait correspondre un mot à une fonction, et la réalisation syntaxique de ce mot au nombre et au type des arguments repérés syntaxiquement et utilisés par sa représentation sémantique.

2. La *structure événementielle* qui identifie le type d'événement d'un verbe ou d'une phrase. Il s'agit d'une structure ternaire composée du type d'événement (état, processus ou transition), du foyer de l'événement et des règles de composition avec les autres événements.

3. La *structure des qualiae* définit les attributs essentiels des objets, événements et des relations associés à une unité lexicale. Les qualiae peuvent être vus comme des prédicats prototypiques et des relations associés à une entrée lexicale.

4. La *structure d'héritage lexical* détermine de quelle façon un mot est relié à un autre mot dans le lexique.

Un certain nombre de mécanismes génératifs connectent ces quatre niveaux afin d'engendrer une interprétation en contexte des mots. Le plus important de ces mécanismes est une transformation sémantique nommée *coercion de type*, comme dans les langages de programmation. Tout comme en λ-calcul, la coercion de type est une transformation qui permet de passer d'un langage de représentation monomorphique à un langage implicite utilisant des types polymorphiques.

En fait, toute la logique de ce modèle est basée sur la notion de type, et ce sont les règles de bonne formation des types qui assurent la cohérence et la validité des représentations construites. Toutes les unités lexicales figurant dans le lexique sont fortement typées, mais peuvent aisément s'adapter à de nouveaux environnements de types grâce à la coercion.

La structure des qualiae représentant l'originalité majeure de ce modèle, nous allons la présenter brièvement.

3.6.1.1 Les qualiae

Cette structure permet de décrire la sémantique des noms, au même titre que la structure argumentative des verbes.

La structure des qualiae d'un nom spécifie quatre aspects de son sens :

1. *Rôle constitutif* : la relation entre un objet et ses constituants ;
2. *Rôle formel* : ce qui le distingue au sein d'un domaine plus vaste ;
3. *Rôle téléologique* : son but et sa fonction ;
4. *Rôle agentif* : les facteurs impliqués dans son origine ou sa réalisation.

Certains de ces rôles se rapprochent des descripteurs utilisés par de nombreux chercheurs (par exemple Wilks [135]). Les rôles exprimés ainsi déterminent une description sémantique minimale d'un mot avec des conséquences à la fois sémantiques et syntaxiques.

Copestake et Briscoe [23] notamment, utilisent la structure de qualiae comme des ensembles de contraintes de typage. Leur système doit alors contenir une hiérarchie de types assez semblable à un treillis conceptuel, comme l'exemple de la figure 3.4.

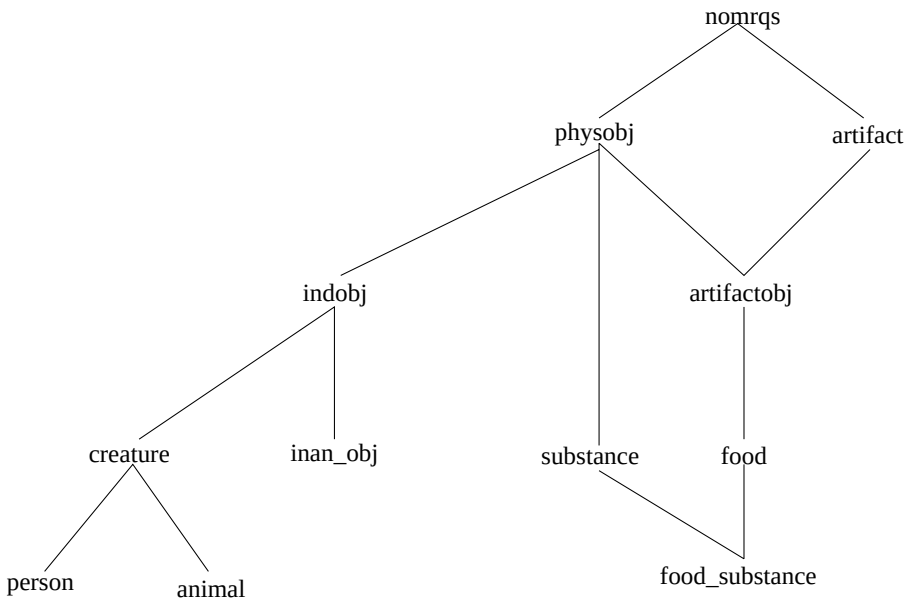


FIG. 3.4: Fragment d'une hiérarchie de types (extrait de [23])

Les contraintes sur les types **artifact** et **physobj** sont données ci-dessous[1] (avec une liste partielle de qualiae pour simplifier l'illustration) :

$$\begin{bmatrix} \textbf{artifact(x)} \\ \text{TELIC} = \textbf{Pred(E, x, y)} \end{bmatrix}$$

$$\begin{bmatrix} \textbf{physobj(x)} \\ \text{FORMAL} = \textbf{physform(x)} \\ \text{PHYSICAL-STATE} = \textbf{solid(x)} \end{bmatrix}$$

Pred est un sous-type de **formula**, et impose donc que tout sous-type de **artifact** appliqué à un objet x doit posséder une relation s'exprimant par une formule **Pred** (pour un événement de type E) portant sur l'usage ou la fonction de x.

[1] Tiré de Pustejovsky et Boguraev [95].

Grâce à la propagation de ces contraintes (par l'héritage des types vers leurs sous-types), on comprend comment une phrase telle que *Mary finishes her sandwich*[1] reçoit une interprétation par défaut : le nom *sandwich* contient des informations sur l'"activité de manger" en tant que contrainte sur son rôle téléologique, en raison de sa position dans la structure de typage.

$$\begin{bmatrix} \textbf{sandwich(x)} \\ \text{CONST} = \textbf{\{bread, ...\}} \\ \text{FORMAL} = \textbf{physobj(x)} \\ \text{TELIC} = \textbf{eat(P, w, x)} \\ \text{AGENTIVE} = \textbf{artifact(x)} \end{bmatrix}$$

Ainsi, **eat(P, w, x)** dénote un processus P établi entre une entité w et un objet physique x.

3.6.1.2 Coercion de type

Le contenu sémantique est ramené, dans le GENERATIVE LEXICON, à une expression fortement typée en λ-calcul. Mais un typage fort conduit inéluctablement à un échec en interprétation de pratiquement n'importe quelle expression en langage naturel, par violation des contraintes de sélection sur ses sous-type. C'est pourquoi l'opération de *coercion de type* a été introduite, afin d'assouplir ces contraintes.

Cette opération repose sur l'hypothèse que toute expression (en λ-calcul) α possède deux ensembles associés :

— l'ensemble Σ_α des *alias* de α (cet ensemble contient des opérateurs modifiant le type et la dénotation de α) ;

— et un ensemble Σ'_α d'opérateurs qui transforme le type de α en un surtype.

Cela permet d'obtenir des variantes de l'application fonctionnelle capables de changer le type d'un argument (vers un type alias, ou vers un surtype) pour éviter les erreurs de typage. De ce fait, le langage servant à dénoter la sémantique des expressions devient polymorphique, et donc plus souple.

Par exemple, sans la coercion de type, l'expression *to begin a book* provoquerait une erreur de typage puisque *begin* attend normalement un complément nominal exprimant un événement, et pas un objet physique. Or, si l'on dispose de la structure de qualiae suivante pour *book*[2] :

$$\begin{bmatrix} \textbf{book(x)} \\ \text{FORMAL} = \textbf{physobj(x)} \\ \text{TELIC} = \textbf{read(P, y, x)} \\ \text{AGENTIVE} = \textbf{write(T, w, x)} \end{bmatrix}$$

il y a coercion possible du type **physobj** en un sous-type d'événement grâce au rôles TELIC et AGENTIVE de la structure de qualiae de *book* (qui fournissent deux alias pour *book*). On a donc ici coercion du type `[individual physobj]` vers le type `[event]`. De cette façon, l'ambiguïté inhérente au mot *book* ne pose pas de problème d'interprétation.

[1] Exemple cité par Pustejovsky et Boguraev [95].

[2] Donné par Pustejovsky et Boguraev [95].

3.6.1.3 Niveaux d'abstraction

Héritage typé

L'ambiguïté qui a été levée ci-dessus est en fait une ambiguïté syntaxique. Pustejovsky et Boguraev [95] proposent en outre de traiter ce qu'ils appellent des "ambiguïtés cachées" : dans le cas d'une hiérarchie de types à héritage multiple, il y a fréquemment ambiguïté sur le bon surtype d'une instance, en fonction de facteurs contextuels et de structure lexicale. Par exemple, en posant dans une hiérarchie de types que *play* et *dictionary* sont deux sous-types de *book*, on s'expose à des contradictions d'usage difficilement explicables logiquement : ainsi on dira *to read a play* mais pas *to read a dictionary*, et inversement, *to consult a dictionary* est acceptable alors que *to consult a play* ne l'est pas.

La solution adoptée dans le GENERATIVE LEXICON est de *typer* l'héritage de types en fonction des structures de qualiae. Il y aura donc quatre treillis orthogonaux dans la même hiérarchie de types :

— book `is_formal` physobj

— book `is_telic` literature

— book `is_agent` literature

— dictionary `is_formal` book

— dictionary `is_telic` reference

— dictionary `is_agent` compiled_material

— play `is_agent` literature

— play `is_telic` book

Cet enrichissement de la notion d'héritage renforce l'aspect polymorphique du langage de dénotation sémantique du GENERATIVE LEXICON et permet d'éviter nombre de problèmes usuellement associés avec l'héritage multiple.

Méta-entrées

Ces auteurs pensent également que leur théorie permet de factoriser les sens d'un mot en une seule méta-entrée qu'ils nomment *lexical conceptual paradigm* (ou LCP). Un LCP permet d'engendrer un jeu de patrons sémantiques applicables pour différentes unités lexicales. Par exemple, les noms *joint venture* et *merger* entrent dans un LCP exprimant le concept d'**union** :

$$\begin{bmatrix} \textbf{union(x)} \\ \text{CONST} = \textbf{\{entity(y), entity(z)\}} \\ \text{FORMAL} = \textbf{entity(x)} \\ \text{AGENTIVE} = \textbf{artifact(x)} \end{bmatrix}$$

Une union est donc un événement qui produit une entité à partir de deux autres et qui est un sous-type d'artefact.

Un LCP est ensuite doté d'un jeu de schémas lexicaux d'actualisation. Pour le LCP **union**, on aura par exemple les schémas (ou N est une instance de **union**, et X et Y sont des arguments) :

— N of X and Y

— X's N with Y

— Y's N with X

— N between X and Y

— N of Z ($Z = X + Y$)

— N between Z

La notion de LCP permet de capter des schémas systématiques de polysémie. Par exemple, les noms *window*, *door* et *room* sont conceptuellement des relations entre un objet physique et une ouverture ou une clôture, ce qui pourra être encodé par un LCP particulier. Au contraire, les noms *lid*, *cap* et *cover* ne réfèrent qu'à l'objet physique et pas à l'ouverture. Ils ne dérivent donc pas du même LCP.

3.6.1.4 Limitations

Le mécanisme de *coercion de type* utilisé dans le GENERATIVE LEXICON a pour but de relâcher les contraintes de restriction de la sélection des constituants sémantiques d'une phrase. Il permet d'utiliser un mot de façon non standard en changeant sa catégorie conceptuelle lorsque celle-ci n'est pas compatible avec le contexte.

Cependant, ce modèle de représentation lexicale n'est pas adapté au traitement des formes de polysémie complexes que sont la polysémie d'acception et la polysémie d'usage. Ses auteurs le reconnaissent eux-mêmes (cf. Pustejovsky et Boguraev [95]). Notamment, la multiplicité des interprétations simultanément valides, qui est omniprésente dans les phénomènes de polysémie d'usage, ne peut être gérée par ce modèle en l'état (tel qu'il est décrit par Pustejovsky et Boguraev dans (*op. cit.*)).

Illustrons ce problème sur l'exemple suivant, tiré de notre corpus de test :

Les pompiers font la fine bouche.

Cette phrase est un exemple typique de jeu de langage (une des quatre catégories fondamentales de phénomènes de polysémie d'usage). Elle se caractérise par une superposition entre un usage attesté (*faire la fine **bouche*** pris au sens de *être difficile*) et une isotopie dans le taxème de la //lutte contre le feu// (qui a pour conséquence un effet d'amorçage de *pompiers* vers *bouche [d'incendie]*).

Prenons la structure de qualiae pour la sorte **bouche** la plus proche possible de la structure que nous utilisons pour l'évaluation de notre propre modèle (cf. p. 199) :

$$\left[\begin{array}{l} \textbf{bouche(x)} \\ \text{CONST} = \textbf{\{lèvres, dents, langue\}} \\ \text{FORMAL} = \textbf{organe(x)} \\ \text{TELIC} = Q_T(x) \\ \text{AGENTIVE} = Q_A(x) \end{array}\right]$$

Si nous nous en tenons aux exemples figurant dans l'article [95], toutes les formules figurants dans les différents rôles d'une telle structure sont des conjonctions. Cela ne permet de représenter des alternatives d'interprétation qu'entre chaque qualia. De plus, la hiérarchie des types conceptuels étant composée de quatre hiérarchies superposées, on se limite à au plus quatre alternatives de sous-typage.

De ce fait, il devient difficile de représenter tous les emplois de *bouche* qui sont représentés dans notre propre structure (p. 199). On peut envisager d'identifier le rôle agentif à un *orifice* :

$$Q_A(x) = \textbf{orifice}(x)$$

mais comment représenter le fait qu'une bouche (en général) peut servir à ingérer ou expulser une substance, à goûter une saveur ou à exprimer un message ? Une formule conjonctive n'est pas acceptable puisqu'il suffit qu'une de ces finalités ne soit pas réalisée pour que l'ensemble du rôle téléologique soit invalidé.

Nous allons donc supposer la possibilité d'utiliser des disjonctions et écrire ainsi les rôles TELIC et AGENTIVE par comparaison avec notre propre représentation :

$$\begin{aligned} Q_T(x) = \textbf{contenir}(E, y, x, z) \quad & \vee \quad \textbf{ingérer}(P, y, x, z) \\ & \vee \quad \textbf{dire}(P, y, x, z) \\ & \vee \quad \textbf{expulser}(P, y, x, z) \\ & \vee \quad \textbf{accéder}(P, y, x, z) \end{aligned}$$

$$\begin{aligned} Q_A(x) = \textbf{cavité-buccale}(x) \quad & \vee \quad \textbf{orifice}(T, y, x, z) \\ & \vee \quad \textbf{fin}(T, y, x, z) \\ & \vee \quad \textbf{appétit}(E, y) \\ & \vee \quad \textbf{convive}(x) \\ & \vee \quad \textbf{attitude}(E, y, x) \\ & \vee \quad \textbf{message}(T, P, y, x, z) \end{aligned}$$

Cette formalisation n'est pas explicitée dans l'article auquel nous nous référons mais n'est pas pour autant proscrite. Il s'agit en fait de représenter un ensemble d'alias pour le prédicat nominal **bouche**(x).

De la même façon, et en nous référant, ainsi que nous l'avons fait pour notre propre lexique, au dictionnaire Petit Larousse 1989, on peut définir l'ensemble d'alias suivant pour la structure prédicative associée à **faire**(x) :

$$\begin{aligned} \lambda x.\big(\textbf{fabriquer}(x) \quad & \vee \quad \textbf{préparer}(x) \\ & \vee \quad \textbf{vendre}(x) \\ & \vee \quad \textbf{accomplir}(x) \\ & \vee \quad \textbf{adopter_attitude}(x) \\ & \vee \quad \textbf{devenir}(x) \\ & \vee \quad \textbf{sembler}(x) \\ & \vee \quad \textbf{produire_effet}(x) \\ & \vee \quad \textbf{état_du_ciel}(x)\big) \end{aligned}$$

(Nous avons ici simplifié les structures argumentatives afin d'alléger l'écriture de cette expression.)

Chacun de ces alias implique un certain nombre de conditions sur l'argument x :

$$\textbf{fabriquer}(x) \Rightarrow \textbf{artefact}(x) \in \Sigma'_x$$
$$\textbf{préparer}(x) \Rightarrow \textbf{artefact}(x) \in \Sigma'_x$$
$$\textbf{vendre}(x) \Rightarrow \textbf{numéraire}(x) \in \Sigma'_x$$
$$\textbf{accomplir}(x) \Rightarrow \textbf{action}(P, y, x, z) \in \Sigma_x$$
$$\textbf{adopter_attitude}(x) \Rightarrow \textbf{attitude}(E, y, x) \in \Sigma_x$$
$$\textbf{devenir}(x) \Rightarrow \exists\sigma \in \Sigma_x / \textbf{qualificatif}(E, y, x) \in \Sigma'_\sigma$$
$$\textbf{sembler}(x) \Rightarrow \exists\sigma \in \Sigma_x / \textbf{qualificatif}(E, y, x) \in \Sigma'_\sigma$$
$$\textbf{produire_effet}(x) \Rightarrow \exists\sigma \in \Sigma_x / \textbf{adverbe}(E, y, x) \in \Sigma'_\sigma$$
$$\textbf{état_du_ciel}(x) \Rightarrow \exists\sigma \in \Sigma_x / \textbf{qualificatif}(E, y, x) \in \Sigma'_\sigma$$

De là, la phrase

Les pompiers font la fine bouche.

sera modélisée par une λ-expression (simplifiée ici en considérant *les pompiers* comme une constante, suite à des calculs antérieurs) du style :

$$E(x) = \lambda x.\big(\textbf{faire}(\text{pompiers}, x) \wedge \textbf{bouche}(x) \wedge \textbf{fine}(x) \wedge \textbf{déterminé}(x)\big)$$

Or, étant données les contraintes sur les différents alias de $\textbf{faire}(\ldots)$ que nous avons énumérées, il vient les modèles d'unification suivants, en supposant que la hiérarchie de types sur laquelle repose le système est construite de manière traditionnelle, c'est-à-dire en appliquant des règles de bon sens sur un fonds de connaissances encyclopédiques (la notation $\Phi[\alpha \rightarrow \beta]$ représente l'expression résultant de la substitution de α par β dans l'arbre syntaxique de Φ) :

— $E(x)[\textbf{faire} \rightarrow \textbf{fabriquer}] \models \emptyset$ car $\textbf{bouche}(x)$ ne peut dériver de $\textbf{artefact}(x)$;

— $E(x)[\textbf{faire} \rightarrow \textbf{préparer}] \models \emptyset$ pour la même raison ;

— $E(x)[\textbf{faire} \rightarrow \textbf{vendre}] \models \emptyset$ car $\textbf{bouche}(x)$ ne peut dériver de $\textbf{numéraire}(x)$;

— $E(x)[\textbf{faire} \rightarrow \textbf{devenir}] \models \emptyset$ car la détermination de *bouche* empêche le syntagme nominal *la fine bouche* de jouer un rôle qualificatif ;

— $E(x)[\textbf{faire} \rightarrow \textbf{sembler}] \models \emptyset$ pour la même raison ;

— $E(x)\textbf{faire} \rightarrow \textbf{état_du_ciel}] \models \emptyset$ pour la même raison encore une fois ;

— $E(x)[\textbf{faire} \rightarrow \textbf{produire_effet}] \models \emptyset$ car la détermination de *bouche* empêche le syntagme nominal *la fine bouche* de jouer un rôle d'adverbe.

Les deux autres interprétations (*accomplir* et *adopter l'attitude*) restent cependant envisageables, et ramènent l'expression $E(x)$ respectivement à :

$$E_1(x) = \lambda x.\Big(\textbf{accomplir}(P, \text{pompiers}, P') \wedge \lambda z.\big(\textbf{ingérer}(P', \text{pompiers}, x, z) \vee \textbf{dire}(P', \text{pompiers}, x, z) \vee \textbf{expulser}(P', \text{pompiers}, x, z) \vee \textbf{accéder}(P', \text{pompiers}, x, z)\big) \wedge \textbf{fine}(x)\Big)$$

et

$$E_2(x) = \lambda x.\Big(\textbf{adopter_attitude}(P, \text{pompiers}, E) \wedge \textbf{attitude}(E, \text{pompiers}, x) \wedge \textbf{fine}(x)\Big)$$

Avec ce modèle simplifié, il existe donc cinq alternatives d'interprétation pour la phrase qui nous intéresse. Cependant, la définition de l'*application coercitive* donnée par Pustejovsky et Boguraev dans [95] induit les règles suivantes pour les cas d'incompatibilité de types :

— Si α est de type $< b, a >$ et β est de type c, alors s'il existe un alias $\sigma(\beta) \in \Sigma_\beta$ qui est de type a on a $\alpha(\sigma(\beta))$ de type a.

— Si α est de type $< [bc], [a] >$ et β est de type $[bd]$, alors s'il existe un alias $\sigma'(\beta) \in \Sigma'_\beta$ qui est de type c on a $\alpha(\sigma(\beta))$ de type a.

Cela signifie qu'en cas d'incompatibilité sémantique (dans notre exemple *faire* et *bouche* sont incompatibles car que veut dire *faire un organe* littéralement ?), il suffit de trouver *un* alias qui rétablisse la compatibilité pour que l'expression soit interprétée en choisissant cet alias.

Dans notre cas, cinq alias satisfont les règles de coercion, or rien ne nous permet d'en choisir un plutôt qu'un autre. Il y a donc *réintroduction* d'un choix arbitraire entre les différentes solutions qui permettent de "résoudre" l'ambiguïté.

Au mieux, on peut supposer qu'il existe dans le système un moyen de favoriser l'interprétation $E_2(x)$ en vertu d'un principe de *préférence* entre *bouche* et *fine* pour l'alias **attitude**, ce qui correspond bien à l'interprétation la plus plausible de notre phrase. Cependant, quel que soit l'alias choisi, il n'y en a pas moins perte d'information.

On pourra comparer cette situation au résultat fourni par notre propre modèle (cf. p 245) qui nous donne comme interprétation de *bouche*, et dans l'ordre croissant de préférence (cf. l'estimation des interprétations, §4.3.4) :

1. cavité buccale (estimation neutre) ;
2. bouche d'incendie (estimation de $1/2$) avec pour traits caractéristiques saillants : pompiers et eau ;
3. personne difficile (estimation de 1).

Les interprétations de *faire ... bouche* comme action n'ont pas été retenues car elles ne correspondent pas à des usages attestés (i.e. réellement rencontrés) du mot *bouche*.

Ce résultat contient donc l'interprétation principale de la phrase (*faire la fine bouche*, c'est se comporter comme une personne difficile) en plus de l'interprétation induite par l'isotopie que nous avions signalée (la cooccurrence de *pompiers* et *bouche* évoque naturellement l'idée d'une bouche d'incendie), tout en n'occultant pas le fait que le mot *bouche* est très majoritairement utilisé pour désigner la cavité buccale et que cette information pourra éventuellement être utile par la suite.

Comme on le voit, la relaxation de contraintes offerte par le GENERATIVE LEXICON, si elle conduit le système à interpréter des phrases qui le conduiraient normalement en échec, ne permet pas de rendre compte de la *richesse informative* de la polysémie d'usage. Ce modèle ne *traite* pas la polysémie d'usage ; il la *résoud*.

3.6.1.5 Récapitulation

La théorie sémantique sur laquelle s'appuie le GENERATIVE LEXICON démontre l'importance d'un certain nombre de caractéristiques d'un lexique dynamique :

— L'usage des mots peut s'encoder facilement à un niveau sublexical, ce qui nous amène naturellement à adopter une structure onomasiologique.

— Prendre en compte l'influence du contexte (ou des schémas d'actualisation lexicaux) autour d'un mot entraîne nécessairement de recourir à des opérations de *relaxation de contraintes sémantiques* (ici, la coercion de type).

Cependant, cette théorie nous pose deux problèmes majeurs dans le cadre de notre travail :

1. Tous les exemples d'ambiguïté dans les articles portant sur le GENERATIVE LEXICON sont des instances de polysémie fonctionnelle (en majeure partie) et de polysémie d'acception (plus rarement). Malgré la revendication théorique attribuant au GENERATIVE LEXICON la faculté de traiter des usages imprévus par génération au coup par coup de nouvelles représentations sémantiques, la polysémie d'usage n'est jamais abordée dans les exemples illustrant cette théorie, ne serait-ce qu'implicitement, et rien ne prouve qu'il est réellement possible de la traiter à l'intérieur de ce formalisme.

2. Cette théorie repose sur un typage fort, ce qui est une façon de réintroduire l'utilisation des concepts au détriment des signifiés. L'établissement d'une hiérarchie de types est toujours arbitraire, un compromis entre la connaissance scientifique et le sens commun. Que les sorties de ce lexique soient compatibles avec une telle hiérarchie nous semble souhaitable, mais que le système en ait besoin pour ses traitements internes nous paraît épistémologiquement suspect. Toutes les observations linguistiques que nous avons faites (cf. chapitre 2) nous ont laissé entendre que la langue, dans son utilisation, tend à échapper aux logiques de sens commun aussi bien que formelles, et surtout n'est pas fixe dans le temps. Comment, dans ce cas, se référer à des catégories abstraites et absolues sans mettre une fois de plus en danger la souplesse de description que nous avait suggéré la sémantique différentielle ?

3.6.2 Le modèle EDGAR

Le modèle EDGAR (*Entry Driven Graph for Ambiguity Representation*) a été développé par Prince [86, 88, 89, 92] pour permettre de mettre en œuvre un lexique électronique qui *ne se débarrasse pas de l'ambiguité lexicale*, mais au contraire la conserve dans ses représentations.

Le paradigme de Winograd

Ce modèle se caractérise notamment par un rejet de ce que l'auteur appelle le "paradigme de Winograd pour la sémantique lexicale" et qui est basé sur les deux postulats suivants (que nous avons déjà mis en évidence dans les chapitres précédents) :

1. Dans une phrase donnée, chaque mot n'a qu'un seul sens ; toutes les autres alternatives sont des parasites qui doivent être ignorés.

2. Découlant du point précédent, si des alternatives sont des parasites, c'est qu'elles n'ont pas de significations dans le contexte considéré. Par conséquent, en pratique il n'y a aucune relation entre les différentes significations attachées à un mot, autrement chaque mot pourrait avoir plus d'une lecture valide dans un contexte donné.

Prince observe que les systèmes et théories les plus répandus respectent le paradigme de Winograd. Ainsi, en linguistique systémique (cf. Halliday [48, 47]) la diversité des significations est prise en compte par les liens typés existant entre les différents termes / entités, et qui peuvent être utilisés pour classifier le sens (cf. Evens *et al.* [31]). Ce genre de taxonomie permet d'améliorer des modèles de la signification existants, tels que la théorie du "sens-texte" de Melçuk [74], les représentations sémantiques de Woods [138], et les modèles sémantiques inspirés de KL-One (cf. Brachman [10]). L'auteur observe également que, bien que certains systèmes soient basés sur une utilisation conjointe des informations syntaxiques et sémantiques (cf. Mann [69]), d'autres sur des stratégies opportunistes par gestion de l'attente d'informations complémentaires (cf. Rich [102]) et d'autres encore sur l'utilisation du "sens commun" en tant que règle d'usage (cf. Dahlgreen [25]), ce genre de classification de la signification n'est pas satisfaisant en soi, puisque tous ces systèmes cherchent à exhiber "la bonne et unique solution", ce qui implique un recours à un processus de choix arbitraire à un moment ou à un autre.

Concernant les systèmes traitant le français plus particulièrement, que ce soit des systèmes basés sur l'apprentissage des mots avec les fonctions syntaxiques qu'ils remplissent en présence d'autres mots (cf. Debili [27]), ou par recours à la théorie de la profondeur sémantique variable (cf. Kayser [57]), Prince fait observer que l'ambiguïté est toujours conçue comme résultant d'une *imprécision* inhérente à nos *instruments de mesure sémantique*.

3.6.2.1 Originalités d'EDGAR

EDGAR se démarque tout d'abord par le fait qu'il a été conçu pour s'insérer dans une architecture de traitement automatique des langues naturelles plus vaste : CARAMEL-1 (cf. Sabah *et al.* [107, 108]), qui est une architecture multi-agents d'experts dédiés chacun à une sous-tâche particulière dans la chaîne de traitement des textes. Le but original qui a donné naissance à EDGAR était de produire un expert pour CARAMEL-1 qui permette de discriminer les signifiés lexicaux de façon assez précise, et qui, pour cela, respecte les contraintes suivantes :

1. Prendre en compte les trois contraintes lexicales mises en évidence par Fass [32] dans son système MET* :
   - la *préférence* (basée sur la théorie de Wilks [136]) qui permet de discriminer la classe sémantique d'unités lexicales cooccurrentes,
   - l'*assertion* qui impose de l'information aux élements possédant des contraintes lexicales syntaxiques semblables,
   - et la *pertinence* en contexte.
2. Etre aussi compatible que possible avec le formalisme des graphes conceptuels (cf. Sowa [116]) utilisé dans CARAMEL-1.
3. Fournir un comportement d'expert coopératif.
4. Avoir le format de description le plus compact possible.

5. Ne pas tenter de résoudre le problème de discrimination du sens de façon dichotomique.

Le modèle EDGAR se situe dans la catégorie des lexiques dynamiques puisqu'il en possède les deux caractéristiques principales : une description statique minimale et des mécanismes dynamiques d'inférence largement développés. Ces inférences sont de trois types :

1. Des règles pragmatiques qui, au contraire de tous les autres systèmes que nous avons pu observer, *déclenchent* l'évaluation du sens au lieu de la compléter. Ces règles mettent en œuvre les principes de préférence, d'assertion et de pertinence de Fass et Wilks, et ajoutent également des règles de "sens commun".

2. Des contraintes sémantiques permettant d'envisager la cohérence sémantique des interprétations sur une période de discours (en fait, le contexte proche).

3. Un raisonnement par défaut, comme dans le système de Copestake et Briscoe [23], pour essayer de dériver de l'information à partir de structures partiellement instanciées dans le but de les compléter.

Concepts partiels

Une dernière caractéristique essentielle de ce modèle est que les liens qu'il possède entre les mots (unités lexicales) et les concepts (la base de graphes conceptuels de CARAMEL-1) sont uniquement des liens de *pertinence* (cf. Rieger [103]). Il n'imposent aucun prérequis sur les concepts associés aux unités lexicales, en dehors de leur description locale en terme de traits pertinents. Cela est basé sur une observation de Prince concernant l'intensionnalité des concepts abstraits ou généraux : il est souvent difficile d'arriver un consensus sur leur contenu intensionnel (leur définition exacte) alors que chacun peut les lexicaliser correctement. L'auteur propose alors de considérer plutôt des "définitions intensionnelles relatives" qu'elle appelle *vues conceptuelles*. Ces vues permettent de décrire un concept à l'aide des propriétés qui sont pertinentes pour le mot qui sollicite ce concept. Ainsi, dans l'expression *une glace maison*, même si c'est le concept d'artisanat qui est prééminent, le concept d'entreprise est néanmoins sollicité puisque artisanat et entreprise sont liés. L'auteur estime alors que c'est une vue sur l'entreprise qui est sollicitée par *maison* pour peu que la vue conceptuelle du concept d'entreprise soit décrite par des propriétés comme : profession, savoir-faire, artisanat. Ainsi artisanat ne joue pas entièrement le rôle d'un concept, mais celui d'une propriété permettant une description pertinente d'une entité symbolique s'apparentant à entreprise.

Cette intuition a été validée par une étude de l'auteur sur les sens proposés par un dictionnaire (le Grand Robert), et qui l'a amenée à dire que ce genre de propriétés sont les traits qui suffisent à exprimer la *participation* du concept d'entreprise dans les interprétations attestées du mot *maison*. Le concept d'entreprise n'est pas épuisé (d'où l'intensionnalité "partielle") par sa description intensionnelle sollicitée par *maison*. Il peut être sollicité par d'autres réalisations lexicales, comme *société* ou *compagnie*, avec d'autres propriétés, des recoupements et des différenciations. Le concept d'entreprise est sollicité par tous ces mots, mais partiellement par chacun. L'unicité du concept, en utilisant la définition de concept donnée par Kayser [58] « entité symbolique recevant des propriétés inférentielles », est maintenue par le fait qu'un concept reçoit les propriétés inférentielles de *toutes* ses sollicitations dans ses réalisations lexicales possibles.

Du coup, Prince adopte les hypothèses suivantes pour son modèle :

— Il ne doit pas contenir de représentation des concepts, puisque seules des vues conceptuelles relatives sont sollicitées. Ce sont elles qui sont mises en avant, étant donné une réalisation lexicale.

— La représentation des connaissances lexicales est axée sur les rapports entre un mot et les vues conceptuelles qu'il sollicite.

3.6.2.2 Composante statique

Les connaissances statiques associées à un mot m sont regroupées dans sa *structure descriptive* DS_m. Cette structure est composée de deux sous-ensembles : $DS_m^1 = \{C_1, \ldots, C_n\}$ qui est l'ensemble des vues conceptuelles sollicitées par ce mot, et $DS_m^2 = \{a_1, \ldots, a_t\}$ qui est l'ensemble des traits qui spécialisent les vues conceptuelles. Ainsi, dans l'expression *processus père*, une vue conceptuelle de *père* est *hiérarchie*, qui est spécialisée par les traits *antécédance* et *génération*[1].

Afin d'assurer la corrélation entre une vue conceptuelle et ses traits, l'auteur utilise un prédicat B_m qui indique l'existence (ou non) d'un lien entre une vue conceptuelle et un trait dans le contexte lexical du mot m. $B_m(C_i, a_j) = 1$ si $C_i \in DS_m^1$, $a_j \in DS_m^2$ et a_j est un trait de C_i relativement à m. Sinon, $B_m(C_i, a_j) = 0$.

Le couple $S_m = (DS_m, B_m)$ est appelé *potentiel sémantique* de m. Cela indique que S_m contient l'ensemble des interprétations possible de m.

3.6.2.3 Composante dynamique

Les règles d'inférences utilisées par EDGAR manipulent des *taux d'acivité* des composants du potentiel sémantique. Le taux d'activité d'un élément dénote dans quelle mesure cet élément participe à la construction de la signification en contexte du mot. Il s'agit en fait de taux *symboliques* pouvant exprimer une haute activité (une grande participation), une activité existante mais difficilement remarquable, ou une absence de pertinence en contexte. Il existe également une activité importante dans le cadre d'un contexte négatif pour traiter des cas comme *Ce n'est décidément **pas** une **bonne mère***[2].

Cela conduit à utiliser des valeurs d'activité prises dans l'ensemble

$$V = \{\text{saillant}, \text{valide}, \text{ignoré}, \text{inhibé}\}$$

Les règles d'inférence se divisent en deux grands groupes :

— les règles qui dérivent l'activité à partir du contexte : elles attribuent un taux d'activité à un élément du potentiel sémantique à partir de schéma détectés dans le contexte proche ;

— les règles qui propagent l'activité à travers le potentiel sémantique.

Toutes ces règles se basent sur les principes suivants :

1. Lorsqu'un élément à reçu un taux d'activité, cette activité peut se propager aux autres éléments, par un système de "taxation", de sorte à fonctionner avec le minimum d'information possible.
2. Si un élément est susceptible de recevoir différents taux d'activité, les taux spécifiques (comme saillant et inhibé) sont favorisés par rapport aux taux implicites.
3. Le système de taxation doit montrer quels éléments sont pertinents dans le contexte (saillant ou inhibé), quels éléments sont valides et quels éléments ne sont pas pertinents (ignoré). En cas d'absence d'activité, une connaissance implicite (valide) est un bon candidat à l'interprétation.

[1] Tiré de Prince [88].
[2] Analysé par Prince dans [88].

Règles de dérivation

Ce sont les *règles pragmatiques* qui évaluent l'impact de l'information contextuelle sur l'interprétation. Ces règles font appel au *domaine d'information* I_m du mot m. I_m est en fait une structure d'attributs ouverte. Chaque attribut évalue une certaine propriété du contexte et reçoit une valeur de type arbitraire (mais fonction de l'information contenue dans cette propriété).

L'ensemble des règles pragmatiques associées au mot m est

$$P_m^1 = \{[\kappa_j \rightarrow p(x, v)], \kappa_j \in I_m, x \in DS_m, v \in V\}$$

où les p sont des prédicats définis comme suit :

$$\begin{array}{rcccl} p & : & DS_m \times V & \longrightarrow & \{0, 1\} \\ & & (x, v) & \longrightarrow & 1 \text{ si } v \text{ est assigné à } x \text{ par une règle} \\ & & (x, v) & \longrightarrow & 0 \text{ sinon.} \end{array}$$

Règles de propagation

Il existe deux types de règles de propagation : les *contraintes sémantiques* qui expriment des contraintes de cohérence sémantique spécifiques à chaque mot, et des *règles de propagation par défaut*.

Contraintes sémantiques Les contraintes de cohérence sémantique sont de deux sortes :

— Un trait a *implique* un trait b décrivant la même vue conceptuelle lorsque b reçoit la même activité que a dans tous les contextes.

— un trait a *s'oppose* à un trait b décrivant la même vue conceptuelle lorsque b reçoit l'activité inverse de a dans tous les contextes. Ceci induit un opérateur d'inversion opp sur V qui déclare saillant et inhibé comme inverses l'un de l'autre, et laisse valide et ignoré invariants.

Formellement, soient a_i et a_k deux traits d'une même vue conceptuelle C pour le mot m, i.e. soient $a_i, a_k \in DS_m^2$ et $C \in DS_m^1$ tels que $B_m(C, a_i) \wedge B_m(C, a_k)$.

L'implication est définie par

$$\forall v_i \in V, a_i \rightarrow a_k \Rightarrow [p(a_i, v_i) \rightarrow p(a_k, v_i)]$$

L'opposition est définie par

$$\forall v_i \in V, a_i \not\rightarrow a_k \Rightarrow [p(a_i, v_i) \rightarrow p(a_k, opp(v_i))]$$

L'ensemble des contraintes sémantiques de m est donné par :

$$P_m^2 = \left\{ \left[[p(a_i, v_i) \rightarrow p(a_k, v_i')] \wedge B_m(C, a_i) \wedge B_m(C, a_k)\right], C \in DS_m^1, a_i, a_k \in DS_m^2, v_i, v'i \in V \right\}$$

Règles par défaut Ces règles sont indépendantes des mots contenus dans le lexique. Par convention, on appelle a un trait, c une vue conceptuelle et x un élément indifférencié (trait ou vue). v dénote un taux d'activité et p un prédicat qui attribue une valeur v à un élément x. B est le prédicat qui donne les associations entre une vue conceptuelle et les traits qui la décrivent (sans indice m, puisque les règles par défaut sont valables sur tout le lexique).

1. Si x a reçu une valeur v et une autre règle tente de lui attribuer la valeur valide, alors x garde sa valeur précédente. Tout autre cas est un problème d'incohérence entre deux règles de même type (par exemple deux règles pragmatiques) et fait échouer l'interprétation.
2. Si a est un trait décrivant la vue c et a reçu la valeur saillant ou inhibé, alors a propage la valeur saillant à la vue c, si c n'a pas encore reçu de valeur.
3. Si a est un trait décrivant c et a a reçu la valeur valide, ou si c' est une autre vue de DS^1 qui a reçu une valeur différente de ignoré, et que c n'a pas encore reçu de valeur, alors c reçoit la valeur valide.
4. Si a est un trait décrivant c et si c a reçu une valeur différente de ignoré, alors c propage la valeur valide à a (et ce pour tous les a tels que $B(c, a)$).
5. Si a est un trait décrivant c et c a reçu la valeur ignoré, alors c propage la valeur ignoré a a (pour tous les a tels que $B(c, a)$).

Algorithme

Prince appelle *configuration* de m une interprétation de m en contexte. Il s'agit d'une liste de paires où le premier élément appartient au potentiel sémantique de m et le second est une valeur d'activité.

Soit m le mot à interpréter, κ le contexte d'interprétation (la phrase où m apparaît, son thème, son foyer, etc.) et (DS_m, B_m) le potentiel sémantique de m. Alors la configuration X de m dans le contexte κ est

$$X(m, \kappa) = \big\{(x, v), \forall x \in DS_m, v \in V\big\}$$

L'algorithme est le suivant :

Pour tout $x \in X(m, \kappa)$
  Tant que $\kappa_j \in I_m \cap \kappa$
    Appliquer $P_m^1 = \{[\kappa_j \rightarrow p(x, v)]\}$
  Si $P_m^2 \neq \emptyset$ **alors**
    Appliquer $P_m^2 = \{[[p(a_i, v_i) \rightarrow p(a_k, v'i)] \wedge B(c, a_i) \wedge B(c, a_k)]\}$
  Appliquer les règles de propagation par défaut

3.6.2.4 Récapitulation

Le modèle EDGAR s'avère particulièrement original par rapport aux modèles habituels de représentation de la signification lexicale. Nous pouvons énumérer un certain nombre de caractéristiques particulièrement intéressantes et qui concordent avec les observations linguistiques du chaiptre 2 :

1. En tout premier lieu, la volonté de Prince de recourrir à des *vues conceptuelles* plutôt qu'à des concepts, en se basant sur la notion de *pertinence*, lui permet de s'affranchir de la notion de concept dans le fonctionnement de son modèle, tout en restant compatible avec elle au niveau des sorties. Ce qui signifie surtout que de façon interne, le modèle EDGAR manipule ce qu'on peut considérer comme une bonne approximation des *signifiés* de la sémantique différentielle. En effet, la lexicalisation de la vue conceptuelle C par le mot c relativement au mot m n'implique par que la même lexicalisation c sollicite la même vue conceptuelle relativement à un autre mot m'.

2. Le fait d'utiliser des *lexicalisations* pour décrire le potentiel sémantique d'un mot permet d'évacuer le problème de l'intensionnalité : ce ne sont pas des relations sémantiques qui sont directement codées dans une structure descriptive, mais plutôt des *régularité d'usage* des mots qui servent à les exprimer dans la langue vécue. Ceci débarrasse le modèle EDGAR de tout postulat intensionnel concernant la réalité des essences conceptuelles, et leur adéquation à l'expression (par le truchement de la vérité notamment).

3. Il s'agit d'un modèle hybride entre la représentation symbolique et les réseaux connexionistes, ce qui permet de jouer sur la dualité symbole / faisceau de traits des entités et d'atteindre un nivau de description sublexical sans pour autant définir de constituants sémantiques en-dessous du niveau du mot (donc d'éviter la tentation de la compositionnalité).

4. Il n'y a aucune différence de statut entre interprétations "littérales" et interprétations "métaphoriques" : toutes sont considérées comme interprétations attestées et traitées de la même façon (sans diktat épistémologique, il n'y a en effet plus moyen de différencier une signification prétendue normale ou orthodoxe d'une signification "déviante"). Les sens figurés sont alors aussi naturels que les autres dans ce modèle.

5. Les taux d'activité symboliques permettent de représenter assez fidèlement les "importances" des diverses significations en présence dans l'interprétation d'un mot.

6. Et finalement, mais fait d'une importance capitale, ce modèle ne produit pas de solution atomique. La solution type d'EDGAR est un *ensemble* de significations *coparticipantes* dans une interprétation en contexte.

Nous retiendrons cependant un certain nombre de limitations pour le modèle EDGAR :

— La notion ethnométhodologique d'*indexation* des significations par le contexte local qui, dans le cas d'EDGAR est entièrement circonscrite par le domaine d'information, n'est, à notre avis, pas suffisamment prise en compte (en tout cas si l'on veut adapter ce genre de modèle à la polysémie d'usage).

— De façon corollaire, cela tend à amoindrir la notion de profondeur sémantique qui n'est présente que dans la différenciation vue conceptuelle / trait. Nous pensons qu'il manque dans ce modèle des niveaux de généralisation supérieurs à celui de la vue conceptuelle.

— Enfin, les sorties d'EDGAR ne fournissent aucune indication à un éventuel expert en aval travaillant sur une base conceptuelle sur les choix qu'il doit faire entre les différentes significations coparticipantes. EDGAR fournit bien entendu des sorties idéales par rapport à la sémantique différentielle (en termes d'absence de choix, justement), mais il faut également pouvoir faire des compromis avec les systèmes de traitement automatique des langues naturelles plus larges existants. Ceux-ci

travaillent, comme nous l'avons vu, sur des représentations atomiques et conceptuelles. Il semblerait alors judicieux que les interprétations d'un modèle du genre EDGAR contiennent des indications de confiance ou d'importance permettant au reste du système de guider ses choix d'interprétation conceptuelle, quitte à procéder par retour-arrière sur les valeurs décroissantes de ces indications.

En conclusion, le modèle du traitement de la polysémie d'usage que nous allons présenter au chapitre 4 s'inspirera très largement d'EDGAR en tant que *paradigme*. Nous nous sommes attachés, lors de la constitution de notre modèle, à conserver les caractéristiques cruciales d'EDGAR qui nous avons citées plus haut, tout en corrigeant les lacunes que nous venons d'évoquer.

3.7 Récapitulation

Nous avons vu dans ce chapitre un large éventail de modèles computationnels destinés à (ou capables de) traiter les ambiguïtés lexicales. On pourra tirer de ces observations les conclusions suivantes :

— Les modèles classiques (tels que le WEP, les PWet l'architecture CARAMEL) qui se fondent sur les théories linguistiques de la référence et de l'inférence ne sont pas adaptés au traitement des formes de polysémie complexes, en particulier de la polysémie d'usage. Ils reposent sur une conception du lexique utilitaire et centrée sur la syntaxe, ce qui les place d'emblée dans des cas de polysémie assimilables à l'homonymie comme la polysémie fonctionnelle. Cela les conduit à adopter des stratégies de résolution des ambiguïtés lexicales de type *winner takes all* : une seule interprétation est sélectionnée, au prix d'une perte d'information, voire de non pertinence.

— Le modèle de Brugère (cf. §3.4) considère la polysémie d'acception de façon autonome. Ses sorties sont ensemblistes plutôt par défaut (lorsqu'il n'est pas capable de déterminer la bonne acception) qu'à dessein.

— Le modèle morphodynamique de Victorri *et al.* et sa mise en œuvre connexioniste (cf. §3.5) vise le traitement de toutes les formes de polysémie. Cependant, son utilisation est rendue lourde et opaque en raison des choix de formalisation et d'implantation sur machine.

— Le GENERATIVE LEXICON de Pustejovsky (cf. §3.6.1) offre une alternative efficace aux lexiques computationnels utilisés habituellement. Cependant, son recours à une hiérarchie de types conceptuels le replace pour partie dans le cadre de la sémantique référentielle. D'autre part, sa stratégie de relaxation de contraintes par coercion de type n'admet pas la gestion ensembliste des interprétations.

— Finalement, le modèle EDGAR de Prince (cf. §3.6.2) est particulièrement bien adapté au traitement de la polysémie (tout comme le modèle de Victorri *et al.*), tout en offrant une mise en œuvre beaucoup plus légère et transparente. De plus, il intègre parfaitement la notion de signifié linguistique en ne calculant que des paraphrases (du type de celles utilisées en sémantique différentielles) intralinguistiques des interprétations d'une expression polysémique. Il s'agit selon nous d'un bon paradigme théorique pour la constitution d'un lexique gérant l'ambiguïté de manière naturelle, même s'il présente certaines limitations qui l'empêchent de traiter en l'état tous les cas de polysémie d'usage.

Chapitre 4

Le modèle PELEAS

> « Les ordinateurs ne sont que des systèmes dotés d'une grande inconscience : tout y est tenu en mémoire de façon immédiatement accessible, et sujet à des programmes lancés par l'opérateur. L'opérateur, donc, est la conscience de l'ordinateur. »
>
> **Frank Herbert**, *Destination Vide*

4.1 Introduction

Ce chapitre présente le modèle que nous avons élaboré pour le traitement automatique des formes de polysémie complexes : polysémie d'usage *et* polysémie d'acception.

Un tel modèle doit respecter deux types de contraintes : il doit tout d'abord prendre en compte les observations linguistiques du chapitre 2 et s'y conformer. De plus, ayant pour vocation d'être mis en œuvre sur machine, il se doit de respecter des critères d'efficacité (rapidité, mémoire occupée réduite) et de compatibilité avec d'autres systèmes plus larges de traitement automatique des langues naturelles. Pour cela, il doit reprendre certaines techniques et formalisations issues de systèmes existants, tout en évitant leurs écueils (cf. chapitre 3).

Dans un premier temps, nous présenterons une expérience (cf. §4.2) que nous avons menée afin de compléter les données théoriques issues de la sémantique différentielle (cf. §2.5.2).

Le §4.3 décrit le modèle PELEAS de manière formelle, puis nous abordons au §4.4 la question des critères d'automatisation de la construction du lexique propre à PELEAS. Et finalement, nous procédons à une évaluation préliminaire de ce modèle sur un corpus de test (cf. §4.6).

4.2 Experience préliminaire

Aux chapitres précédents, nous avons passé en revue un large éventail de modèles linguistiques et computationnels concernant la sémantique et l'interprétation de la polysémie. Cela nous a permis, par rapport à notre objectif qui est de produire un modèle computationnel de la polysémie d'usage, de récuser certains cadres théoriques, et de cerner certaines directions de recherche. En résumé, nous tirons les trois conclusions suivantes :

1. La linguistique computationnelle chomskyenne traditionnellement utilisée en traitement automatique des langues naturelles (s'appuyant sur les théories sémantiques de la référence et de l'inférence) est inadéquate pour produire un modèle de la polysémie d'usage, notamment en raison de :
   - son attachement à l'utilisation des concepts au détriment des signifiés,
   - son incapacité à envisager des interprétations multiples simultanément (le *paradigme de Winograd)*.
2. Le cadre théorique le plus souple et le plus puissant, en ce qui nous concerne, est celui de la sémantique différentielle.
3. Un modèle, linguistique ou computationnel, de la polysémie en général, et de la polysémie d'usage en particulier, doit pouvoir prendre en compte les phénomènes suivants :
   - la connotation (cf. §2.7.1),
   - la profondeur variable (en termes de niveaux de généralité) des interprétations,
   - la continuité sémantique entre les différentes interprétations valides.

Le problème inhérent à l'utilisation de la sémantique différentielle dans un cadre informatique est la difficulté qu'il y a à la mettre en œuvre sur machine. La sémantique différentielle fournit beaucoup de considérations théoriques intéressantes, mais peu de critères pratiques pour l'appliquer de manière automatique. Ainsi, le système de représentation sémantique différentielle développé par Beust [7] se réfère à un cadre extérieur afin de normaliser la notion de pertinence (qui est la plus difficile à opérabiliser). Ce cadre est un modèle de dialogue homme-machine (cf. Nicolle [78]) basé sur une logique interne d'apprentissage et des gestion des incomplétudes par les dialogues incidents (cf. Lehuen [66]).

De même, nous avons cherché un cadre extérieur auquel nous référer pour normaliser le problème de la pertinence sémique. Nous présentons ici une expérience que nous avons menée dans ce but : chercher des critères pratiques grâce à l'observation de faits. Dans notre cas il s'agit d'observer l'activité de traduction et de résolution des ambiguïtés lexicales qui surviennent au cours de cette activité.

4.2.1 L'interprétation comme traduction

La sémantique différentielle s'inscrit dans un paradigme *interprétatif* : le but visé est de modéliser non la compréhension d'un texte, mais son interprétation. L'interprétation est un processus abstrait, donc non observable, de transformation des représentations des énoncés. Afin de se ramener à un phénomène observable, Rastier [99, 101] appelle interprétation la production d'une explicitation (sous forme de paraphrases linguistiques, une analyse sémique pouvant faire office de paraphrase). Cela se rapproche d'une démarche de traduction (qui est un autre processus basé sur l'interprétation abstraite), comme illustré sur la figure 4.1.

Dans le cas de l'interprétation-explicitation, comme dans le cas de la traduction, il s'agit de passer d'un texte source à un texte cible : les données, comme les résultats, sont de nature linguistique. La seule différence tient dans le fait que pour l'interprétation, les textes source et cible sont exprimés dans une même langue, alors que dans le cas de la traduction, il s'agit de deux langues différentes. Bien sûr, lorsqu'on traduit un texte, on ne cherche généralement pas à *expliciter* le texte source, mais on peut toujours considérer que le texte cible constitue une paraphrase du texte source, au niveau interlinguistique.

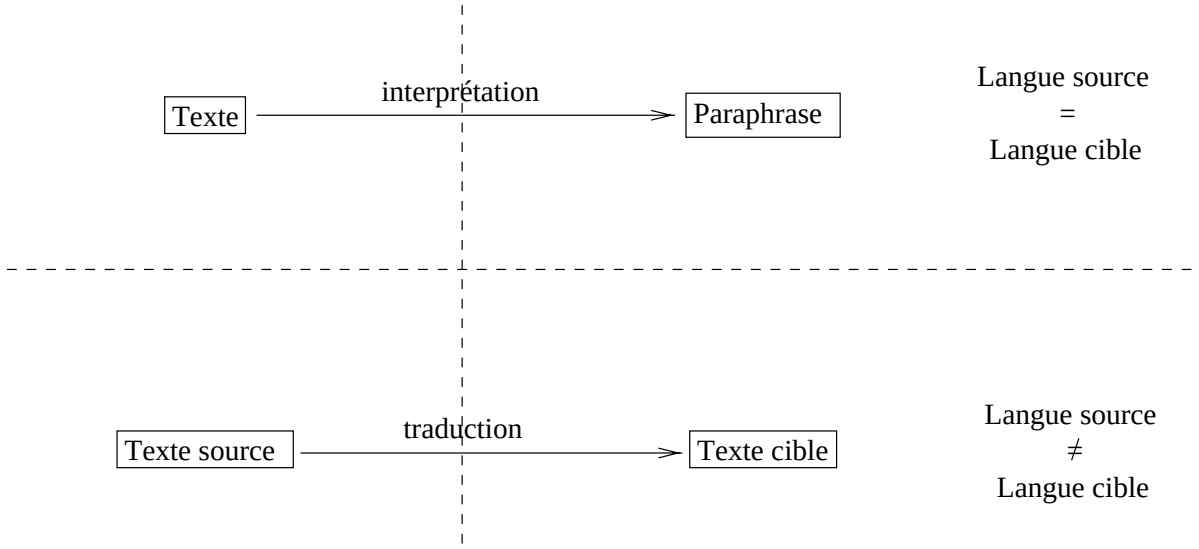


FIG. 4.1: Comparaison entre interprétation et traduction

C'est pourquoi nous nous sommes tournés vers l'observation des processus de traduction afin d'y rechercher des critères opératoires qui pourraient nous servir dans la constitution de notre modèle. Nous nous appuyons pour cela sur une confusion délibérée entre explicitation et traduction, qui n'a d'autre justification qu'une similitude formelle apparente entre processus, et sur une approche intuitive.

4.2.2 Observations

Nous supposons donc que l'interprétation d'une occurrence de polysémie d'usage s'apparente à un processus de traduction. Nous cherchons ici à construire un modèle fonctionnel de l'activité de traduction de textes ambigus comportant des occurrences de polysémie d'usage (et éventuellement d'acception).

L'hypothèse pratique qui motive cette observation est que la traduction se constitue en *métier*, comportant donc une composante didactique d'apprentissage, et une composante d'expertise. Nous pouvons procéder à une modélisation de cette tâche par :

— une observation extérieure, en partant du principe que toute activité qui s'apprend peut être enseignée, et donc que les traducteurs sont capables, dans une certaine mesure, d'expliciter leurs propres critères ;

— une observation intérieure, en ce qui concerne la composante d'expertise, afin de procéder à une extraction des connaissances que les traducteurs ne peuvent expliciter.

4.2.2.1 Recueil des données

Le recueil des données s'est étalé sur une période d'un an, structurée comme suit :

1. une phase d'observation participante de six mois au sein de la société Linguatech[1] à Boulogne-Billancourt ; l'observateur tenant le rôle de *réviseur technique* qui comporte une part importante de

[1] A ce jour Bowne Global Solutions.

traduction d'expressions techniques ambiguës, polysémiques ou idiomatiques (cette phase permet de réaliser l'observation intérieure de l'aspect expertise de la traduction) ;

2. une phase d'entretiens, durant les six mois suivants avec neuf traducteurs de la même société, afin de leur demander d'expliciter leur démarche, et de fournir un retour sur la modélisation de leur activité :

   — neuf séances d'une heure d'entretien, une séance pour chaque traducteur,

   — une séance d'une heure et demie avec les neuf traducteurs collectivement, afin de faire une synthèse et d'évaluer la proposition de modèle.

Les activités de traduction de l'observateur et des traducteurs observés portaient sur des textes en anglais à contenu fortement technique, traduits en français. Ces textes constituaient l'ensemble de la documentation (électronique et papier) du jeu de logiciels Office 95 de Microsoft.

4.2.2.2 Description synthétique

La démarche adoptée par les traducteurs face à une expression ambiguë polysémique s'apparente à un processus incrémental contenant les étapes présentées à la figure 4.2.

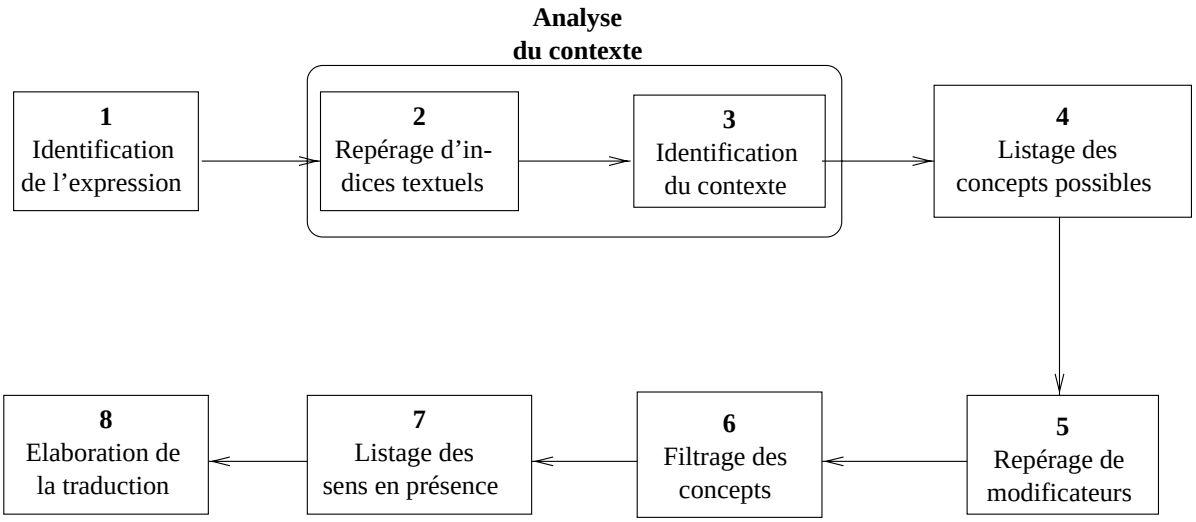


FIG. 4.2: Processus de traduction

Nous cherchons à modéliser les étapes 2 à 7. Nous considérons que l'étape 1 est du ressort d'un agent extérieur (humain dans le cadre d'un outil d'aide au traitement de la polysémie, artificiel dans le cadre d'une chaîne logicielle de traitement automatique des langues naturelles). L'étape 8 n'est pertinente que dans un contexte de traduction, et pas dans celui du traitement de la polysémie en général. Nous nous fixons comme but uniquement d'arriver à trouver les différents sens (signifiés) présents dans la signification en contexte d'une occurrence polysémique.

Nous détaillons maintenant les étapes qui nous intéressent :

2. **Repérage d'indices textuels** : la première étape consiste à rechercher toutes les informations morpho-syntaxiques, typographiques et rhétoriques possibles qui peuvent fournir une amorce d'interprétation. Ces informations sont recherchées dans l'expression ambiguë elle-même, ainsi que

dans le contexte proche, qui se limite en général à la phrase dans laquelle s'insère cette expression. Il s'agit d'une phase d'*élagage* (de suppression d'interprétations potentielles non pertinentes) par référence à des situations connues. Ainsi la présence d'une majuscule à l'initiale de l'expression ambiguë, ou de guillemets, ou bien une position d'épithète lorsqu'il s'agit d'un nom commun peuvent indiquer une "direction" de traduction de façon pertinente.

3. **Identification du contexte** : cette deuxième étape d'analyse du contexte porte sur des aspects pragmatiques et sociolinguistiques. Il s'agit principalement d'identifier deux caractéristiques :

   — la ou les *notion(s)*, que les traducteurs ont aussi qualifiées de "directions générales d'interprétation". Ce sont des signifiés d'un haut niveau de généralité, superordonnés aux signifiés susceptibles d'interpréter l'expression ambiguë. Ainsi, dans l'exemple de la *cellule mère* (cf. p. 32), la direction générale d'interprétation porte sur la notion de *hiérarchie* (plutôt qu'une notion d'*individu* ou d'*institution*). Une notion délimite implicitement un champ sémantique pour les interprétations ultérieures.

   — le ou les *domaine(s)* d'application, qui correspondent aux domaines de la sémantique différentielle (cf. p. 52). Un domaine d'application caractérise l'activité qui est décrite à cet endroit précis du discours. Toujours pour l'exemple de la *cellule mère*, le domaine d'application serait celui de la biologie. Il s'agit d'une information importante car chaque domaine d'application possède ses propres règles d'usage linguistique qui peuvent influencer le style, la syntaxe ainsi que les règles de compatibilité sémantique entre les différents concepts impliqués.

4. **Listage des concepts possibles** : après cette première analyse du contexte, un certain nombre de signifiés sont repérés comme candidats potentiels à l'interprétation. Il s'agit d'une pré-interprétation assez floue, laissant un large degré de liberté à l'indentification précise des signifiés réellement mis en jeu dans l'occurrence présente de l'expression ambiguë.

5. **Repérage de modificateurs** : cette étape est similaire à l'étape 2. Le but est de repérer dans le contexte proche des éléments textuels révélateurs d'*usages attestés*. Ces "modificateurs" peuvent être de n'importe quelle nature (typographique, morpho-syntaxique, rhétorique, pragmatique, socioculturelle) et apportent des indices sur la façon d'explorer les notions retenues. Ces marqueurs linguistiques peuvent éventuellement conduire à des conclusions différentes en fonction des domaines d'application qui ont été remarqués. Ils apportent une indication indirecte concernant les références culturelles implicites présentes dans le contexte (ce que nous avons appelé *connotation* au §2.7.1).

6. **Filtrage des concepts** : les informations qui ont été déduites des marqueurs relevés à l'étape précédente vont servir aux traducteurs à restreindre le champ sémantique qu'ils sont en train d'explorer, ou au contraire, à inclure de nouveaux champs sémantiques qui ne sont pourtant pas contigus au(x) champ(s) de départ. Toutes ces adjonctions et restrictions sont plus ou moins "importantes" en fonction des relations entre les marqueurs textuels et les domaines d'application relevés. Par exemple, si nous avions trouvé un mot comme *enfant(s)* (ou un synonyme) dans le contexte proche de *cellule mère*, nous aurions été amenés à explorer de surcroît la validité des interprétations *qui donne la vie*, *qui nourrit* et *qui élève*. Un M majuscule à *mère* aurait pu (mais dans une moindre mesure, relativement au domaine 'biologie') nous conduire à ajouter //religion// ou //symbolisme// à notre champ sémantique d'investigation.

7. **Listage des sens en présence** : les étapes précédentes permettent aux traducteurs de circonscrire un champ sémantique (pas forcément continu) contenant un ensemble d'interprétations possibles pour l'expression polysémique. Mais au sein de cet ensemble, certaines interprétations sont jugées inacceptables lorsqu'une interprétation potentielle "importante" en contredit une autre "moins importante". Deux interprétations potentielles d'importance équivalente, mais contradictoires, peuvent conduire à être cataloguées "possibles mais peu probables dans ce contexte". Plusieurs interprétations peuvent aussi concourir à renforcer mutuellement leurs plausibilités respectives.

   Il s'agit d'un processus de *filtrage* par renforcement et inhibition mutuels, qui ressemble au comportement des réseaux connexionistes. A la fin de ce processus, certaines interprétations potentielles sont classifiées "très présentes [dans la signification en contexte de cette occurrence]", d'autres seront "faiblement présentes", et les autres seront "absentes". Une situation idéale (pour le traitement automatique orthodoxe des langues naturelles) se présente quand une seule interprétation est "très présente", et toutes les autres sont "absentes".

L'étape 8 consiste à trouver (ou à construire) une expression dans le langage cible qui rend compte d'au moins toutes les interprétations "très présentes", et si possible qui véhicule également des allusions (des connotations) aux interprétations "faiblement présentes". Mais cette étape ne rentre pas dans le cadre de notre recherche.

4.2.2.3 Interprétation théorique

En observant les résultats synthétiques de cette extraction de connaissances, on trouve de fortes similarités avec la sociolinguistique et l'ethnométhodologie. Cela n'est pas étonnant puisque l'étude de phénomènes de connotation, qui sont au cœur de la polysémie d'usage, implique d'observer l'*usage* dans le discours. Cela implique à son tour d'observer une *réalité humaine*, et pas seulement le langage en tant qu'abstraction. C'est pourquoi les étapes 2 et 3 mentionnées ci-dessus, i.e. l'évaluation du contexte sociolinguistique, sont réellement importantes.

Ces étapes nous ramènent à la notion d'*indexicalité*, qui est un des principes de base de l'ethnométhodologie (cf. Garfinkel [41]). Cette notion (introduite p. 43) peut être redéfinie localement ainsi : tout acte énonciatif est relatif à (indexé par) le contexte socioculturel dans lequel il s'inscrit, à savoir :

— une époque de production,

— un lieu de production,

— une langue de production,

— un groupe social auquel appartient l'auteur,

— un groupe social visé,

— et les micro-cultures pratiquées par ces deux groupes.

Les mêmes caractéristiques indexent les actes interprétatifs sur les textes.

Dans notre modèle fonctionnel de la traduction, ces six caractéristiques sont réduites à deux : la *notion* et le *domaine*. Cette réduction peut entraîner une perte d'information, mais on peut considérer qu'en chacune d'elles participent un peu de ces six caractéristiques, en partant du principe qu'une notion et un domaine d'application variant fortement en fonction des micro-cultures locales.

Nous considérerons donc par la suite qu'un couple *[notion, domaine]* est une bonne approximation du contexte socioculturel. Nous nous appuyons pour cela sur les notions théoriques suivantes, proches des concepts de notion et de domaine.

— Tout d'abord, les "directions [sémantiques] générales d'interprétation" mentionnées par les traducteurs se rapprochent significativement de ce que Vignaux appelle les *themata* [131]. Il s'agit de "strates sémantiques profondes" constituant des pôles sémantiques au sein de la structure d'une langue/culture. Ces "tropismes" sémantiques (on s'y réfère inconsciemment mais certainement) sont enracinés très profondément dans notre usage de la langue et sont souvent difficiles à déceler, sauf dans le cas des *topoï,* où ils affleurent de façon explicite. Nous considérons ici les topoï au sens de *lieux communs*, et non pas dans leurs acceptions techniques plus récentes. Un topos constitue une catégorie socioculturelle héritée au travers d'une pratique de la langue, et qui structure notre façon de penser à travers la langue. Par exemple, *la femme est un être faible*. Les topoï forment un fonds de catégories et d'*a priori* communs aux locuteurs d'une même culture.

  Dans notre modèle fonctionnel, les notions ressemblent aux themata, mais sans se situer forcément à un niveau aussi profond. On pourrait plutôt les qualifier de *classèmes* dans la terminologie de la sémantique différentielle. Il s'agit généralement de taxèmes ou de champs.

— D'autre part, le concept de domaine d'application se rapproche de ce que Bourdieu [9] nomme *domaine d'activité*. Sans rentrer dans les détails, l'usage d'un mot dans un texte est grandement influencé par l'activité que ce texte décrit (même implicitement), ou par l'activité dans laquelle il est produit (par exemple, le mot *édition* et ses constructions syntaxiques dans un contexte littéraire *vs* un contexte informatique).

— Pour finir, on notera une forte similitude entre les classements des propositions de traduction (interprétations "très présentes", "faiblement présentes", "possibles", "absentes") avec les taux d'activité symboliques utilisés dans le modèle EDGAR (cf. §3.6.2).

Bien évidemment, cette concordance entre la démarche des traducteurs et des éléments théoriques sociologiques, linguistiques et computationnels est partiellement dûe à une reconstruction de notre part, inhérente à notre connaissance de ces théories. Cependant, lors de la réunion de synthèse finale, les traducteurs ont validé collectivement cette formalisation du contexte en *[notion, domaine]* comme permettant de résumer de façon pertinente leurs diverses démarches individuelles, ainsi que l'étanchéïté des *niveaux de pertinence* de leurs interprétations.

A la lumière de cette observation, nous pouvons tirer un certain nombre de principes pour la constitution d'un modèle informatique de la polysémie d'usage :

— Notre modèle doit contenir un répertoire d'usages attestés pour les expressions à analyser (mais c'est ainsi que sont conçues les vues conceptuelles d'EDGAR).

— La structure de ce répertoire doit refléter les contextes d'analyse sous la forme *[notion, domaine]*, ainsi que des règles relatives à ces contextes d'identification des usages connus, et des règles (également dépendantes des contextes) d'influence mutuelle entre les usages possibles (ce qui se rapproche des règles pragmatiques d'EDGAR).

— Le modèle doit également contenir une partie dynamique concernant l'application des règles d'influence mutuelle en fonction d'informations de toutes natures présentes dans le contexte proche (proche des contraintes sémantiques d'EDGAR).

— Le résultat de l'analyse d'une expression polysémique doit pouvoir nous donner une classification des signifiés présents dans l'interprétation en fonction de leur "degré" de participation.

Cela nous conforte donc dans notre décision de nous référer au modèle EDGAR de Prince comme paradigme pour notre propre modèle. Celui-ci se démarquera d'EDGAR principalement par sa gestion du contexte sociolinguistique permettant de gérer une plus grande flexibilité dans la combinatoire des différents usages d'un mot (et de ne plus limiter son application aux mots courants).

4.3 PELEAS

Le modèle que nous avons mis au point s'appelle PELEAS (pour *Pyramids of Ellipsis as Lexical Entries in Ambiguous Sentences*). Il s'agit d'un modèle pour lequel EDGAR fait figure de paradigme, c'est-à-dire que nous allons largement nous inspirer des principes fondamentaux qui président à EDGAR, mais sans forcément reprendre exactement le même formalisme, eu égard aux remarques que nous avons faites (cf. §3.6.2.4 et §4.2.2.3).

PELEAS est un modèle lexical qui manipule des représentations sémantiques riches et détaillées, ce qui implique des structures computationnelles occupant une place éventuellement importante en mémoire. De plus, l'interprétation d'une occurrence de polysémie d'usage peut prendre un temps non négligable.

C'est pourquoi nous tenterons d'évaluer tout d'abord (§4.3.1) quelles unités lexicales sont intéressantes à représenter de cette façon. Nous présenterons ensuite les composantes statique (§4.3.2) et dynamique (§4.3.3) de PELEAS.

4.3.1 Entrées figurant dans le lexique

La polysémie est un phénomène susceptible d'affecter l'ensemble des unités lexicales pour un lexique non contraint, i.e. lorsqu'on ne force pas la langue à se conformer à une nomenclature. Cependant, nous nous attachons ici plus particulièrement au traitement de la polysémie d'usage. De plus, étant donné que le modèle EDGAR démontre la faisabilité du traitement des polysémies complexes (acception et usage) au sein du paradigme que nous utilisons, notre modèle PELEAS est destiné au traitement de la polysémie d'usage *et* de la polysémie d'acception. Or, nous avons vu (cf. p. 73) que la polysémie d'usage et la polysémie d'acception affectent plus particulièrement les *mots courants* : les mots d'usage quotidien concernant les parties du corps, les processus vitaux, les relations familiales et les mesures de l'espace et du temps, principalement.

Nous pensons donc que seuls des mots considérés comme courants nécessitent d'être représentés à l'intérieur du modèle PELEAS, à condition d'étendre la définition de *courant*. Nous considérerons comme courant soit un mot faisant partie de la catégorie des mots courants selon Prince [86], soit un mot fréquemment utilisé au sein d'un domaine de spécialité et désignant un concept fondamental dans l'univers culturel de ce domaine.

De plus, pour le présent ouvrage, nous avons choisi de nous limiter aux catégories grammaticales pleines (noms communs, verbes et adjectifs). Ces catégories sont celles qui ont le plus grand *apport propre* sémantique (cf. p. 23). Cela permet d'intensifier les contrastes entre le sens prétendu "littéral" et la signification en contexte des usages (l'observation de la pertinence des interprétations en est donc facilitée).

4.3.2 Structure descriptive d'une entrée lexicale

Nous reprenons le vocabulaire d'EDGAR :

— la *structure descriptive* d'une entrée lexicale est la structure de données qui décrit les connaissances du modèle sur l'entrée lexicale de manière statique ;

— le *potentiel sémantique* d'une entrée est l'ensemble des lexicalisations des parcours interprétatifs possibles, avec leurs relations hiérarchiques et leurs contraintes sémantiques ;

— la *participation* d'un élément du potentiel sémantique évalue la pertinence de cet élément dans la construction du sens en contexte d'une occurrence de l'entrée lexicale, c'est-à-dire avec quelle pertinence on peut dire que cet élément décrit ou joue le rôle de synonyme local de l'entrée ;

— une *contrainte sémantique* entre deux lexicalisations est une relation qui indique une modification du taux de participation d'une des lexicalisations en fonction de la participation de l'autre ;

— une *règle* ou *condition contextuelle* est un élément réactif de la structure descriptive qui permet de contraindre le taux de participation de certaines lexicalisations en fonction du contexte au démarrage du processus d'interprétation.

La structure descriptive d'une entrée lexicale est composée de son potentiel sémantique auquel est associé un ensemble de conditions contextuelles.

Le processus d'interprétation en contexte d'une occurrence d'une entrée lexicale consiste à attribuer une participation au sens en contexte pour chaque lexicalisation du potentiel sémantique. Cette participation s'exprime par des *taux d'activité* symboliques. Ils sont attribués initialement à certains éléments du potentiel sémantique par l'application des conditions contextuelles, puis sont *propagés* aux autres éléments grâce aux contraintes sémantiques et aux *règles de propagation par défaut* entre les lexicalisations voisines ou interdépendantes.

4.3.2.1 Taux d'activité

Nous empruntons au modèle EDGAR son utilisation de taux d'activité symbolique. Rappelons que ces taux expriment dans quelle mesure un élément du potentiel sémantique d'un mot m participe à l'interprétation d'une occurrence de m en contexte.

Cette technique a l'avantage de correspondre aux observations que nous avons présentées (cf. §2.8 et §3.7), ainsi que de refléter assez fidèlement les séparations symboliques entre les différentes catégories d'interprétations potentielles des traducteurs que nous avons observés (cf.§4.2) :

— Lors du traitement d'une ambiguïté lexicale, les diverses interprétations possibles sont pondérées par leur *importance* : certaines sont très importantes (tout à fait pertinentes dans le contexte), d'autres sont moyennement importantes (elles sont valides mais jouent un rôle mineur dans le processus d'interprétation), et d'autres encore sont sans importance (non pertinentes dans le contexte).

— Le fait de recourir à des valeurs symboliques permet de rendre compte du "fossé psychologique" entre ces différentes catégories : il n'y a pas de continuité entre les différents degrés.

En outre, l'utilisation d'un taux d'activité négatif permet d'ajouter aux sorties de PELEAS des informations potentiellement utiles à un niveau plus élevé du système de traitement automatique des langues naturelles pour la gestion de la contradiction avec le contexte.

Nous avons de plus ajouté un symbole "neutre" permettant de signaler qu'un élément du potentiel sémantique n'a pas encore été traité afin d'optimiser l'efficacité de l'algorithme d'interprétation dynamique (cf. §4.3.3) et d'interdire les conflits d'attribution de taux d'activité.

Définition 1 (Taux d'activité) *L'ensemble des valeurs possibles pour les taux d'activité est*

$$\mathcal{V} = \{\mathsf{salient}, \mathsf{valid}, \mathsf{inhibited}, \mathsf{negated}, \mathsf{unknown}\}$$

Ces symboles s'interprètent ainsi :

— *salient* : élément particulièrement pertinent jouant un rôle majeur dans l'interprétation en contexte de l'occurrence examinée ;

— *valid* : élément qui n'est pas contredit par le contexte, mais qui joue un rôle mineur dans l'interprétation. On peut le considérer comme une information "accompagnante".

— *inhibited* : élément rendu non pertinent par le contexte, qui n'a aucune pertinence et ne joue aucun rôle dans l'interprétation courante. Cette valeur correspond au taux d'activité "ignoré" d'EDGAR, et représente l'opération différentielle de *virtualisation* d'un sème. Nous avons changé l'étiquette de ce symbole pour mieux rendre compte du caractère actif de l'opération de virtualisation : le sème n'est pas seulement laissé pour compte (ignoré) mais a été désactivé par le contexte.

— *negated* : élément pertinent mais de façon négative : il s'agit généralement d'un trait rendu saillant par une contradiction explicite. Cette valeur correspond au taux d'activité "inhibé" d'EDGAR. Nous avons cette fois-ci changé l'étiquette dans le seul but de la rendre plus parlante.

— *unknown* : élément dont le taux d'activité n'a pas encore été calculé.

Pour faciliter la définition des *contraintes sémantiques* qui seront présentées au §4.3.2.3, nous définissons trois opérateurs sur cet ensemble : l'opérateur d'opposition $\neg$ qui servira à exprimer la contrainte d'opposition, l'opérateur d'incrémentation $\uparrow$ pour exprimer la contrainte de renforcement et l'opérateur de décrémentation $\downarrow$ pour exprimer la contrainte d'affaiblissement.

Définition 2 (Opérateurs $\neg$, $\uparrow$ et $\downarrow$) *Ces trois opérateurs unaires de $\mathcal{V}$ dans $\mathcal{V}$ sont définis par le tableau suivant :*

	salient	valid	inhibited	negated	unknown
$\neg$	negated	valid	inhibited	salient	unknown
$\uparrow$	salient	salient	valid	negated	unknown
$\downarrow$	valid	inhibited	inhibited	valid	unknown

4.3.2.2 Potentiel sémantique

Le potentiel sémantique d'une entrée lexicale e contient l'ensemble de ce que nous appelons ses *descripteurs*, i.e. les lexicalisations des spécifications intensionnelles partielles qui sont susceptibles d'être sollicitées en contexte par le mot (ou l'expression) e. De plus, le potentiel sémantique de e contient l'ensemble des contraintes sémantiques qui expriment les tolérances mutuelles de co-participation entre les différentes interprétations candidates co-occurrentes.

Les descripteurs d'une entrée lexicale peuvent avoir quatre statuts hiérarchisés, suivant qu'ils sont la lexicalisation :

1. d'un domaine,
2. d'une notion,
3. d'une vue conceptuelle,
4. ou d'un trait (une propriété spécialisante d'une vue conceptuelle).

Représentation du contexte

Nous avons défini au §4.2.2.3 une approximation du contexte d'énonciation comme une paire *[notion, domaine]*. En termes ethnométhodologiques, il s'agit d'un *index* qui permet de repérer un acte énonciatif ainsi que les lois locales d'interprétation qui lui sont associées. Cela revient à dire que l'acte énonciatif est localisé dans un repère, comme illustré sur la figure 4.3.

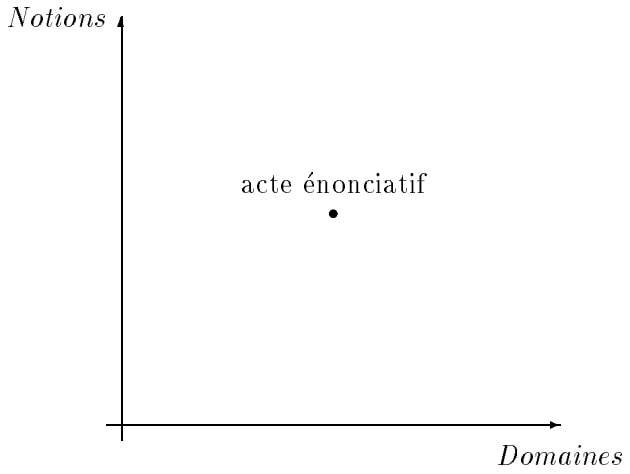


FIG. 4.3: Repérage d'un acte énonciatif par son contexte

Les deux axes *Notions* et *Domaines* sont considérés comme quasiment indépendants l'un de l'autre (d'où l'orthogonalité dans le repère). Logiquement, notre modèle devrait donc contenir une seule couche de sommets bi-étiquetés pour représenter le contexte. Cependant, nous avons choisi de séparer cette représentation du contexte en deux couches hiérarchisées (*Domaines* $>$ *Notions*) pour les raisons suivantes :

1. Après concertation avec les traducteurs que nous avons observés, ces derniers ont fait remarquer qu'ils rencontraient usuellement plus de notions que de domaines. Terminologiquement, il est donc

plus efficace (moins de place consommée, moins de temps nécessaire pour retrouver un élément) de transformer notre index contextuel à double entrée par un index hiérarchisé, une seule entrée à la fois, en mettant en haut de la hiérarchie les entrées les moins nombreuses. C'est en outre un avantage évident pour la mise en œuvre informatique de notre modèle.

2. Sur le plan purement informatique cette fois, il nous a semblé qu'une mise en œuvre de PELEAS ne serait réellement efficace (à grande échelle) que si elle est capable de coopérer avec d'autres outils automatiques d'analyse de textes. Or, en l'état actuel des choses, il est plus facile d'automatiser une discrimination sur corpus de domaines que de champs conceptuels. Par conséquent, placer les domaines au-dessus des notions permet, dans l'éventualité d'une coopération PELEAS-analyseur de domaines, de produire automatiquement des potentiels sémantiques possédant une structuration partielle des couches contextuelles. Et dans ce cas, il est plus efficace de trouver les critères de discrimination le plus haut possible dans le graphe, qui possède ainsi une macro-structure plus harmonieusement répartie.

Structure

De façon générale, nous n'allons pas réutiliser la structure "plate" décrite dans EDGAR. Nous jugeons plus naturel d'utiliser une structure hiérarchique ("tridimensionnelle") permettant d'étager les descripteurs en fonction de leur niveau de généralité (afin d'adopter une représentation naturellement adaptée à une profondeur sémantique variable).

La structure globale d'un potentiel sémantique peut être conçue de deux manières converses :

— soit comme un graphe possédant des contraintes structurelles particulières ; il s'agit en fait d'un *graphe pyramidal* (dont la définition est donnée ci-dessous) de hauteur 5,

— soit comme un hypergraphe d'ordre 4, ce que nous appellerons une *pyramide* (cette structure est définie à l'annexe F, ainsi que sa relation d'équivalence avec les graphes pyramidaux).

Structure mathématique

La structure de données correspondant à un potentiel sémantique est basée sur la structure mathématique de graphe pyramidal qui consiste en fait à superposer une structure d'arbre n-aire avec des structures de graphes. Elle est définie le plus génériquement possible, en la paramétrant par le type Σ des étiquettes de lexicalisation, l'ensemble V des taux d'activité symbolique et l'ensemble $R(V)$ des règles de calcul qui vont servir à la propagation de l'activité.

Définition 3 (Graphe pyramidal) *Soit $\mathcal{A}_{\Sigma,V,R(V)} = \langle S_{\Sigma,V}, A_{R(V)} \rangle$ un arbre n-aire dont chaque sommet $s \in S_{\Sigma,V}$ est* étiqueté *dans Σ et* valué *dans V (i.e. il existe une fonction* etiq *de $S_{\Sigma,V}$ dans Σ et une fonction* val *de $S_{\Sigma,V}$ dans V), et dont chaque arc est* étiqueté *par un élément de $R(V)$ (on notera un arc $x \xrightarrow{f} y$ où $x, y \in S_{\Sigma,V}$ et $f \in R(V)$ est une fonction de V dans V).*

Un graphe pyramidal *est un graphe orienté construit en ajoutant à un tel arbre $\mathcal{A}_{\Sigma,V,R(V)}$ de nouveaux arcs de telle sorte qu'à tout arc $x \xrightarrow{f} y$ de $\mathcal{A}_{\Sigma,V,R(V)}$ correspond un nouvel arc $y \xrightarrow{f'} x$, et que tous les sommets de tout ensemble de sommets fils d'un même sommet père dans $\mathcal{A}_{\Sigma,V,R(V)}$ soient reliés deux-à-deux.*

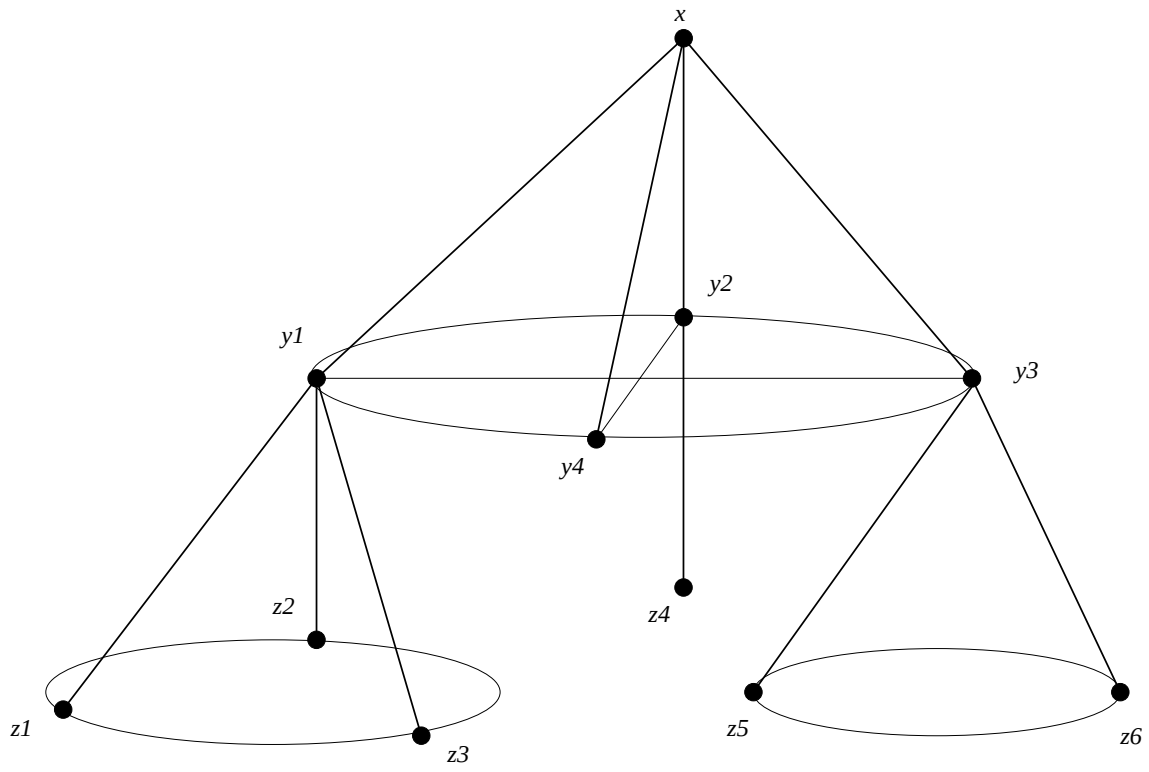


FIG. 4.4: Exemple de graphe pyramidal de hauteur 3

Formellement, on notera un graphe pyramidal $G_{\Sigma,V,R(V)} = \left\langle \mathcal{A}_{\Sigma,V,R(V)}, A'_{R(V)}, A''_{R(V)} \right\rangle$ *où* $\mathcal{A}_{\Sigma,V,R(V)}$ *est un arbre tel que décrit ci-dessus et :*

— $\left(x \xrightarrow{f} y\right) \in A'_{R(V)} \Rightarrow \exists f' \in R(V), \left(y \xrightarrow{f'} x\right) \in A_{R(V)}$ *avec* $x, y \in S_{\Sigma,V}$ *et* $f \in R(V)$ *;*

— $\forall \{y_1, \ldots, y_n\} \subset S_{\Sigma,V} / \exists! x \in S_{\Sigma,V}, \left(x \xrightarrow{f_1} y_1\right), \ldots, \left(x \xrightarrow{f_n} y_n\right) \in A_{R(V)}$ *on a :*

- $\forall i, j, i \neq j, \exists g \in R(V) / y_i \xrightarrow{g} y_j \in A'_{R(V)}$ *;*
- $\left(x \xrightarrow{f} y\right) \in A''_{R(V)} \Rightarrow \exists! z \in S_{\Sigma,v}, \exists g, g' \in R(V)$ *tels que* $\left(z \xrightarrow{g} x\right) \in A_{R(V)}$ *et* $\left(z \xrightarrow{g'} y\right) \in A_{R(V)}$.

Définition 4 (Hauteur, profondeur, racine, feuilles) *Par analogie structurelle, on appelle* hauteur *d'un graphe pyramidal* $G_{\Sigma,V,R(V)} = \left\langle \mathcal{A}_{\Sigma,V,R(V)}, A'_{R(V)}, A''_{R(V)} \right\rangle$ *la hauteur de l'arbre* n*-aire* $\mathcal{A}_{\Sigma,V,R(V)}$*. De même, la racine et les feuilles de l'arbre* n*-aire seront dites respectivement* racine *(notée* Rac*) et* feuilles *du graphe pyramidal.*

La profondeur *(notée* Prof*) d'un sommet* s *est la différence entre la hauteur de l'arbre* n*-aire sous-jacent et celle du sous-arbre de racine* s.

Propriété 1 (Invariance par symétrie d'échelle) *Etant donnés un graphe pyramidal noté* $G_{\Sigma,V,R(V)} = \left\langle \mathcal{A}_{\Sigma,V,R(V)}, A'_{R(V)}, A''_{R(V)} \right\rangle$ *, et* $\mathcal{B}_{\Sigma,V,R(V)}$ *un sous-arbre de* $\mathcal{A}_{\Sigma,V,R(V)}$*, soit la structure* $H_{\Sigma,V,R(V)} =$

$\left\langle \mathcal{B}_{\Sigma,V,R(V)}, B'_{R(V)}, B''_{R(V)} \right\rangle$*, où* $B'_{R(V)}$ *et* $B''_{R(V)}$ *sont les restrictions de* $A'_{R(V)}$ *et* $A''_{R(V)}$ *aux arcs dont les sommets appartiennent à* $\mathcal{B}_{\Sigma,V,R(V)}$*.* $H_{\Sigma,V,R(V)}$ *est aussi un graphe pyramidal (démonstration triviale).*

La figure 4.4 (p. 111) représente un graphe pyramidal de hauteur 3. Sur cette figure, les arcs sont représentés non orientés et non étiquetés afin de faciliter la lecture. Cependant, eu égard à la définition que nous avons donnée, chaque arc sur cette figure (i.e. chaque trait ou courbe entre deux points) représente *deux* arcs du graphe pyramidal, orientés de façons contraires et portant chacun une étiquette. Les arcs plus épais sur la figure permettent de repérer les arcs de l'arbre n-aire sous-jacent.

Structure de données

D'un point de vue informatique, nous nous appuyons sur la structure mathématique de graphe pyramidal pour définir la structure de données qui permet de représenter une entrée lexicale dans PELEAS.

Définition 5 (Potentiel sémantique) *Le* potentiel sémantique *d'une entrée lexicale* e *est un graphe pyramidal de profondeur 5 au plus dont la racine a pour étiquette* e*. Les ensembles* Σ*,* V *et* $R(V)$ *introduits plus haut sont instanciés de la manière suivante :*

- Σ *est un ensemble de chaînes de caractères qui permet de noter les lexicalisations des descripteurs d'une entrée lexicale :*
  - *les domaines, situés au niveau de profondeur 1 par rapport à la racine ;*
  - *les notions, au niveau de profondeur 2 ;*
  - *les vues conceptuelles, au niveau de profondeur 3 ;*
  - *et les traits, au niveau de profondeur 4.*
- V *est l'ensemble* $\mathcal{V}$ *des taux d'activité définis au §4.3.2.1 ;*
- $R(V)$ *est l'ensemble des contraintes sémantiques définies au §4.3.2.3.*

Sous-graphes vides et sommets fantômes

Utiliser une structure de graphe pyramidal de profondeur 5 comme structure fondamentale de notre modèle nous conduit à manipuler des potentiels sémantiques très riches, comportant un nombre éventuellement important de détails. Cependant, certaines données nécessaires à la constitution d'un potentiel sémantique peuvent ne pas être connues lors de la phase de construction du lexique. C'est pourquoi nous voulons être capable de signaler que certaines portions du potentiel sémantique sont "incomplètes", et de reporter leur complétion à un moment ultérieur. Nous proposons deux mécanismes pour cela :

1. Un sommet (autre que l'entrée lexicale ou un trait) peut n'avoir aucun sommet subordonné (il n'existe aucun arc vertical avec ce sommet en partie gauche). Ce sommet est alors la racine d'un *sous-graphe vide* (cf. figure 4.5).
2. Il peut manquer un niveau de détermination dans la structure du potentiel sémantique. Par exemple, pour un domaine donné, les vues conceptuelles courantes, ainsi que leurs traits pertinents, ont été répertoriées, mais aucune notion n'a encore été identifiée. Pour indiquer ce genre de "niveaux manquants", on utilisera alors un *sommet fantôme* étiqueté $\emptyset$ (cf. figure 4.6).

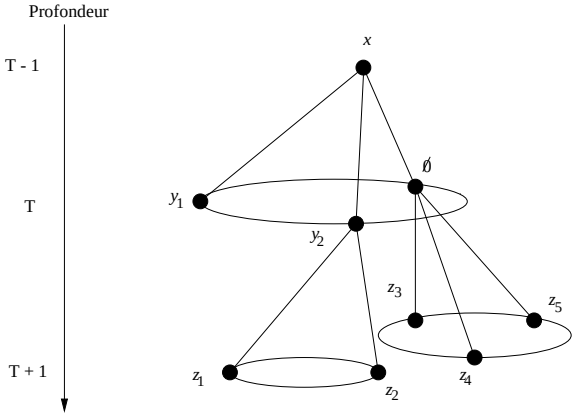


FIG. 4.5: Sous-graphe vide

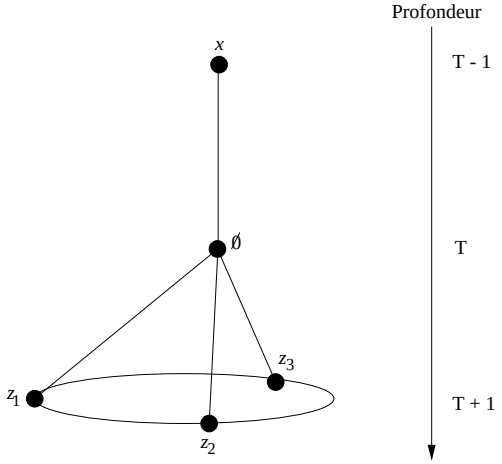


FIG. 4.6: Sommet fantôme

L'utilisation des sommets fantômes fournit un mécanisme pour la construction incrémentale des potentiels sémantiques : lorsque des données correspondent à des sommets de profondeur $T + 1$ mais que les catégories de profondeur T qu'ils sont sensés décrire ne sont pas encore connues, alors tous ces sommets de profondeur $T + 1$ peuvent être reliés à un unique sommet fantôme de profondeur T, comme sur la figure 4.6. Puis, lorsque des informations pourront être ajoutées à la profondeur T, les sommets de profondeur $T + 1$ seront redistribués entre les différents sommets qui viendront remplacer le sommet fantôme.

De plus, comme nous ne souhaitons pas entrer dans une démarche de description exhaustive, les catégories représentées par des sommets dans un potentiel sémantique sont *épistémologiquement ouvertes*. Cela nous permet de conserver un sommet fantôme à l'intérieur d'un groupe de sommets de même profondeur, comme illustré sur la figure 4.5. Ainsi, des sommets de profondeur T peuvent être discriminés de façon opportuniste du sommet fantôme. Chaque nouveau sommet y_i entraîne avec lui un certain nombre de sommets z_j, tandis que les sommets de profondeur $T + 1$ pour lesquels on ne connaît pas encore de superordonné restent sous le sommet fantôme.

4.3.2.3 Relations entre les sommets

Arcs verticaux

Les arcs verticaux dénotent une relation sémantique indifférenciée *est-décrit-par* : tous les sommets dont le type est de rang n sont des *descripteurs* du sommet dont le type est de rang $n - 1$ auquel ils sont reliés par un arc vertical.

Cette relation de description (qui subsume les relations sémantiques d'hyponymie, d'instanciation, de synonymie, d'antonymie, de méronymie et de modification, cf. Evens *et al.* [31]) nous permet de nous ramener à une pure observation lexicale. En effet, avec ce modèle, nous sommes plus intéressé par les récurrences lexicales des termes apparaissant dans les paraphrases désambiguïsantes des acceptions ou usages particuliers d'un mot dans des contextes ambigus (cf. Prince [85, 87]). Nous ne nous fixons donc pas pour but de catégoriser des connaissances.

Arcs horizontaux

Les arcs horizontaux d'un potentiel sémantique portent, tout comme dans EDGAR, des *contraintes sémantiques*.

Il s'agit de règles exprimant les contraintes imposées par les descriptions intensionnelles partielles implicites des vues conceptuelles et de leurs traits. Ces contraintes réintroduisent les contraintes d'intégrité intensionnelle et de restriction de la sélection issues des sémantiques référentielle et inférentielle. Mais il ne s'agit pas d'un retour à l'intensionnalité puisque nous ne travaillons que sur des récurrences de lexicalisation. Ces contraintes expriment plutôt des régularités dans la cooccurrence ou dans la non cooccurrence de lexicalisations.

Ces contraintes n'opèrent que sur des couples de sommets vues conceptuelles ou traits afin de laisser un plus grand degré de liberté aux couches contextuelles (un modèle comme PELEAS n'a en effet d'intérêt qu'appliqué dans un contexte le plus ouvert possible).

Il existe quatre contraintes sémantiques possibles entre des sommets de même niveau de profondeur (donc de même statut) :

1. *Contrainte d'implication* : le sommet x **implique** le sommet y si l'usage du descripteur x se caractérise par un usage de même pertinence du descripteur y à contexte égal, i.e. si le taux d'activité de x se transmet à y pour toute occurrence, ce qu'on notera $x \longrightarrow y$.

$$x \longrightarrow y \Longrightarrow \mathrm{val}(y) \leftarrow \mathrm{val}(x)$$

2. *Contrainte d'opposition* : les sommets x et y sont dits **en opposition** si les emplois des descripteurs x et y ne peuvent jamais être simultanés dans le même contexte, i.e. si leurs taux d'activité sont opposés quelles que soient les circonstances, ce qu'on notera $x \leftarrow\!\!/\!\!\rightarrow y$.

$$x \leftarrow\!\!/\!\!\rightarrow y \Longrightarrow \mathrm{val}(x) \leftrightarrow \neg\, \mathrm{val}(y)$$

   N.B. : cet arc est toujours orienté dans les deux sens (on pourra considérer qu'il s'agit en fait de deux arcs).

3. *Contrainte de renforcement* : le sommet x **renforce** le sommet y si l'usage du descripteur x implique un usage encore plus pertinent du descripteur y dans le même contexte, i.e. si le taux d'activité de y est toujours un cran supérieur à celui de x, ce qu'on notera $x \stackrel{+}{\longrightarrow} y$.

$$x \stackrel{+}{\longrightarrow} y \Longrightarrow \mathrm{val}(y) \leftarrow\!\!\uparrow \mathrm{val}(x)$$

4. *Contrainte d'affaiblissement* : le sommet x **affaiblit** le sommet y si l'usage du descripteur x implique un usage moins pertinent du descripteur y dans le même contexte, i.e. si le taux d'activité de y est toujours un cran inférieur à celui de x, ce qu'on notera $x \stackrel{-}{\longrightarrow} y$.

$$x \stackrel{-}{\longrightarrow} y \Longrightarrow \mathrm{val}(y) \leftarrow\!\!\downarrow \mathrm{val}(x)$$

4.3.2.4 Conditions contextuelles

Nous avons vu à la p. 102 que les traducteurs emploient par deux fois un processus qui consiste à extraire des informations de toutes natures du contexte proche afin de diriger leur interprétation. Dans un cas, il

s'agit de repérer des indices donnant les premières directions d'interprétation, et dans l'autre, d'extraire des "modificateurs" qui influencent le raffinage des interprétations en cours.

Nous nous proposons de factoriser ces deux étapes en un seul mécanisme : les *conditions contextuelles*. Ce sont des éléments réactifs qui sont déclenchés en fonction d'informations extraites du contexte, et qui ont pour effet de fixer le taux d'activité de certains sommets du potentiel sémantique. Ces éléments ne font pas partie de la structure de graphe pyramidal, mais y sont attachées de manière globale comme liste de propriétés.

Les conditions contextuelles sont semblables aux règles pragmatiques d'EDGAR et permettent d'initialiser les calculs d'interprétation. Il s'agit de règles dont la prémisse est une formule du calcul propositionnel (où les propositions sont des propriétés du contexte de l'occurrence de l'entrée lexicale) et dont la conclusion est un ensemble de couples associant un sommet du potentiel sémantique à la valeur qui doit lui être attribuée si la prémisse est vraie pour le contexte d'analyse de l'occurrence.

Définition 6 (Contexte d'analyse) *Pour une occurrence $\dot{e}$ de l'entrée lexicale e, on appelle $\kappa_{\dot{e}}$ le contexte d'analyse de cette occurrence. Dans le cas d'un système de traitement automatique des langues naturelles hébergeant une mise en œuvre de* PELEAS, *$\kappa_{\dot{e}}$ est l'ensemble des représentations présentes en mémoire dans ce système au moment où $\dot{e}$ est analysée.*

La prémisse d'une condition contextuelle est une formule bien formée du calcul propositionnel composée d'un ensemble $\{p_i\}$ de prédicats évalués sur $\kappa_{\dot{e}}$ connectés par $\wedge$, $\vee$ et $\neg$, ainsi que les parenthèses. Chacun de ces prédicats évalue une propriété du contexte local (on trouvera une justification computationnelle de cette organisation au §5.4.4).

Une condition contextuelle se présente donc ainsi :

$$\Phi \Longrightarrow \{(s_i, v_i), s_i \in S, v_i \in \mathcal{V}\}$$

ce qui signifie que les sommets s_i se voient attribuer le taux d'activité v_i si la formule Φ est évaluée à VRAI sur le contexte local.

Définition 7 (Ensemble des conditions contextuelles) *Les conditions contextuelles attachées à l'entrée lexicale e sont une propriété du potentiel sémantique G de e :*

$$CT(G) = \{\Phi \Longrightarrow \{(s_i, v_i)\}\}$$

Finalement, pour des raisons pratiques, nous avons choisi de particulariser une condition contextuelle pour chaque entrée lexicale. Cette condition contextuelle *optionnelle* porte le nom prédéfini DEFAUT. Si elle existe, elle n'est appliquée que dans le cas où aucune autre condition contextuelle ne s'applique. Elle permet d'activer les sens *usuels* de l'entrée lexicale en l'absence de toute autre information. Nous insistons cependant pour dire qu'il ne s'agit pas d'indiquer un sens premier ou littéral, mais des usages courants (les plus utilisés).

Les conditions contextuelles et les contraintes sémantiques font partie des potentiels sémantiques car elles sont propres à chaque entrée lexicale. Cela permet d'exprimer le fait que la lexicalisation d'un signifié influe sur son usage dans la langue, plus que son contenu sémantique. Nous nous situons donc résolument au niveau du matériau brut de la langue.

4.3.3 Propagation de l'activité

La propagation de l'activité d'un sommet à l'autre du potentiel sémantique est gérée par un ensemble de règles de calcul étiquetant les arcs entre ces sommets. Il s'agit en fait de représenter grâce à ces règles les processus de "filtrage" utilisés par les traducteurs (cf. p. 103) en modélisant les influences mutuelles des différents éléments d'interprétation.

Cette propagation d'activité constitue la composante dynamique de PELEAS. Elle s'effectue sur les sommets du potentiel sémantique et son résultat est une *configuration* particulière du potentiel sémantique.

Définition 8 (Configuration) *Soit G le potentiel sémantique de l'entrée lexicale e. La* configuration *de G pour l'occurence $\dot{e}$ est l'ensemble $\{(s, v), \forall s \in S, v \in \mathcal{V} - \{\mathsf{unknown}\}\}$ des sommets de G affectés d'un taux d'activité calculé, sachant que les prémisses des conditions contextuelles de G ont été évaluées sur $\kappa_{\dot{e}}$, le contexte local de $\dot{e}$.*

Le calcul d'une configuration est basé sur l'application des conditions contextuelles et des contraintes sémantiques propres à l'entrée lexicale, mais représente une description *minimale* du potentiel sémantique. La plus grande partie de la propagation des taux d'activité entre sommets est réalisée par des règles de *défaut*, qui sont inhérentes au modèle PELEAS. Elles sont identiques pour toutes les entrées lexicales, et ne sont donc pas représentées dans les potentiels sémantiques.

4.3.3.1 Règles de propagation par défaut

Ces règles maintiennent l'intégrité des résultats partiels successifs conduisant à la configuration finale, et permettent également d'éviter les conflits d'attribution d'activité.

Tous les arcs du potentiel sémantique dont on n'a pas spécifié l'étiquette vont pouvoir propager leur activité entre les sommets qu'ils connectent grâce à ces règles de propagation par défaut. Il en existe trois : horizontale, ascendante et descendante, suivant les arcs sur lesquels elles vont s'appliquer.

Les règles horizontale et descendante mettent en œuvre un système de "taxation" de l'activité basé sur la notion de *co-participation* : un élément activé propage aux éléments qui lui sont le plus proches une activité légèrement dégradée. Cela s'apparente en fait aux techniques de *priming* (amorçage) de la psychologie cognitive. L'idée en est que, au sein d'un réseau mémoriel fortement connecté, on ne peut activer un élément (un "souvenir" ou une "connaissance") sans activer légèrement les éléments connexes.

La règle ascendante est une règle de "positivation" : tout sommet activé propage à son superordonné son activité *en valeur absolue* (i.e. *negated* est ramené à *salient*). En effet, si un élément descriptif spécialisé (de grande profondeur, c'est-à-dire à un niveau bas du potentiel sémantique) est activé d'une certaine façon, alors forcément, les niveaux de description plus généraux ont eux aussi été activés.

Définition 9 (Règles de propagation par défaut) *Ces trois règles sont représentées par les fonctions F (propagation horizontale), G (propagation descendante) et H (propagation ascendante) de $\mathcal{V}$ dans $\mathcal{V}$. Elles sont définies par le tableau suivant :*

	salient	valid	inhibited	negated	unknown
F	valid	valid	unknown	valid	unknown
G	valid	valid	inhibited	valid	unknown
H	salient	valid	unknown	salient	unknown

Ces règles sont techniquement semblables aux contraintes sémantiques, mais elles expriment des relations que nous estimons être des invariants de la langue. Au contraire, les contraintes sémantiques représentent des violations locales de ces règles, et leur domaine d'application (i.e. les descripteurs entre lesquels elles supplantent les règles de défaut) est spécifique de chaque entrée lexicale et de ses usages.

Méta-règles

Le modèle PELEAS dispose en outre de deux méta-règles :

1. *Résolution de conflits* : une règle (contrainte sémantique ou de propagation par défaut) ne s'applique que si les sommets qu'elle est susceptible d'affecter n'ont pas encore d'activité affectée (activité *unknown*).
2. *Macro-inhibition* : un sommet qui reçoit l'activité *inhibited* propage immédiatement cette activité à tous ses descendants.

Les méta-règles ont pour but de supprimer les conflits d'attribution d'un taux d'activité à un sommet du potentiel sémantique. Elles n'ont pas pour but d'interdire les contradictions sémantiques : ainsi, au terme du processus d'interprétation, deux sommets reliés par une contrainte d'opposition sémantique peuvent avoir le même taux d'activité. C'est la marque d'un paradoxe, comme dans l'expression *un tyran éclairé* (avec le sens moderne de tyran).

4.3.3.2 Algorithme de propagation

Tout comme dans EDGAR, et au contraire de tous les autres systèmes de représentation du lexique actuels, les conditions contextuelles de PELEAS *amorcent* le calcul d'une configuration au lieu de le compléter. Cette approche est cohérente à la fois avec la conception de l'ambiguïté comme choix simultanément nécessaire et impossible (cf. p. 12) et avec la démarche des traducteurs (cf. p 102). Le traitement d'une ambiguïté n'est envisageable qu'à partir du moment où l'on est capable de recenser dans son contexte proche suffisamment d'informations pour avoir une impression de "déjà-vu", sans pourtant pouvoir se replacer exactement dans un cas particulier recensé.

Le principe de notre algorithme sera donc le suivant :

1. Essayer de vérifier au moins une condition contextuelle. Si ce n'est pas possible, alors on applique la condition de défaut. Au terme de cette étape, un certain nombre de sommets du potentiel sémantique ont reçu une valeur d'activité.
2. Pour chacun des sommets activés (énumérés dans l'ordre où ils ont été activés), propager l'activité
   (a) par les contraintes sémantiques ;
   (b) par défaut aux frères (homogénéité du niveau de généralité) ;
   (c) par défaut en descendant le long des arcs verticaux ;
   (d) par défaut en remontant le long des arcs verticaux.
3. Recommencer à propager l'activité tant que des nouveaux sommets reçoivent une valeur d'activité.

Formellement, l'algorithme se présente ainsi pour une occurrence $\dot{e}$ d'une entrée lexicale e de potentiel sémantique $G =< S, A, A', A'' >$. On appellera $K(\dot{e})$ la configuration du potentiel sémantique pour $\dot{e}$.

Début
*** Initialisation de la configuration ***
$K(\dot{e}) \leftarrow \{(s, \textit{unknown}), \forall s \in S\}$

*** Evaluation des conditions contextuelles ***
$L \leftarrow \emptyset$ *** Liste des sommets activés ***
contexte $\leftarrow$ Faux
pour tout $R = (\Phi \Longrightarrow \{(s_i, v_i)\}) \in CT(G)$
si $|\Phi|_{\kappa(\dot{e})} =$ Vrai **alors**
pour tout $(s_i, v_i) \in R$
$\mathrm{val}(s_i) \leftarrow v_i$
$L \leftarrow L \cup \{s_i\}$
fin pour tout
contexte $\leftarrow$ Vrai
fin si
fin pour tout

*** Vérifier qu'au moins une règle s'applique ***
si contexte $=$ Faux **alors**
pour tout $(s_i, v_i) \in$ DEFAUT
$\mathrm{val}(s_i) \leftarrow v_i$
$L \leftarrow L \cup \{s_i\}$
fin pour tout
fin si

*** Tant que de nouveaux sommets ont été activés ***
tant que $L \neq \emptyset$
soit $s \in L$ *** Nouveau sommet à traiter ***
$L \leftarrow L - \{s\}$ *** qu'on ne retraitera plus ***

*** Macro-inhibition ***
si $\mathrm{val}(s) =$ *inhibited* **alors**
$D \leftarrow \emptyset$
pour tout $(s \rightarrow s_i) \in A \cup A$
si $\mathrm{Prof}(s) - \mathrm{Prof}(s_i) = 1$ **alors**
$D \leftarrow D \cup \{s_i\}$
fin si
fin pour tout
tant que $D \neq \emptyset$
soit $d \in D$
$D \leftarrow D - \{d\}$
$\mathrm{val}(d) \leftarrow$ *inhibited*
pour tout $(s \rightarrow s_i) \in A \cup A'$
si $\mathrm{Prof}(s) - \mathrm{Prof}(s_i) = 1$ **alors**

$D \leftarrow D \cup \{s_i\}$
fin si
fin pour tout
fin tant que

*** Propagation proprement dite ***
pour tout $(s \xrightarrow{f} s_i) \in A \cup A' \cup A''$

*** Appliquer les contraintes sémantiques ***
si $\mathrm{Prof}(s) = \mathrm{Prof}(s_i) \wedge f \neq \emptyset \wedge \mathrm{val}(s_i) =$ *unknown* **alors**
$\mathrm{val}(s_i) \leftarrow f(\mathrm{val}(s))$
$L \leftarrow L \cup \{s_i\}$
fin si

*** Appliquer la propagation horizontale ***
si $\mathrm{Prof}(s) = \mathrm{Prof}(s_i) \wedge f = \emptyset \wedge \mathrm{val}(s_i) =$ *unknown* **alors**
$\mathrm{val}(s_i) \leftarrow F(\mathrm{val}(s))$
$L \leftarrow L \cup \{s_i\}$
fin si

*** Appliquer la propagation descendante ***
si $\mathrm{Prof}(s) = 1 - \mathrm{Prof}(s_i) \wedge \mathrm{val}(s_i) =$ *unknown* **alors**
$\mathrm{val}(s_i) \leftarrow G(\mathrm{val}(s))$
$L \leftarrow L \cup \{s_i\}$
fin si

*** Appliquer la propagation ascendante ***
si $\mathrm{Prof}(s) = 1 + \mathrm{Prof}(s_i) \wedge \mathrm{val}(s_i) =$ *unknown* $\wedge\, \mathrm{Prof}(s_i) \neq 0$ **alors**
$\mathrm{val}(s_i) \leftarrow H(\mathrm{val}(s))$
$L \leftarrow L \cup \{s_i\}$
fin si
fin pour tout
fin tant que
Fin.

Cet algorithme a été conçu en se basant sur le modèle des systèmes de propagation de contraintes. Les contraintes sémantiques y jouent un rôle équivalent à celui des opérateurs de coercion de type du GENERATIVE LEXICON (cf. §3.6.1.2), sauf qu'elles contraignent la cohérence entre les activités de différents sommets, et non plus entre les types de différents qualiae. Parallèlement à ces contraintes, la relation de co-participation (i.e. les arcs horizontaux non étiquetés) permettent de relâcher les contraintes sur les schémas de lexicalisation, et autorisent le système à prendre en compte des expressions originales (non répertoriées) à condition qu'elles comportent au moins des parties de lexicalisations répertoriées.

En somme, cet algorithme se comporte comme un système linéaire d'équations de récurrence sur n variables (correspondant aux n sommets d'un potentiels sémantique hormis son entrée lexicale), pour lequel le résultat de l'application des conditions contextuelles forme les conditions initiales.

4.3.4 Estimation des interprétations

Le modèle PELEAS est *a priori* conçu pour fonctionner au mieux dans une perspective de coopération avec d'autres modules de traitement automatique des langues naturelles qui sont capables de gérer des représentations ensemblistes. Mais la quasi totalité des systèmes de traitement automatique des langues naturelles utilisent des représentations atomiques plutôt qu'ensemblistes. Afin d'autoriser PELEAS à coopérer avec ces systèmes, il faut introduire un moyen d'extraire (si besoin est) une représentation atomique des configurations de potientels sémantiques produites par PELEAS. De façon intuitive, nous choisissons d'ajouter aux sorties de PELEAS un critère de choix (en fait une projection des configurations sur un ensemble de représentations conceptuelles atomiques) fondé sur une estimation du taux de participation des sommets descripteurs dans la construction de l'interprétation en contexte de leur entrée lexicale.

Il s'agit d'une estimation sommaire du taux de participation d'une vue conceptuelle dans la construction du sens en contexte de l'entrée lexicale (son "importance"). Cette estimation est un réel dans $[0; 1]$ qui est calculé en se basant sur le principe suivant : plus un sommet est situé profondément dans le potentiel sémantique et plus il correspond à des informations précises sur le sens de l'entrée lexicale. Par conséquent, plus une vue conceptuelle possède de traits fortement activés, plus on a trouvé dans le contexte d'indications se rapportant à cette vue particulière, donc plus elle a de chances de participer fortement au sens de cette occurrence de l'entrée lexicale.

Nous définissons donc l'estimation de la participation d'une vue v par :

$$\widehat{v} = \begin{cases} \dfrac{|\{v.t/\ \mathrm{val}(v.t) = \textit{salient}\}| + |\{v.t/\ \mathrm{val}(v.t) = \textit{negated}\}|}{|\{v.t\}|} & \text{si } |\{v.t\}| \neq 0 \\ 0 & \text{si } |\{v.t\}| = 0 \end{cases}$$

où $\{v.t\}$ est l'ensemble des traits de la vue conceptuelle v.

L'estimateur $\widehat{v}$ est une mesure *relative*, c'est-à-dire qu'on peut interpréter $\widehat{v_1} < \widehat{v_2}$ comme "la participation de v_2 est sans doute plus importante que la participation de v_1". Cependant $\widehat{v} = 0$ ne signifie pas que v ne participe pas à la construction du sens de l'entrée lexicale, mais seulement que l'on ne dispose pas d'information identifiant précisément la participation de v (la saillance de v est supputée ou inférée).

4.4 Construction d'un lexique

La construction d'un lexique adapté au modèle PELEAS, i.e. composé de potentiels sémantiques, est une tâche qui demande un travail important d'expertise linguistique et terminologique. On peut s'appuyer, pour effectuer ce travail, sur des dictionnaires détaillés, possédant des étiquettes de domaines d'activité et de champs conceptuels, ainsi que des exemples d'usage. Cependant, des dictionnaires aussi détaillés ne sont pas facilement disponibles pour le français sous forme électronique. Et même si tel n'était pas le cas, ils représenteraient une abstraction de la langue figée à un moment donné, alors que l'esprit de notre modèle est d'utiliser un raisonnement basé sur des cas réels, fidèles reflets d'un contexte socio-culturel.

Pour respecter simultanément cette exigence de fidélité et une volonté d'accélérer la phase de construction d'un lexique, nous avons envisagé la conception une méthode d'extraction automatique de données pouvant être réutilisées directement dans le potentiel sémantique d'une entrée lexicale. La conception et le développement d'un tel système constituent en eux-mêmes un travail de recherche à part entière, qui n'a été qu'effleuré ici et demande à être poursuivi ultérieurement. Nous organiserons donc cette section comme un rapport d'étude préliminaire : quelles devraient être les fonctionnalités précises

d'un tel logiciel (§4.4.2) ? Comment les réaliser (§4.4.3) ? On trouvera des questions d'ordre plus pratique au chapitre suivant : quelles hypothèses ont été retenues pour la mise en œuvre (§5.5.1) ? et enfin quel prototype a été effectivement réalisé (§5.5.2) ?

4.4.1 Pôles sémantiques

L'algorithme de propagation de l'activité de PELEAS (cf. §4.3.3) calcule en fait des interprétations composées d'usages attestés. Ce sont ces usages qui figurent dans le potentiel sémantique d'une entrée lexicale, et nous les considérons comme des *pôles sémantiques*.

Nous appelons pôle sémantique une nuance d'interprétation attestée par l'usage, depuis des données contextuelles très générales (contextuelles) jusqu'aux données les plus spécifiques. Formellement, un pôle sémantique se repère dans le potentiel sémantique d'une entrée lexicale comme un *chemin* de l'arbre n-aire sous-jacent, c'est-à-dire comme une suite *domaine.notion.vue.trait* (ou *domaine.notion.vue* si la vue conceptuelle en question ne possède pas de trait).

Une interprétation calculée par PELEAS est en fait un ensemble de tels chemins pondérés par des taux d'activité relatifs. Cela signifie que l'on peut considérer l'ensemble des chemins d'un potentiel sémantique comme une base de vecteurs indépendants, le potentiel sémantique lui-même étant l'espace engendré par les chemins. Les sorties de PELEAS forment donc des coordonnées vectorielles dans cet espace, ce qui permet de prendre en compte des interprétations non présentes dans le graphe pyramidal, qui est la représentation *minimale* de l'espace (potentiel) sémantique d'une entrée lexicale.

Cependant, l'indépendance vectorielle est rarement réalisée sur tous les chemins considérés deux à deux. Pour s'en rapprocher, on s'attachera, lors de la construction du potentiel sémantique, à maximiser le contraste entre les différents descripteurs d'un même sommet. Cela signifie qu'à chaque niveau de spécificité supérieure, on veillera à ce que les sommets de ce niveau qui sont tous reliés entre eux, lexicalisent les usages les plus différents possible les uns des autres.

4.4.2 Fonctionnalités souhaitées

L'idée d'extraire des données destinées à alimenter la construction d'entrées PELEAS provient de ce que nous avons dit au §4.4.1 : chaque chemin descendant à l'intérieur du potentiel sémantique d'une entrée correspond à un *usage attesté* en contexte. Or un locuteur humain ne peut se constituer une telle base d'usages attestés que par l'intermédiaire de son expérience : dans sa pratique quotidienne de la langue orale ou écrite, un nouvel usage d'un mot ou d'une expression se présente à lui. Sans vouloir entrer ici dans les méandres d'une discussion sur l'acquisition du langage, nous supposerons que ce nouvel usage est intégré lorsque le locuteur reçoit une *explicitation* de cet usage (issue d'un autre locuteur ou inférée de connaissances antérieures).

En somme, un humain se constitue une base d'usages attestés en observant perpétuellement le corpus langagier oral et écrit qui est constitué de ses expériences des langues. Il semble donc judicieux de chercher à reproduire ce processus sur machine afin d'automatiser la constitution des lexiques PELEAS.

Le potentiel sémantique d'une entrée lexicale se compose de trois ensembles d'objets de natures différentes :

1. les sommets descriptifs, de nature linguistique, dont les étiquettes balisent les parcours interprétatifs du potentiel sémantique de l'entrée lexicale ;

2. les conditions contextuelles, de nature pragmatique, qui décrivent l'influence du contexte d'énonciation sur les préférences entre les différents parcours interprétatifs ;

3. et les contraintes sémantiques, de nature sémantique, qui expriment des contraintes de proximité et de tolérance entre les parcours interprétatifs.

Par conséquent, un système permettant de simuler la constitution d'une base d'usages attestés devrait procéder à trois types d'acquisition de données :

1. *Extraction des sommets descriptifs* : le système devrait pouvoir extraire d'un corpus suffisamment large des expressions linguistiques correspondant aux quatre types d'informations encodées dans les sommets descriptifs de PELEAS : des étiquettes de domaines d'activité, des étiquettes de notions, des étiquettes de vues conceptuelles (i.e. des aspects paraphrastiques de l'entrée lexicale considérée), et des étiquettes de traits (ou informations co-occurrentes de l'entrée ou de ses paraphrases).

2. *Extraction des conditions contextuelles* : il s'agirait plutôt de déduire ces conditions que de les extraire, dans une phase venant après l'extraction des sommets descriptifs. Il surgirait alors deux problèmes :

   — Pour identifier les prémisses des conditions, il faudrait disposer d'une grille de caractéristiques pertinentes valable pour tout texte (ou plusieurs grilles valables pour des classes de textes assez larges), qui permettrait d'obtenir à un moment donné une mesure fidèle de l'état du texte. On est donc ramené à un problème de pertinence qui défie justement les rationnalisations prédictives.

   — Constituer une condition contextuelle revient à mettre en rapport des prémisses (une mesure pertinente du contexte) avec une signification balisée, c'est-à-dire une configuration du potentiel sémantique de l'entrée lexicale. Mais dans la définition que nous avons donnée du modèle PELEAS, les conditions contextuelles servent justement à amorcer le calcul de la configuration du potentiel sémantique. C'est un problème infiniment récursif à moins d'imaginer une méthode itérative par auto-amorçage, qui réintroduirait donc une intervention humaine sous forme d'expertise linguistique.

3. *Extraction des contraintes sémantiques* : disposant de configurations balisées du potentiel sémantique, il deviendrait alors possible d'envisager une mesure des tolérances réciproques de ces différentes configurations (notamment par observation des co-occurrences). Mais ceci demanderait d'aller puiser des informations dans une base de connaissances encyclopédiques (l'équivalent du fonds culturel du locuteur), ce qui nous ramène, outre les difficultés pour rendre compatibles ces informations avec celles manipulées par PELEAS, à un problème de pertinence également : quelles informations encyclopédiques exactement faut-il sélectionner ?

Du fait des difficultés énoncées dans les deux derniers points, et du temps imparti à un travail de thèse, nous avons choisi de nous intéresser ici uniquement au premier point.

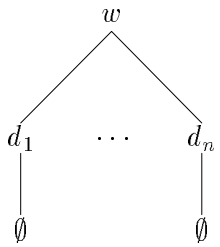
4.4.3 Modélisation

L'algorithme que nous décrivons ici vise à extraire d'un corpus de référence suffisamment large des mots ou expressions qui serviront d'étiquettes pour les sommets descriptifs d'une entrée lexicale.

Soit le mot w donné, dont on cherche à construire le potentiel sémantique par extraction de données sur un corpus $\mathcal{C}$:

1. Dans la mesure du possible, on essayera de définir une partition de $\mathcal{C}$ en sous-corpus $\mathcal{C} = \mathcal{C}_1 \oplus \ldots \oplus \mathcal{C}_n$ où chaque $\mathcal{C}_i$ relève d'un domaine d'activité identifié et étiqueté d_i. S'il existe certains textes de $\mathcal{C}$ pour lesquels on ne peut identifier de domaine, on pourra créer un domaine fantôme (étiqueté $\emptyset$).

   A partir de là, on peut déclarer les d_i comme domaines de w dans son potentiel sémantique.



   Les schémas que nous fournissons pour illustrer cet algorithme sont des projections planes arborescentes du potentiel sémantique de w.

2. Prenons maintenant un sous-corpus $\mathcal{C}_i$. On va tout d'abord rechercher les *équivalents en contexte* de w. Il s'agit en fait de retrouver dans $\mathcal{C}_i$ tous les mots w' dont les occurrences ne sont pas indépendantes des occurrences de w. Ce processus de recherche des équivalents est décrit par Ferrari et Prince [34] comme étant un défaut de la méthode dite de l'*information mutuelle* (une mesure classique de non indépendance des occurrences) utilisée par Church [22] et définie comme suit : soient a et b deux termes dont on veut juger de la corrélation en calculant leur information mutuelle. Celle-ci est établie entre a et b dans un corpus C par

$$I(a,b) = \log_2 \frac{P_C(a,b)}{P_C(a)P_C(b)}$$

   où $P_C(a)$ est la probabilité de trouver a dans le corpus C, $P_C(b)$ est la probabilité de trouver b et $P_C(a,b)$ est la probabilité d'avoir a si b.

   La critique de cette méthode mentionnée par Ferrari et Prince porte sur le fait que l'information mutuelle « dénote une propriété de forte corrélation, mais elle n'indique pas de quelle nature est cette relation [...] ». Cette lacune (considérée dans le contexte de l'utilisation qu'on font les auteurs) est en fait un atout pour nous. En effet, la méthode de l'information mutuelle permet de faire ressortir l'ensemble des termes b régulièrement co-occurrents de a, mais sans avoir besoin de distinguer s'il s'agit de synonymes de a dont la co-utilisation dans une relation anaphorique permet d'éviter la répétition de a, ou bien s'il s'agit de spécialisations (*differentiae*), d'antonymes ou de descripteurs thématiques dont la co-occurrence a un but d'explicitation. Dans le cadre du modèle PELEAS, nous ne nous intéressons qu'à des relations du type *est-décrit-par*, c'est-à-dire volontairement indifférenciées.

   Cependant, outre le fait que l'information mutuelle est abondamment critiquée du point de vue de sa significativité, c'est une mesure qui est plus particulièrement adaptée à la détection de *bi-grammes* dépendants, i.e. des suites orientées de deux mots. Or notre critère de pertinence est fortement lié au phénomène d'isotopie locale (cf. §2.5.2.4), donc dans notre cas, la co-occurrence

de deux mots ne s'exprime plus en termes de bigrammes, mais comme occurrence simultanée de deux mots à l'intérieur d'une fenêtre où la récurrence des isotopies locales reste forte, sans souci d'orientation (les deux mots peuvent apparaître dans n'importe quel ordre). Nous nous sommes donc tournés vers une autre méthode de calcul de non indépendance : le *test de l'Ecart Réduit* (cf. Bonhomme [8]). Cette méthode, ainsi que les hypothèses que nous avons utilisées pour l'exploiter, sont présentées de façon plus détaillée au §5.5.1.

Pour cette première étape de l'algorithme d'extraction, nous allons nous servir du score d'écart réduit pour construire une liste d'équivalents de w dans $\mathcal{C}_i$: $E = \{e_1, \ldots, e_n\}$. L'hypothèse utilisée ici est la suivante : si les co-occurrences de e_j et w ont un fort score d'écart réduit positif, alors e_j a de bonnes chances d'être une paraphrase ou un terme explicatif de w (hypothèse déduite des observations présentées par Prince [85] sur la *récurrence sémantique*).

3. Pour le domaine d_i du corpus $\mathcal{C}_i$, on dispose donc d'un ensemble de termes $E = \{e_j\}$ sur lequel on va appliquer le processus itératif suivant, qui suppose que l'on dispose d'une base de relations lexicales sémantiques (comme décrites par Evens *et al.* [31] par exemple) :

   — Si $E = \emptyset$, on arrête l'itération.

   — Soient e_j et $e_{j'}$ deux éléments différents de E. Si e_j et $e_{j'}$ sont répertoriés dans la base de relations de telle façon que $e_{j'}$ ***membre-de*** e_j, ou $e_{j'}$ ***partie-de*** e_j ou $e_{j'}$ ***attribut-de*** e_j (trois cas que nous résumerons par la notation $e_{j'} \subseteq e_j$), alors on place e_j dans la structure descriptive de w comme vue conceptuelle de la notion $d_i.\emptyset$, et $e_{j'}$ comme trait de la vue $d_i.\emptyset.e_j$, puis on supprime e_j et $e_{j'}$ de E, et on relance l'itération (cf. figure 4.7).

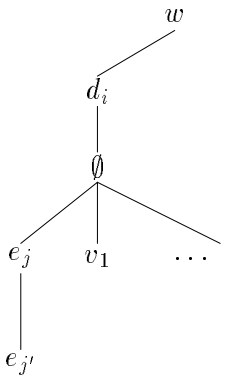


FIG. 4.7: Sous-classement d'un trait sémantique

   — S'il n'est pas possible de trouver un tel couple $(e_j, e_{j'})$, alors soit $e_k \in E$. S'il existe une vue conceptuelle $d_i.\emptyset.v$ telle que $e_k \subseteq v$, alors on place e_k comme trait de v. Si au contraire on a $v \subseteq e_k$ alors on remplace v par e_k, et v devient un trait de e_k. Dans les deux cas, on supprime e_k de E, et on relance l'itération (cf. figure 4.8).

   — Lorsque E est réduit à un ensemble de termes qu'on ne peut mettre en relation ni entre eux, ni avec les vues de de $d_i.\emptyset$, alors chacun de ces termes devient une nouvelle vue de $d_i.\emptyset$, et $E = \emptyset$.

4. Supposons maintenant que la base de relations lexicales sémantiques inclue une structure de thésaurus, c'est-à-dire qu'elle permette d'organiser les termes par la relation hyperonymie / hyponymie. On notera $a \leq b$ si b est un hyperonyme de a ; la relation $\leq$ définit l'ordre partiel usuel sur les termes du thésaurus.

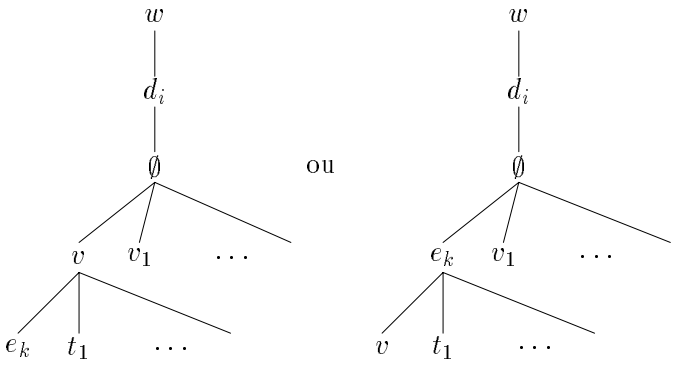


FIG. 4.8: Spécialisation d'un trait sémantique

Grâce à la relation d'ordre induite par $\leq$, on peut associer à toute partie $p \in \mathcal{P}(V_{d_i})$ de l'ensemble des vues du domaine d_i un unique plus petit commun hyperonyme.

On cherche ici à découper V_{d_i} en groupes de vues étiqueté chacun par une notion différente et pertinente. Cela revient à trouver une partition de V_{d_i} dont chaque partie est un ensemble de vues superordonnées par un hyperonyme non trivial, i.e. dont le niveau d'abstraction est suffisamment bas pour que les différentes vues soient sémantiquement reliées de façon pertinente (un hyperonyme commun comme $\top$, ou *objet_abstrait* serait un "mauvais hyperonyme", dans le sens où la partie de V_{d_i} correspondante serait un "fourre-tout" sémantique).

Supposons la partition suivante : $V_{d_i} = V_{d_i}^1 \oplus \ldots \oplus V_{d_i}^k$, chaque partie $V_{d_i}^t$ étant munie d'un hyperonyme commun h_t (i.e. $\forall v \in V_{d_i}^t, v \leq h_t$). Notons $z = \left(\langle V_{d_i}^1, h_1\rangle, \ldots, \langle V_{d_i}^k, h_k\rangle\right)$ cette configuration (i.e. la partition de V_{d_i} munie de ces hyperonymes). On notera de plus $N_t(z) = |V_{d_i}^t|$ le nombre d'éléments de la partie $V_{d_i}^t$ et $H_t(z) = \mathrm{Prof}(h_t)$ la profondeur de l'hyperonyme h_t dans le thésaurus (en posant par convention $\mathrm{Prof}(\top) = 0$).

Nous pouvons alors définir la fonction $F(z)$ qui évalue à la fois la taille moyenne des groupes de vues et le niveau d'abstraction de leurs hyperonymes (i.e. la pertinence des notions)

$$F(z) = \sum_{t=1}^{k} \left(\frac{N_t(z)}{k}\right) + H_t(z)$$

Une "bonne" partition de V_{d_i} serait alors une configuration z qui maximiserait la fonction F, ce qui nous permettrait de trouver les notions les plus générales, mais les moins abstraites possibles. Cela revient à trouver une solution optimale au programme linéaire suivant :

$$\left|\begin{array}{l} \mathrm{Max}\, F(z) \\ \forall t, N_t(z) \geq 1 \\ \forall t, \forall v \in V_{d_i}^t, v \leq h_t \end{array}\right. \tag{4.1}$$

Dans ce cas, les h_t peuvent être identifiés aux notions du domaine d_i et venir remplacer l'unique notion $d_i.\emptyset$ dont nous disposions jusqu'à présent et qui aura été préalablement supprimée. En effet, si dans la solution l'un des $h_t = \top$, il faudra réintroduire une nouvelle notion $d_i.\emptyset$ pour le seul groupe $V_{d_i}^t$ correspondant.

Trouver cette solution n'est pas une tâche triviale, et une heuristique permettant de faciliter la résolution pourrait être d'utiliser les *regroupements sémantiques* d'un dictionnaire hiérarchique.

> Dans un tel outil, les entrées lexicales polysémiques sont subdivisées de façon arborescente et récursive en sous-entrées. Or, on constate généralement que les subdivisions de premier niveau peuvent correspondre à des notions du modèle PELEAS. Il conviendrait alors de pouvoir faire figurer ces subdivisions parmi les contraintes du programme 4.1.

La mise en œuvre de cette méthode d'extraction automatique est constituée du logiciel BARD que nous décrivons au §5.5 avec les hypothèses particulières que nous avons faites pour l'application du test d'écart réduit.

4.5 Récapitulation

A partir des observations linguistiques et pratiques que nous avons présentées aux chapitres précédent, nous avons conçu un modèle de représentation du lexique particulièrement adapté au traitement des phénomènes de polysémie complexe : polysémie d'acception *et* polysémie d'usage.

Ce modèle, PELEAS, est un modèle purement dénotatif : les éléments qu'il manipule ne supposent aucune mise en référence par la machine à des objets du monde réel. Il se base entièrement sur les principes sous-tendant la sémantique différentielle, dans le sens où la représentation d'un signifié est un *système* dans lequel les éléments s'interdéfinissent. Plus particulièrement, seule la structure d'une représentation est importante pour le fonctionnement du modèle. Les étiquettes lexicales des différents éléments (sommets) balisent des récurrences et co-occurences lexicales attestées autour d'emplois ou d'acceptions socialement reconnus, mais ne jouent aucun rôle dans le processus de calcul des interprétations.

PELEAS est un modèle hybride entre la classique représentation symbolique et les réseaux connexionnistes puisqu'il implique à la fois des notations symboliques centralisées et un processus dynamique de calcul des interprétations par propagation de taux d'activité. Il s'agit en outre d'un *lexique dynamique* associant une partie statique minimale et une partie dynamique. Tout comme le GENERATIVE LEXICON, PELEAS s'apparente à un système de propagations de contraintes avec possibilité de relaxation, et tout comme EDGAR, dont il s'inspire fortement, il ne fait aucune différence entre sens littéral et sens métaphorique ou figuré.

La particularité de ce modèle à laquelle nous sommes le plus attaché est qu'il *assume sa dénotativité*. Nous voulons dire par là qu'il n'y a aucune tentative de manipuler des représentations sémantiques telles qu'on les conçoit habituellement : le modèle PELEAS ne tente absolument pas de simuler la compréhension des unités lexicales qu'il traite. Il se contente de manipuler des dénotations au travers de lexicalisations : les signifiés n'apparaissent même pas directement, on ne manipule que des co-occurrences de lexicalisations et des contraintes de tolérance entre ces occurrences.

Nous tenons en effet à rester dans un cadre non anthropomorphique : un système informatique *ne comprend pas* les données qu'il manipule, pas même en jouant sur la définition du verbe *comprendre*. C'est pour cela que les étiquettes des sommets sont en fait non informatives dans PELEAS : il s'agit uniquement de repères pour l'opérateur humain qui est seul à pouvoir injecter ou extraire du sens dans les structures manipulées par la machine.

Les raisonnements de PELEAS se situent donc uniquement dans un cadre différentiel et non référentiel : seule la structure d'un potentiel sémantique est importante (les arcs, et pas les sommets).

De plus, le modèle PELEAS ne prévoit pas d'enregistrement des configurations qu'il calcule dans une base de connaissances. En effet, un de ses buts premiers est la tolérance aux usages nouveaux ou

aux combinaisons originales d'usages connus. Epistémologiquement, seuls doivent figurer dans sa base de connaissances les usages attestés qui sont socialement balisés. Une configuration particulière a de grandes chances de constituer un "usage limite" ponctuel ou appartenant à un microlecte, donc sans reconnaissance sociale.

4.6 Expérimentation du modèle et de sa mise en œuvre

Afin de tester l'efficacité du modèle PELEAS, nous avons décidé de procéder à une première évaluation des réponses fournies par sa mise en œuvre (cf. §5) pour un lexique comportant 21 entrées lexicales testées sur un corpus de 110 phrases.

Les entrées de ce lexique se répartissent en trois catégories :

— 16 noms communs : *bouche*, *fille*, *fils*, *frère*, *langue*, *maison*, *mère*, *mort*, *nez*, *peau*, *père*, *roi*, *soeur*, *soleil*, *vie*, *système* ;

— 3 verbes : *manger*, *toucher*, *voir* ;

— 2 adjectifs : *jeune*, *vieux*.

Ces mots ont été retenus en fonction des deux critères suivants :

1. ils doivent correspondre à des *mots courants* (cf. Prince [86]), i.e. ils désignent des parties du corps (*bouche*, *langue*, *nez*, *peau*), des relations anthropologiquement importantes (*père*, *mère*, *fils*, *fille*, *frère*, *sœur*, *roi*), des processus vitaux (*vie*, *mort*, *manger*, *toucher*, *voir*) ou des états vitaux (*jeune*, *vieux*). Le cas de *système* est à part : il n'est pas courant dans la langue quotidienne mais dans divers langages de spécialités (c'est un cas de mot joker) ;
2. ils doivent se décliner selon plusieurs axes conceptuels (ce que nous appelons les *notions*), c'est-à-dire que leur définition dans un dictionnaire hiérarchique (du type Larousse ou Robert) doit contenir plusieurs subdivisions de premier niveau. De plus, ils doivent contenir des usages de spécialité, afin de tester notre définition étendue des mots courants.

Les structures descriptives de ces entrées lexicales figurent en annexe (cf. §A.1). Elles ont été construites en utilisant deux outils :

1. leur définition dans le *Petit Larousse 1989* (qui est structuré comme un dictionnaire hiérarchique) ;
2. les résultats statistiques de corrélation des co-occurrences de ces entrées lexicales avec d'autres mots accompagnants, résultats fournis par le logiciel BARD qui est présenté au §5.5. Ces statistiques ont été produites en explorant un corpus formé, pour chaque entrée lexicale, de tous les articles du *Monde 1994 sur CD-ROM* contenant cette entrée.

Le corpus des phrases sur lesquelles ces structures descriptives ont été testées est donné en annexe (cf. §A.2). Il comporte 110 phrases comportant des difficultés diverses : de l'interprétation monosémique à la polysémie d'usage (les difficultés d'interprétations soulevées par ces phrases figurent à la même

section d'annexe que le corpus). Les phrases contenant des phénomènes de polysémie d'acception (marqués d'une astérisque dans l'annexe) ont toutes été testées avec succès avec le modèle EDGAR. Elles constituent donc un test de non-régression de PELEAS vis-à-vis d'EDGAR.

Les difficultés d'interprétation soulevées par ces phrases se répartissent en 11 catégories qui sont susceptibles de se combiner mutuellement :

— absence d'indice contextuel (sous-détermination de l'information) ;

— présence d'un emploi idiomatique qui prend le pas sur l'emploi le plus courant ;

— neutralisation d'un emploi idiomatique (en général suite à une isotopie qui virtualise les sèmes caractéristiques de cet emploi) ;

— présence d'un emploi catachrétique (qui ne respecte pas les conditions de restriction de la sélection) ;

— présence d'une analogie (transfert de sèmes entre deux champs sémantiques attestés) ;

— présence d'une généralisation (actualisation de sèmes inhérents à un taxème attesté) ;

— superposition de deux emplois ou de deux acceptions ;

— collision (entraînant une dissimilation) entre deux emplois ou deux acceptions ;

— dissimilation d'emplois au sein d'une structure tautologique (lorsque l'entrée lexicale apparaît deux fois dans la phrase) ;

— oscillation entre deux emplois ou deux acceptions (i.e. l'un des deux risque d'être inadéquat mais il n'est pas possible de décider duquel il s'agit) ;

— changement de catégorie syntaxique.

Ces phrases ont été soit construites pour l'occasion, soit proviennent de romans, de titres de journaux ou de poésies. Elles sont presque toutes susceptibles de mettre en échec un système d'analyse sémantique basé sur un raisonnement par restriction de la sélection cherchant à conserver une unique interprétation aux dépends de toutes les autres.

Nous avons décidé de constituer un tel corpus *ad hoc* parce que l'originalité de PELEAS réside dans sa capacité à gérer des phrases réputées difficiles, c'est-à-dire présentant des contextes locaux où l'information est dégradée, "brouillée" ou inattendue. Dans le cas d'une phrase sans difficulté (c'est-à-dire où les emplois de tous les mots correspondent aux attentes syntaxiques et sémantiques de façon univoques), l'ensemble des règles contextuelles d'une entrée lexicale se comporte tout naturellement comme un test sélectif classique. Ceci découle directement du fait que les conditions contextuelles de PELEAS n'offrent aucune différence formelle avec un faisceau de conditions nécessaires et suffisantes utilisé pour pratiquer une restriction de la sélection. Dans ce genre de situation, PELEAS n'offre donc aucun avantage par rapport à un système de résolution d'ambiguïtés lexicales classique : on obtient une unique vue conceptuelle sélectionnée.

Par conséquent, il ne nous a pas semblé intéressant de procéder à une évaluation de PELEAS sur un corpus de grande taille constitué de phrases non sélectionnées. Un tel corpus contiendrait en effet une majorité de phrases interprétables uniquement par restriction de la sélection, ce qui ne consitue pas un test significatif des apports essentiels de PELEAS : sa robustesse face aux informations de mauvaises qualités ou pléthoriques, et sa capacité d'interprétation ensembliste.

4.6.1 Objectifs de l'expérimentation

Les questions auxquelles nous cherchons une réponse dans cette expérimentation concernent aussi bien le modèle PELEAS lui-même que sa mise en œuvre. Nous les présentons toutes ici afin d'éviter au lecteur non informaticien la lecture du §5 qui n'est pas strictement nécessaire à la compréhension globale de notre travail.

Ces questions sont les suivantes :

— **Le modèle PELEAS est-il robuste ?** En effet, le défaut majeur des analyseurs sémantiques classiques est leur mise en échec par la polysémie d'usage notamment : lorsque les contraintes de restriction de la sélection ou bien les contraintes d'intégrité de leur base de connaissances (suivant les modèles mis en œuvre) ne sont pas respectées, ils sont incapables de fournir une réponse, ce qui peut conduire à un arrêt du processus global de traitement de l'énoncé. Nous voulons savoir ici si notre mise en œuvre de PELEAS supporte les situations dégradées (manque d'information ou contradiction des indices textuels) et reste capable de fournir une interprétation.

— **Le modèle PELEAS est-il puissant ?** Dans quelle mesure les réponses que nous obtenons sont-elles pertinentes ? Obtient-on des interprétations farfelues ? Y a-t-il des cas où certaines interprétations importantes sont omises dans la réponse ?

— **La mise en œuvre est-elle efficace ?** Combien de temps faut-il pour construire une structure descriptive ? Quelle place occupe le lexique ? Combien de temps faut-il pour traiter une entrée lexicale ?

— **La mise en œuvre est-elle coopérative ?** Peut-elle s'insérer facilement dans une architecture coopérative de traitement automatique des langues naturelles ? A quelles conditions ?

— **Comment construire une "bonne" structure descriptive ?** Y a-t-il des critères ou des conseils à suivre pour construire une structure descriptive ? Quel rôle jouent les résultats statistiques de co-occurrence lexicale dans la construction d'une structure descriptive ?

4.6.2 Caractéristiques et mesures statistiques

Nous souhaitons pouvoir évaluer quantitativement une partie du comportement du modèle PELEAS. Cette évaluation quantitative est menée par un certain nombre de mesures statistiques sur la mise en œuvre que nous en avons fait (cf. §5).

Nous essayons de couvrir tous les niveaux de la mise en œuvre : depuis la phase de construction du lexique (§4.6.2.1 ; acquisiton statistique sur corpus suivant la méthodologie définie au §4.4), en passant par des mesures sur les structures descriptives (§4.6.2.2), jusqu'aux mesures concernant le processus d'évaluation du sens en contexte et ses résultats (§4.6.2.4).

4.6.2.1 Acquisition statistique des données

Les informations utilisées pour construire la structure descriptive d'une entrée lexicale proviennent d'une part de sa définition dans un dictionnaire hiérarchique, et d'autre part des sorties du logiciel BARD[1] (cf. §5.5).

[1] Rappelons que ces sorties indiquent, pour un corpus donné et un mot-foyer donné, quels mots ont une fréquence significative de co-occurrence avec le mot-foyer dans une fenêtre centrée d'au plus 3 phrases.

Pour construire les structures descriptives de notre lexique de test, nous avons pris les résultats fournis par BARD sur des corpora issus du *Monde 1994 sur CD-ROM*. Pour chaque entrée lexicale, le corpus en entrée de BARD était constitué de l'ensemble des articles contenant l'entrée lexicale (jouant le rôle de mot-foyer).

Le tableau 4.1 donne une description statistique de la taille T des corpora fournis en entrée de BARD, et du temps H nécessaire pour effectuer l'analyse statistique des co-occurrences.

	T (Mo)	H (h)
Moyenne	$8,4$	$4,6$
Ecart-type	$1,2$	$0,7$

TAB. 4.1: Description statistique des corpus pour BARD

Ces temps sont assez importants. En fait, l'efficacité de BARD est largement grévée par le processus de lemmatisation qui est appliqué à pratiquement chaque mot du corpus (cf. §5.6).

4.6.2.2 Description statique du lexique

Le tableau 4.2 présente une description statistique des mesures du lexique utilisé (en termes de composants statiques) :

— le temps total H requis pour construire une structure descriptive (i.e. le temps pris par BARD pour extraire des données du corpus adéquat, plus le temps nécessaire pour intégrer les informations du dictionnaire grâce au logiciel MELISANDE) ;

— la taille F d'un fichier contenant une structure descriptive (ce qui permet de mesurer l'occupation sur disque du lexique) ;

— le nombre N de sommets dans une structure descriptive ;

— le nombre de contraintes sémantiques C dans une structure ;

— le nombre de conditions contextuelles R dans une structure (ces trois dernières mesures permettent d'avoir une idée de la complexité moyenne d'une structure descriptive).

	H (h)	F (Ko)	N (sommets)	C (contraintes)	R (règles)
Moyenne	$5,94$	$2,88$	$33,95$	$6,20$	$16,85$
Ecart-type	$0,61$	$0,99$	$9,71$	$3,38$	$6,27$
Minimum	$4,83$	$1,43$	20	1	8
Maximum	$6,58$	$4,78$	55	14	28

TAB. 4.2: Mesures statiques du lexique

La phase de codage des informations issues du dictionnaire ne représente donc en moyenne qu'un peu plus d'une heure pour chaque entrée lexicale (en calculant $H - T$). Il est donc assez facile de traduire les informations du dictionnaire dans le format informatique propre à PELEAS.

Les mesures du nombre de sommets, de contraintes sémantiques et de règles contextuels montrent bien que les structures descriptives sont en moyenne riches et complexes, ce qui justifie notre choix de

sélectionner les entrées lexicales à faire figurer dans les lexiques de PELEAS. Parallèlement, l'occupation moyenne sur disque d'une entrée lexicale est relativement réduite (moins de 3 Ko) en regard de sa richesse descriptive.

4.6.2.3 Principe du test

Le test est effectué par un petit programme (écrit en C++) qui pilote le moteur d'inférences LIGHTPELEAS (cf. §5.2). Ce programme de test établit une relation client-serveur avec LIGHTPELEAS, auprès duquel il tient lieu d'interface pour récupérer des résultats lisibles.

Le principe est de simuler le fonctionnement de LIGHTPELEAS à l'intérieur d'une architecture générale de traitement automatique des langues naturelles. C'est pourquoi les prédicats composant les conditions contextuelles des diverses entrées du lexique (cf. §A.1) ne testent que des informations susceptibles d'être produites par des modules classiques : analyseur morphologique, analyseur syntaxique, treillis de types conceptuels, gestionnaire de thèmes et gestionnaire de références.

Pour simuler l'existence de cette architecture, le programme de test prend en entrée un fichier contenant la phrase à tester, le mot foyer de l'ambiguïté, ainsi qu'une évaluation (réalisée par un opérateur humain dans notre cas) de tous les indices textuels susceptibles d'être consultés pour l'évaluation de la structure descriptive correspondante. Il s'agit en fait des valeurs prises, pour cette occurrence de l'entrée lexicale, par tous les prédicats figurant dans les conditions contextuelles de la structure descriptive.

Ces valeurs sont ensuite inscrites dans la Base de registre (cf. p. 340) du système d'exploitation, où elles seront consultées par les bibliothèques de prédicats (cf. §5.1.2.1 et §5.4) attachées au lexique. Le recours à la Base de registre, qui est le dépositaire privilégié des informations partagées au sein du système d'exploitation, est une mise en œuvre comme une autre du principe de tableau noir généralement utilisé dans les architectures de traitement automatique des langues naturelles.

4.6.2.4 Mesures dynamiques

Le tableau 4.3 donne une description statistique du nombre de propositions d'interprétation r (vues conceptuelles saillantes ou niées) fournies par LIGHTPELEAS, ainsi que le temps T (en ms) mis pour calculer une configuration du potentiel sémantique.

	r (propositions)	T (ms)
Moyenne	$3,29$	$413,00$
Ecart-type	$1,65$	$199,51$
Minimum	1	170
Maximum	10	931

TAB. 4.3: Interprétation d'une entrée lexicale

Les temps de réponse de LIGHTPELEAS ont été mesurés sur un Pentium 100 Mhz, doté de 48 Mo de mémoire vive, et fonctionnant avec le système WINDOWS NT4.

Chaque interprétation en contexte d'une occurrence d'une entrée lexicale contient en moyenne 3 vues conceptuelles, pour des informations linguistiques figurant dans un dictionnaire commun. Cela prouve la nécessité d'un traitement ensembliste : la sélection de la "meilleure proposition" aurait en moyenne conduit à une perte de deux-tiers des informations sémantiques.

Comme on pouvait s'en douter, et comme le prouve n'importe quel outil de profilage de l'exécution, les temps mesurés ici sont en fait fonction du nombre de prédicats contenus dans les conditions contextuelles d'une entrée lexicale (et accessoirement du nombre de bibliothèques de prédicats), puisque la majeure partie de ces temps est consacrée à l'édition de liens dynamiques (cf. annexe C) pour pouvoir évaluer les prédicats qui figurent en partie gauche des règles contextuelles. Ces temps sont néanmoins satisfaisants : la machine de test est lente, et le test induit volontairement plusieurs niveaux d'indirection système (simulations d'appel de processus distants enchaînées) afin de tester l'aspect distribué de la mise en œuvre.

4.6.2.5 Test de corrélation

Nous voudrions maintenant savoir si les variables N, C, R, et r sont corrélées deux-à-deux, c'est-à-dire s'il est possible d'en déduire une en fonction d'une autre. Auquel cas, l'une de ces variables serait *endogène* et donc prédictible. Par voie de conséquence, cela sous-entendrait qu'il est possible de concevoir un modèle statique de la polysémie d'usage/d'acception capable de prédire l'interprétation d'un mot sans recourir à un processus dynamique.

Pour vérifier les relations qu'entretiennent ces quatre variables, nous allons calculer leur *taux de corrélation* deux-à-deux. Le taux de corrélation $\rho_{X,Y}$ de deux variables aléatoires $X \rightsquigarrow \mathcal{N}(\mu_X; \sigma_X)$ et $Y \rightsquigarrow \mathcal{N}(\mu_Y; \sigma_Y)$ est défini par

$$\rho_{X,Y} = \frac{\mathrm{Cov}(X,Y)}{\sigma_X . \sigma_Y}$$

où, si n est la taille des échantillons,

$$\mathrm{Cov}(X,Y) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_X)(y_i - \mu_Y)$$

Ce coefficient est compris entre -1 (variables anticorrélées) et 1 (variables corrélées), 0 représentant l'indépendance.

Le tableau 4.4 donne les coefficients de corrélation de ces quatre variables.

	N	C	R	r
N	1,000000000			
C	0,399331931	1,000000000		
R	0,878043527	0,443529074	1,000000000	
r	-0,153796384	0,51170761	-0,117989878	1,000000000

TAB. 4.4: Coefficients de corrélation de N, C, R et r

Le seul coefficient de corrélation qui soit significatif est $\rho_{N,R} = 0,878043527$. Cela signifie que le nombre de sommets descriptifs et le nombre de conditions contextuelles varient de manière semblable. Ce n'est pas étonnant : un grand nombre de sommets dans une structure descriptive signifie que son potentiel sémantique englobe un grand nombre de nuances sémantiques, c'est-à-dire qu'il existe un grand nombre d'usages et d'acceptions différent(e)s pour cette entrée lexicale. On a donc besoin d'un grand nombre de conditions contextuelles pour évaluer le contexte puisque ce sont les indices textuels qui permettent de discriminer les différent(e)s usages et acceptions. Ce raisonnement est aussi valable dans la situation inverse : un petit nombre de sommets sera très certainement accompagné d'un petit nombre de conditions contextuelles.

Au contraire, on notera que $\rho_{N,C}$, $\rho_{N,r}$, $\rho_{C,r}$ et $\rho_{R,r}$ ne sont pas significatifs : on ne peut pas prévoir

— le nombre de contraintes sémantiques en fonction du nombre de sommets,

— le nombre d'interprétations en fonction du nombre de sommets,

— le nombre d'interprétations en fonction du nombre de contraintes sémantiques,

— ni le nombre d'interprétations en fonction du nombre de conditions contextuelles.

On pourrait résumer ces observations de la façon suivante : les éléments appartenant à la composante dynamique du modèle (i.e. les contraintes sémantiques et les interprétations) ne semblent pas être liés aux éléments de la partie statique. On peut considérer R comme une variable endogène dépendant de N, mais le fonctionnement du modèle PELEAS laisse entendre que les variables N, C et r sont exogènes.

Cette constatation justifie pour partie l'existence même de PELEAS. En effet, une situation où les variables décrivant les éléments dynamiques du modèle auraient été endogènes aurait laissé entendre qu'un lexique dynamique n'était pas nécessaire à la modélisation des polysémies d'usage et d'acception. Ces résultats confirment l'intuition esquissée aux chapitres 2 et 3 : **l'interprétation en termes d'usages (et d'acceptions) d'une occurrence d'un mot polysémique ne semble pas prédictible en fonction de sa définition statique (conceptuelle)**.

4.6.2.6 Test de pertinence

Afin d'obtenir une idée de la pertinence des réponses du modèle, nous avons procédé à un test quantitatif. Le principe de ce test est de comparer les réponses de PELEAS avec les interprétations faites par des sujets humains sur le corpus d'évaluation. Cependant, la passation de ce test sur des sujets humains s'est avérée impossible (la durée du test est d'environ 2h pour chaque sujet, nous n'avons pu convaincre aucun sujet de se porter volontaire).

Nous n'avons donc comparé les réponses du modèle qu'avec un seul sujet : nous-même. Les résultats de la comparaison sont subjectifs et biaisés, mais fournissent un début d'indication.

Le principe de ce test est le suivant : pour chacune des 110 phrases d'évaluation, nous avons choisi dans le potentiel sémantique de l'entrée lexicale testée les vues conceptuelles qui nous semblaient pertinentes comme interprétations en contexte et nous les avons groupées par classes ordonnées de pertinences équivalentes. Pour comparer cette sélection aux résultats de PELEAS, nous avons utilisé trois indicateurs :

1. *Bruit* : comptabilise toute vue conceptuelle proposée par PELEAS qui ne figure pas dans notre interprétation de l'entrée lexicale correspondante ;

2. *Silence* : comptabilise toute vue conceptuelle figurant dans notre interprétation d'une entrée lexicale et qui n'est pas proposée par PELEAS ;

3. *Désordre* : comptabilise toute vue conceptuelle figurant à la fois dans notre interprétation et dans la liste de propositions de PELEAS, mais pas au même rang de pertinence.

Le tableau 4.6.2.6 présente ces résultats sous forme de pourcentages calculés par rapport au nombre total d'interprétations fournies par PELEAS sur l'ensemble du corpus d'évaluation.

Les résultats sont présentés sans décimales, étant donné la subjectivité des mesures.

Bruit	*Silence*	*Désordre*
8%	1%	4%

TAB. 4.5: Pourcentages de perturbations de la pertinence

4.6.3 Résultats de l'expérimentation

Au vu des données empiriques que nous avons collectées, nous pouvons donner des éléments de réponse aux questions du §4.6.1.

1. *Le modèle* PELEAS *est-il robuste ?*

   Le modèle PELEAS est robuste à condition que l'on s'astreigne à inclure une *règle de défaut* (cf. p. 115) dans une structure descriptive où les conditions contextuelles ne couvrent pas l'ensemble des alternatives contextuelles possibles. Rappelons encore une fois qu'il ne s'agit pas de déterminer un sens littéral, mais d'exprimer que, faute de mieux, on se réfère aux interprétations les plus fréquentes. En appliquant ce principe, on voit que, sur notre corpus de test, toute entrée lexicale reçoit au moins une interprétation.

   Comment PELEAS *gère-t-il les situations dégradées ?*

   — L'absence d'indice contextuel est gérée par la règle de défaut.

   — La contradiction entre des indices contextuels peut conduire soit à l'inhibition d'une partie du potentiel sémantique, soit à la superposition des usages contradictoires (ce qui revient à supposer un phénomène de dissimilation).

   — Les usages non répertoriés dans la structure descriptive sont interprétés en fonction des usages connus (cf. §4.4.1) au mieux, tandis qu'au pire on se ramène au cas où le modèle ne reconnaît aucun indice.

2. *Le modèle* PELEAS *est-il puissant ?*

   Il est difficile de caractériser la pertinence des interprétations comme une propriété intrinsèque du modèle PELEAS. Elle est plutôt fonction de la pertinence des données qui ont été injectées au départ, c'est-à-dire des structures descriptives.

   Cependant, au vu du test que nous avons effectué, nous pouvons considérer que :

   — La pertinence globale des interprétations correspond à celle d'un locuteur dont l'expérience de la langue française serait limitée à la lecture du dictionnaire et du journal *Le Monde*. L'interprétation globale est basée sur les usages associés aux grands traits stylistiques (ou aux manies rédactionnelles) du début des années 90.

   — Le principal défaut de pertinence qu'on pourrait noter est la propension au bruit : en cas d'incertitude, le modèle réagit en superposant tous les usages les plus probables, quitte à sélectionner des usages dont la pertinence est douteuse.

   — Si l'on se ramène à une expérience de la langue telle que celle que nous venons de citer, on ne constate pratiquement pas de silence dans les réponses du modèle : tout usage dont des indices sont présents dans le contexte est pris en compte.

En fait, nous dirons que **le modèle PELEAS est puissant relativement à sa base de connaissances : il se comporte comme un locuteur cherchant à maximiser son interprétation tout en disposant d'une maîtrise de la langue limitée**. Les défauts de puissance imputables à PELEAS sont en fait la conséquence de notre volonté à le rendre robuste. Pour parler plus familièrement, PELEAS préfère "sauver les meubles" que d'être incapable de répondre.

On remarquera également que, par rapport aux 11 catégories de difficultés soulevés par le corpus de test, celles pour lesquelles l'application de PELEAS semble la plus intéressante sont les divers cas de superposition, de collision et autres oscillations entre différents usages, ainsi que l'absence d'information. C'est en effet dans ces situations que la manipulation ensembliste des données inhérente à PELEAS prend toute sa signification. Il en résulte que **PELEAS est particulièrement bien adapté au traitement des jeux de mots, jeux de langage, cumuls d'emplois et mots jokers**.

3. *La mise en œuvre est-elle efficace ?*

   La réponse est partagée :

   — En ce qui concerne la construction d'une structure descriptive, il ne s'agit pas d'une mise en œuvre extrêmement efficace, puisqu'il faut en moyenne 6 heures par entrée lexicale. Cependant, la majeure partie de ce temps est imputable au logiciel BARD : c'est en fait le processus de lemmatisation qui est facteur de ralentissement. Il pourrait être notablement réduit en appliquant d'autres méthodes de mise en œuvre de la lemmatisation. Pour des raisons de logistique, nous avons préféré utiliser une mise en œuvre plus rapide à développer, mais plus coûteuse en temps de calcul.

   — Pour ce qui est du moteur d'inférences LIGHTPELEAS, l'efficacité est très satisfaisante : les structures descriptives occupent une place réduite – en moyenne moins de 3 Ko (mais encore très nettement réductible par des techniques de compression à la Huffman, par exemple) – et les temps de traitement se situent en-dessous de la demi seconde (en moyenne 413 ms) en environnement distribué, sur une machine peu puissante.

   La réponse à cette question est donc : **la mise en œuvre du modèle est efficace à condition que l'on fasse abstraction de la partie traitement statistique de corpus**. Mais les traitements statistiques de corpus ne font pas à proprement parler partie du modèle PELEAS. Ils peuvent donc être étudiés et améliorés dans un cadre autonome.

4. *La mise en œuvre est-elle coopérative ?*

   La réponse est immédiatement oui, puisque le programme de test que nous avons utilisé pour mener cette évaluation à bien simule une architecture coopérative en délocalisant volontairement les données représentant le contexte. De plus, le moteur d'inférences LIGHTPELEAS est muni d'une interface ActiveX clairement définie et facilement manipulable pour permettre à d'autres agents logiciels de le piloter et d'explorer ses réponses.

   Cependant, **le maximum de coopérativité ne peut être atteint que si les étiquettes des vues conceptuelles de toutes les structures lexicales appartiennent à la base de connaissances conceptuelles de l'architecture globale**. En effet, si une proposition d'interprétation fournie par LIGHTPELEAS ne figure pas dans cette base de connaissances, un analyseur sémantique travaillant en aval ne pourra rien en faire.

5. *Comment construire une "bonne" structure descriptive ?*

L'expérience que nous avons acquise pendant cette évaluation nous a permis d'élaborer quelques conseils méthodologiques pour la construction d'une structure descriptive :

— Le modèle PELEAS est essentiellement basé sur la *suffisance*, et non pas sur la nécessité. C'est pourquoi les conditions contextuelles ne doivent pas constituer des *conditions nécessaires et suffisantes* pour identifier tel ou tel usage : elles ne doivent être que suffisantes.

— Il vaut mieux éviter de spécifier un sommet inhibé en partie droite d'une condition contextuelle, sauf si l'on est absolument certain que tel contexte exclut absolument telle direction d'interprétation. Nous avons remarqué dans nos tests que certaines directions d'interprétation, qu'on aurait pu croire s'exclure mutuellement, pouvaient très bien se superposer (majoritairement dans le cadre des *bons mots*). Or, un sommet inhibé entraîne la suppression de toute une partie du potentiel sémantique (en vertu de la règle de méta-inhibition). C'est donc là un outil à manier avec précaution.

— Contrairement à ce que nous supposions au §4.4, les mots statistiquement extraits par le logiciels BARD servent plus à construire des conditions contextuelles (notamment des tests de co-occurrence lexicale) qu'à fournir des étiquettes pour les sommets des structures descriptives.

En résumé, cette évaluation laisse entendre que le modèle PELEAS constitue une modélisation informatique avant tout **robuste** (ce qui était notre premier objectif) permettant de **traiter la polysémie d'usage (et d'acception)** avec **pertinence** (relativement à la base d'usages disponibles) et **efficacité** (une fois que cette base est construite).

Pour ce qui est des quatre catégories d'occurrence de polysémie d'usage que nous avions définies au §2.4.1 :

— *Transferts d'usage* : PELEAS fournit des réponses pertinentes, mais de tels résultats pourraient certainement être obtenus par un système plus classique de restriction de la sélection couplé à un module de relaxation de contraintes fondé sur l'analogie, à condition que ce système soit capable de manipuler des résultats ensemblistes.

— *Cumuls d'emplois, mots jokers* et *jeux de langage* : il nous semble que c'est ici que PELEAS apporte un traitement original et puissant qui est absolument nécessaire pour pouvoir traiter ces situations. Le résultat qui nous semble le plus frappant est celui-ci : **PELEAS est capable de "décortiquer" un jeu de mots (dans les limites de ses connaissances)**.

De plus, il s'avère que la mise en œuvre que nous fournissons est tout à fait à même de s'insérer dans une architecture logicielle modulaire de traitement automatique des langues naturelles.

Chapitre 5

Mise en œuvre

> «Il en est des ordinateurs comme du dressage des chiens. Il faut être plus intelligent que le chien. Si vous dotez un ordinateur d'une intelligence supérieure à la vôtre, ce ne peut être que par accident, synergie, ou intervention divine.»
>
> **Frank Herbert**, *Destination Vide*

5.1 Introduction

Ce chapitre présente les logiciels mettant en œuvre le modèle PELEAS. Il contient essentiellement les spécifications ainsi que la documentation de ces différents logiciels. S'agissant en majeure partie de considérations propres à l'aspect informatique de ce travail, le lecteur non informaticien pourra omettre la lecture de ce chapitre à l'exception de cette introduction, qui présente notamment les motivations des choix de mise en œuvre (cf. §5.1.1), ainsi que l'architecture logicielle développée autour du modèle PELEAS (cf. paragraphe 5.1.2.1).

5.1.1 Choix concernant la mise en œuvre

Nous nous fixons l'objectif principal suivant : la mise en œuvre de PELEAS doit produire un ensemble d'outils logiciels les plus compacts, rapides et modulaires possibles. Nous voulons faire en sorte que ces outils requièrent une quantité de ressources matérielles relativement limitée car ils doivent être utilisables en tant que *composants complémentaires* d'autres applications, notamment des applications de bureautique. En effet, les outils résultant de la mise en œuvre de PELEAS sont susceptibles de toucher deux types de publics : un public issu des milieux du traitement industriel de la langue, et un public de chercheurs en traitement automatique des langues naturelles.

L'importance que nous attribuons à la bureautique est justifiée par le fait que le public d'origine industrielle est principalement composé de traducteurs, travaillant dans des environnements bureautiques : un traitement de textes, une base de données lexicographique et parfois des outils annexes tels qu'un tableur pour gérer les références croisées, ou un outil d'historique de traduction. PELEAS, dont une application possible est l'aide à la décision pour la traduction d'acceptions idiomatiques, doit donc fournir des outils

logiciels tournant avec les autres outils bureautiques. Ils doivent se présenter sous forme de modules *légers, indépendants,* et éventuellement *capables de communiquer avec d'autres applications*.

En ce qui concerne l'emploi de PELEAS dans des milieux de recherche en traitement automatique des langues naturelles, on rencontre les mêmes impératifs d'indépendance, de légèreté et de capacité de communication. En effet, PELEAS est conçu comme étant un modèle *complémentaire*, une sorte d'expert apportant son aide lors du traitement de certains types d'ambiguïtés lexicales. Il faut donc qu'il puisse fonctionner comme un module coopérant avec un système plus large de traitement de la langue, ce qui impose une certaine légèreté. De plus, nous voulons que la mise en œuvre de PELEAS soit la plus indépendante possible du système global auquel il vient se greffer, ce qui conduit à le concevoir comme un module relativement indépendant mais communiquant avec d'autres applications.

Enfin, cela implique aussi qu'il soit relativement portable d'un système à l'autre. Or, étant donné l'évolution actuelle des parcs de machines disponibles, ainsi que des systèmes d'exploitation, nous avons choisi de privilégier une mise en œuvre sur des micro-systèmes de type PC utilisant le système d'exploitation WINDOWS 95/NT4 de Microsoft. Les PC sont en effet les machines les plus répandues actuellement sur le marché de la micro-informatique, tandis que les systèmes d'exploitation WINDOWS 95/NT4 envahissent un parc de machines toujours plus grand, tout en offrant un certain nombre de fonctionnalités dont certaines sont indispensables à la mise en œuvre de PELEAS, comme les DLL (cf. Annexe C) et la technologie ActiveX (cf. Annexe B), et d'autres assurent un confort d'utilisation non négligeable (architecture du système d'exploitation orientée objets, applications multitâches, système possédant des niveaux de sécurité évitant les redémarrages intempestifs et les pertes de données irrémédiables, ...). Le système OS/2 d'IBM était également envisageable car il possède le même genre d'architecture et de fonctionnalités, mais il est présent sur un parc de machines beaucoup plus limité.

5.1.1.1 Multitâche, modularité et communication

Les systèmes d'exploitation modernes proposent des outils permettant de mettre en œuvre les principes fondamentaux du Génie Logiciel, tels que la coopération de modules indépendants, la portabilité et la réutilisabilité des composants logiciels. Ces principes étaient jusqu'ici des guides employés lors des phases de conception et de rédaction du code. Désormais, ils sont présents au niveau même de l'exécution des logiciels.

Nous proposons d'exploiter largement ces possibilités pour les raisons suivantes :

— Les systèmes d'exploitations de la famille WINDOWS 95/NT4 sont basés sur l'existence d'une machine virtuelle : les fonctions de base de ces systèmes s'adressent tous à la même machine hypothétique, les détails de mise en œuvre matériels étant cachés à l'utilisateur. Il n'est donc pas utile d'envisager un développement sur une plate-forme logicielle impliquant un niveau supplémentaire de machine virtuelle (telles que Lisp ou Smalltalk) : la portabilité du code résultant est déjà assurée par le système d'exploitation.

— Toujours dans le même esprit, il est inutile de s'appuyer sur une plate-forme logicielle d'intégration d'applications concurrentes (comme Smalltalk ou Smalltalk distribué) car le système d'exploitation possède lui-même des mécanismes gérant la coopération d'applications multitâches. Les applications sont totalement concurrentes et peuvent être lancées sans prérequis logiciel. Elles peuvent communiquer entre elles soit par l'intermédiaire de flots de communication en mémoire, soit en partageant des espaces de mémoire communs (qui forment des embryons de tableaux noirs, directement disponibles au niveau système).

— La technologie ActiveX offre des mécanismes de communication inter-applications étendue et de distribution des processus basés sur des échanges client/serveur. Nous pouvons trouver, en particulier, un usage immédiat pour trois ensemble de mécanismes gérés par ActiveX :

1. les *Data Objects* qui forment une interface orientée objets pour le transfert des données entre deux applications ;
2. l'interface de programmation *Automation* qui permet à une application d'en piloter une autre ;
3. et enfin les *contrôles ActiveX*. Grâce à ce mécanisme, il est possible de publier une structure de données au niveau du système d'exploitation. Celle-ci est géré par l'application serveur qui en a publié le format. Toute application client peut alors utiliser cette structure sans se soucier des détails de sa mise en œuvre. C'est l'application serveur qui prend en charge de manière transparente le comportement des structures qu'elle publie (en général il s'agit de leur affichage et de leur édition) ; il est aussi à sa charge de publier l'interface de ses structures, ainsi une application client peut demander à l'application serveur de lui fournir l'ensemble des méthodes applicables.

— Pour finir, il nous semble important de recourir aux fichiers DLL qui permettent d'établir des liaisons dynamiques en cours d'exécution avec des portions de codes déjà compilées. Les fonctions contenues dans une DLL deviennent des parties intégrantes du système d'exploitation et il est possible de les invoquer dynamiquement en connaissant uniquement leur nom et le nom de la bibliothèque dans laquelle elles se trouvent. Un autre avantage majeur de l'usage des DLL est qu'il est ainsi possible de réutiliser des fonctions compilées par une autre plate-forme de développement (un autre langage, un autre compilateur) car elles stockent le code exécutable de manière normalisée, et parce que le mécanisme d'édition de liens dynamiques est une fonctionnalité du système d'exploitation, non du compilateur ou du langage.

5.1.1.2 Choix d'une plate-forme de développement

Il existe sur le marché plusieurs plates-formes de développement qui permettent de mettre en œuvre pour WINDOWS 95/NT4 les divers protocoles et technologies que nous venons d'évoquer. Le choix de l'une d'entre elles s'est ramené au choix d'un langage et d'une bibliothèque de structures gérant les appels au système d'exploitation.

Nous avons finalement retenu le couple C++/MFC (c'est-à-dire la plate-forme *Visual C++ 5* de Microsoft) pour les raisons suivantes :

— Il s'agit d'un compilateur C++ à la norme ANSI 3.0 qui présente plusieurs avantages :

1. C'est un langage orienté objets offrant un bon niveau d'encapsulation des données (grâce aux déclarations publiques / protégées / privées), la possibilité d'utiliser l'héritage multiple, et de simuler un certain contrôle des méta-classes (par l'intermédiaire des *membres statiques de classe*).
2. Au contraire du langage C et des versions précédentes de C++, cette version est un langage fortement typé (au même titre que ADA), ce qui évite les changements de types implicites. On dispose également de mécanismes pour contrôler les flux entrant et sortant des routines (paramètres déclarés constants, et paramètres références).

3. Cette version du langage est équipée de la technologie RTTI (pour *RunTime Type Information*) qui permet de créer dynamiquement des objets dont le type n'est pas connu à la compilation.
4. On dispose également de la généricité comme en ADA (ce sont les *templates*), ainsi que du traitement des exceptions.

— Ce compilateur est accompagné de la bibliothèque de classes MFC de Microsoft (pour *Microsoft Foundation Classes*) qui encapsule tous les objets du système d'exploitation (et tous les appels de fonctions systèmes) dans un ensemble de classes cohérent. Cette bibliothèque offre l'avantage particulier d'avoir été écrite par les concepteurs du système d'exploitation, on y trouve donc des classes encapsulant tous ses mécanismes, même les plus avancés.

Bien sûr, il aurait été possible d'utiliser d'autres plates-formes produisant un code exécutable de qualité identique à celle que nous avons choisie. Mais un autre choix aurait signifié soit un autre langage (qui n'aurait pas possédé les qualités que nous recherchons dans C++, c'est-à-dire la rapidité d'exécution du fait de sa compilation, la sécurité proche de celle offerte par ADA, et une souplesse dans la manipulation des types et des méta-classes empruntée, mais non égalée, à Smalltalk), soit une autre bibliothèque de classes (par exemple la bibliothèque OWL de Borland), mais nous avons jugé qu'il était préférable *a priori* d'utiliser la bibliothèque fournie par les concepteurs du système d'exploitation (on peut espérer que l'esprit est le même entre les deux produits).

5.1.2 Architecture

Quatre outils ont été développés autour du modèle PELEAS :

— *un moteur*, LIGHTPELEAS, qui est le programme permettant, pour un mot donné, de charger sa représentation en mémoire sous forme d'entrée lexicale, de vérifier si le contexte textuel s'unifie avec au moins une des règles contextuelles attachées au graphe, et d'appliquer l'algorithme de valuation du graphe si l'unification a réussi ;

— *un éditeur*, MELISANDE, permettant de créer et/ou de modifier des entrées lexicales visuellement ;

— *un analyseur de corpus*, BARD, qui assiste l'expert dans sa tâche de création d'une base d'entrées lexicales en extrayant des données significatives d'un corpus donné ;

— et *un analyseur morphologique*, AMORPH, qui n'est pas directement lié au modèle PELEAS, mais est utilisé par BARD.

Différents types de contraintes opèrent à différents niveaux d'architecture pour la mise en œuvre de PELEAS :

— au niveau de l'architecture logicielle des différents constituants de la mise en œuvre ;

— au niveau de l'architecture des flots d'information à travers le système informatique mettant en œuvre PELEAS ;

— au niveau de l'architecture fonctionnelle interne du moteur de mise en œuvre ;

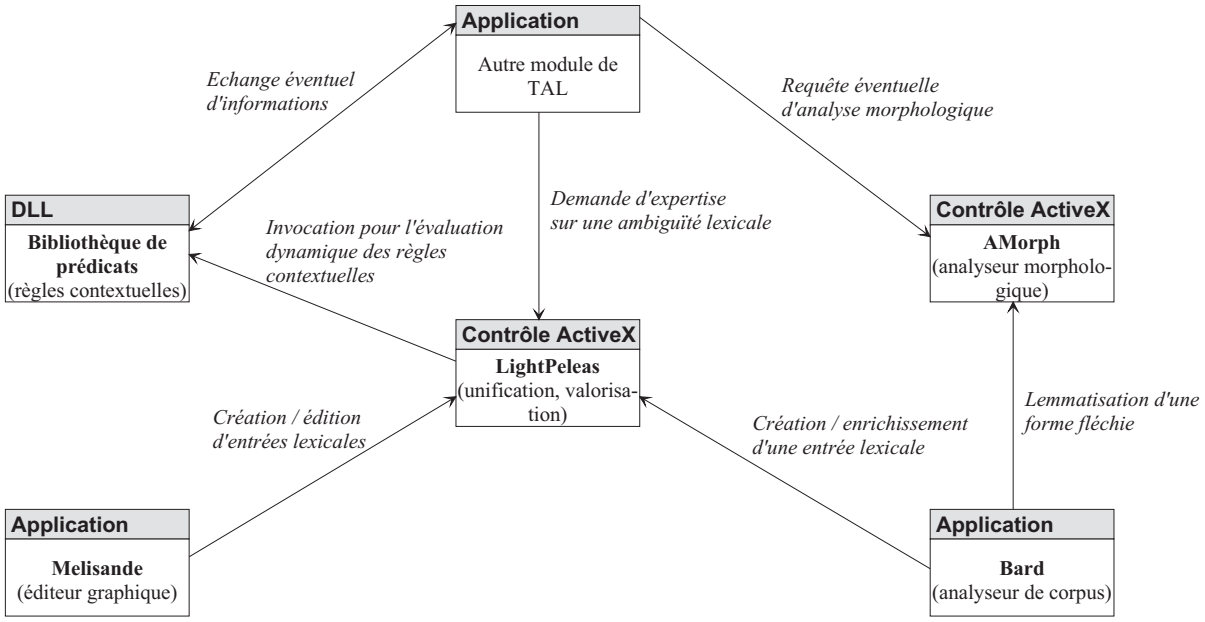


FIG. 5.1: Architecture logicielle

5.1.2.1 Architecture logicielle

Les relations entre les différents composants logiciels sont schématisés sur la figure 5.1. D'autres éléments sont également présents sur cette figure car ils sont utilisés par notre architecture.

A. L'élément fondamental sur lequel repose cette architecture est le contrôle ActiveX LIGHTPELEAS qui contient le moteur d'unification et de calcul des configurations des entrées lexicales (cf. §5.2). Ce contrôle met en œuvre la structure de graphe qui est au centre du modèle PELEAS, et offre toutes les routines permettant de la manipuler. En pratique, il exporte un certain nombre de classes au niveau du système d'exploitation, qui vont permettre d'accéder à tous les éléments constitutifs d'une entrée lexicale, de procéder à son unification avec le contexte ainsi qu'à l'application de l'algorithme de valorisation, et de récupérer les résultats.

B. Le moteur fait lui-même appel à une ou plusieurs DLL pour le fonctionnement des règles contextuelles. En effet, la partie gauche d'une règle contextuelle contient une formule logique formée de prédicats qui représentent des conditions sur différents éléments du contexte d'analyse. Cela nous pousse à implémenter ces prédicats dans des DLL autonomes pour les deux raisons suivantes (cf. §5.4) :

   — Dans le format de représentation statique des graphes PELEAS, les prédicats sont référencés uniquement par leur nom lorsqu'ils apparaissent en partie gauche d'une règle contextuelle. On a donc un problème de liaison lorsqu'on veut unifier la partie gauche d'une règle contextuelle avec le contexte d'analyse : comment retrouver le code exécutable d'un prédicat en connaissant juste son nom ? Ceci suggère l'emploi d'un mécanisme de liaison dynamique, qui est précisément le fondement des DLL.

   — De plus, le code même des prédicats est complètement dépendant du reste du système de traitement automatique de la langue sur lequel se greffe PELEAS (alors que le moteur LIGHTPELEAS est conçu pour en être le plus indépendant possible) : ce code dépend des structures de

données utilisées dans le reste du système, des objets qu'elles incarnent et aussi des processus de communication entre les différents modules. Ainsi, si le contrôle LIGHTPELEAS peut être utilisé avec n'importe quel système de traitement de la langue (pour peu qu'il supporte ActiveX), le code des prédicats doit, lui, être réécrit en fonction de ce système. Or, le fait de le placer dans des DLL autonomes permet à PELEAS de considérer les prédicats comme faisant partie de son système hôte, et donc de le rendre indépendant de leur mise en œuvre.

C. En dehors du moteur LIGHTPELEAS nous trouvons les deux outils qui constituent "l'atelier de maintenance" de la base d'entrées lexicales : l'éditeur/visualisateur de graphes MELISANDE (cf. §5.3) et l'analyseur de corpus BARD (cf. §5.5). Avec ces outils, il devient possible de créer ou de modifier un graphe de deux façons : soit en le construisant à partir d'une structure vide (en utilisant uniquement l'éditeur de graphes MELISANDE), soit en partant d'une structure initiale proposée par l'analyseur de corpus BARD et en la modifiant ensuite.

D. Enfin, on trouve également un contrôle ActiveX dédié à l'analyse morphologique, AMORPH, qui n'a pas directement trait au modèle PELEAS, mais est utilisé par l'analyseur de corpus BARD afin de lemmatiser les formes fléchies qu'il a extraites de corpus.

5.1.2.2 Architecture fonctionnelle

Nous savons maintenant quels modules ou applications constituent la mise en œuvre du modèle PELEAS, de quelle façon ils communiquent entre eux et quels contrôles de processus ils exercent les uns sur les autres. Nous nous attachons ici à décrire non plus les contrôles (i.e. à établir un diagramme de commandes comme au paragraphe précédent), mais quelles informations sont échangées à un niveau macroscopique autour du module LIGHTPELEAS (cf. figure 5.2).

Fonctionnellement, on peut considérer le moteur LIGHTPELEAS comme une boîte noire prenant en entrée un mot (ou une lexie) que l'on suppose être le foyer d'une expression lexicalement ambiguë, ainsi qu'une représentation du contexte d'analyse au moment où l'ambiguïté lexicale a été détectée. Dans les faits, le moteur LIGHTPELEAS n'attend aucun paramètre formel représentant le contexte : celui-ci entre en jeu uniquement au moment de l'évaluation des règles contextuelles. Mais à un niveau macroscopique, cela revient bien à requérir une image mémoire du contexte d'analyse pour que le moteur puisse fonctionner.

Pour son fonctionnement, LIGHTPELEAS requière également deux autres sources de données : une base d'entrées lexicales et une (ou plusieurs) base(s) de prédicats (sur la figure, les flèches grasses indiquent des informations entrantes ou sortantes ayant trait directement avec le traitement de l'ambiguïté lexicale détectée, et les flèches maigres indiquent des flots d'informations secondaires, adjuvantes au traitement).

A l'autre extrémité du traitement, le module LIGHTPELEAS fournit en sortie un graphe valué qui est une représentation de l'interprétation en contexte de l'occurrence du mot qui est le foyer de l'ambiguïté traitée. Mais ce n'est pas suffisant : dans le cadre d'une coopération de LIGHTPELEAS avec d'autres modules de traitement automatique des langues naturelles, des traitements ultérieurs possibles sont indiqués par des flèches en pointillés. Ainsi, pour utiliser les résultats fournis par LIGHTPELEAS dans un système de représentation sémantique plus classique, il est tout à fait envisageable de faire passer le graphe valué à travers un processus d'identification ayant pour but d'apparier les noeuds saillants du graphe avec des représentations sémantiques d'une base de connaissances (grâce à des critères de concordance

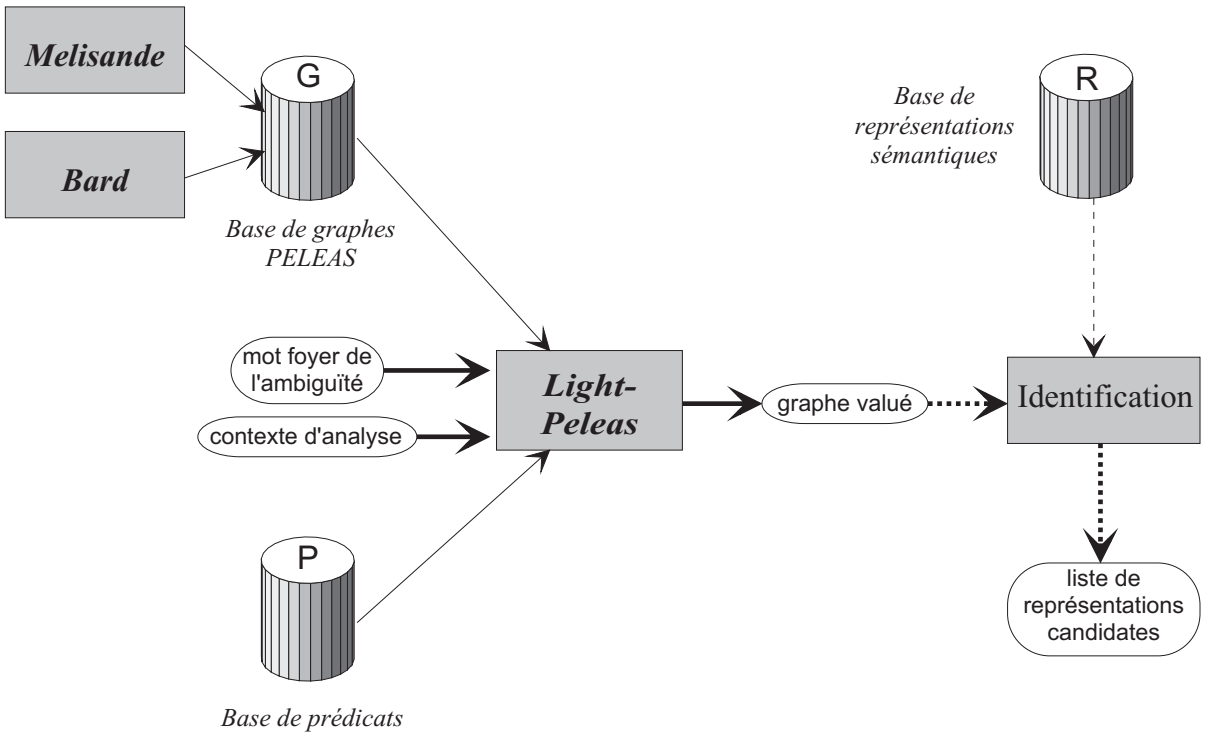


FIG. 5.2: Flot général d'information autour de LIGHTPELEAS

des dénominations entre les nœuds et les représentations) afin de "traduire" le graphe en une liste de représentations candidates au statut d'interprétations valides de l'expression ambiguë.

5.1.2.3 Architecture structurelle

Après avoir fait l'analyse fonctionnelle autour du moteur LIGHTPELEAS (i.e. observer son comportement en le considérant comme une boîte noire), nous procédons ici à son analyse structurelle. Le détail des informations qui circulent à l'intérieur de LIGHTPELEAS lors de l'analyse d'un mot est schématisé sur la figure 5.3.

L'analyse en contexte par LIGHTPELEAS de l'occurrence d'un mot qui est le foyer d'une ambiguïté lexicale comporte quatre étapes, qui sont représentées sur la figure 5.3 par des rectangles. Les données qui circulent à l'intérieur de LIGHTPELEAS pendant l'analyse sont représentées par des rectangles arrondis.

On peut faire l'interprétation suivante de ce schéma :

1. Il faut tout d'abord *retrouver le graphe* correspondant au mot en entrée (le foyer de l'ambiguïté) dans la base d'entrées lexicales que possède LIGHTPELEAS. S'il n'est pas possible de retrouver ce graphe, LIGHTPELEAS est en échec (il s'agit d'un mot inconnu), sinon on appelle G ce graphe et on passe à l'étape suivante.

2. La deuxième étape consiste à effectuer la liaison dynamique du moteur LIGHTPELEAS avec le code exécutable des prédicats figurant en parties gauches des règles contextuelles attachées au graphe G. En effet, dans la représentation statique du graphe, on ne dispose que des noms des prédicats composant les règles contextuelles. On va donc se servir des fonctionnalités de liaison

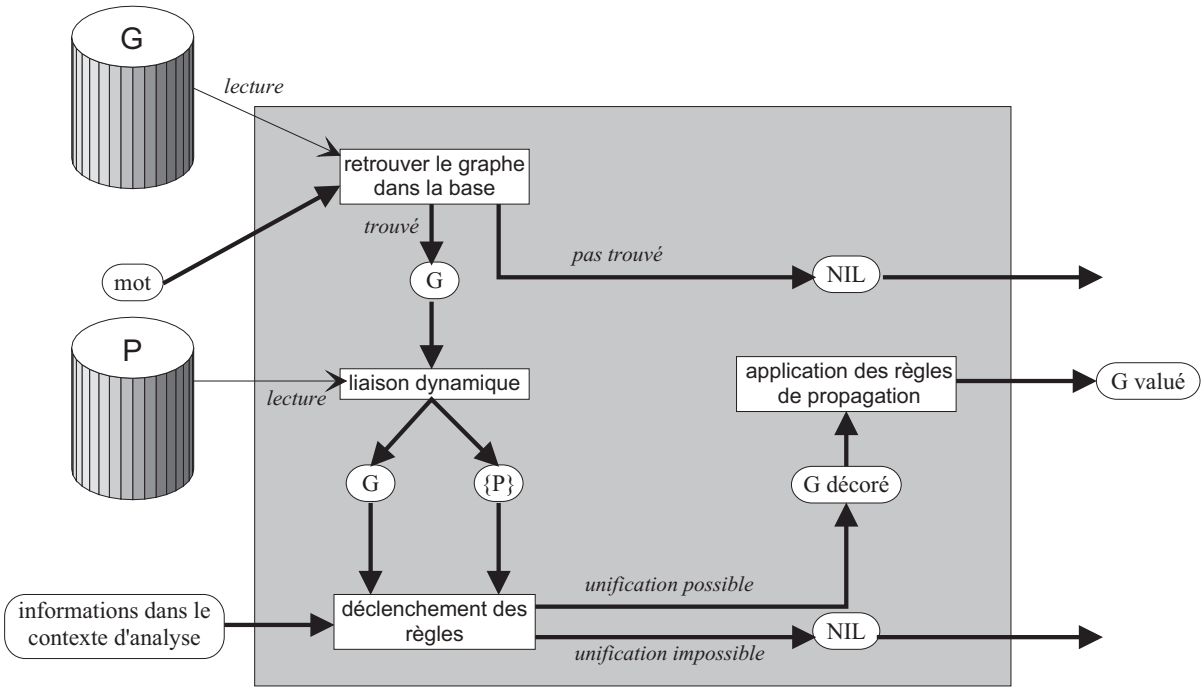


FIG. 5.3: Flot d'information à l'intérieur de LIGHTPELEAS

dynamique évoquées plus haut pour retrouver, en fonction de son nom, l'adresse de chaque prédicat dans la base de prédicats, et le charger en tant que code exécutable dans l'espace mémoire du processus LIGHTPELEAS. A l'issue de cette étape, on dispose donc de deux informations : le graphe G ainsi que la liste $\{P\}$ des prédicats directement exécutables.

3. On peut maintenant entamer le processus de valuation du graphe G. Pour cela, il faut que la partie gauche d'au moins une des règles contextuelles attachées à G soit unifiable avec le contexte d'analyse. Pour vérifier cette condition, on va exécuter les prédicats de la partie gauche de chaque règle et composer leurs résultats avec les connecteurs logiques appropriés. Si au moins une des formules en partie gauche s'évalue à vrai, alors c'est que l'unification de cette règle avec le contexte est possible, sinon il n'y a aucune unification possible, et dans ce cas LIGHTPELEAS est en échec. Pour toutes les unifications possibles, on va déclencher la règle correspondante, i.e. on va procéder aux attributions de valeurs d'activités prescrites pour les noeuds de G qui figurent en partie droite de la règle. On obtient comme résultat une valuation initiale de G, que nous qualifions maintenant de "décoré".

4. La dernière étape est celle où l'on va compléter la valuation de G grâce aux règles de propagation d'activité standard entre les nœuds du graphe. Ces règles sont toujours identiques quel que soit le graphe à valoriser donc LIGHTPELEAS n'a pas besoin d'autre information que le graphe lui-même. Au final, on obtient une version valuée du graphe G en fonction du contexte d'analyse, et c'est le résultat que va fournir LIGHTPELEAS en sortie.

5.1.3 Présentation des logiciels

Les sections suivantes présentent les quatre modules logiciels qui ont été introduits ci-dessus, ainsi qu'une section concernant le développement des bases de prédicats.

Les modules LIGHTPELEAS et AMORPH étant des contrôles ActiveX, il est recommandé de prendre préalablement connaissance de l'annexe B qui présente rapidement les principes fondamentaux de la technologie ActiveX.

De même, la lecture de l'annexe C, qui présente le fonctionnement des DLL, est conseillée pour mieux comprendre le §5.4 qui traite des bases de prédicats.

Pour les §5.2 et §5.6, on pourra trouver en regard (à l'annexe D) les spécifications formelles des classes publiées dans le système d'exploitation. Pour ce faire, nous avons choisi d'adopter le formalisme des *spécifications algébriques*, notées avec le langage PLUSS[1] dont on trouvera une description par Gaudel [42] et Choquer [21]. Pour tester la validité de ces spécifications, nous nous sommes servis de l'outil ASSPEGIQUE (cf. Choppy [19]) et d'un prototypage en C++.

Pour les §5.3 et §5.5 présentant respectivement MELISANDE et BARD, nous avons plutôt fourni une documentation de ces logiciels pouvant servir de manuel utilisateur.

5.2 Le modèle objet LIGHTPELEAS

Nous avons mis en œuvre le modèle PELEAS au sein d'un sorte de "noyau", autour duquel s'articulent les différents outils logiciels qui y font appel, notamment l'atelier d'édition graphique de lexiques MELISANDE (cf. §5.3), et l'outil d'exploration de corpus BARD (cf. §5.5).

Ce noyau, appelé LIGHTPELEAS, est un serveur COM fournissant un accès à 25 coclasses. Celles-ci permettent de manipuler finement l'ensemble des éléments qui composent un potentiel sémantique. Ce serveur a reçu l'attribut de "light" (léger) car il ne contient que des considérations algorithmiques indispensables à la mise en œuvre du modèle PELEAS (ainsi que celles rendues nécessaires par son intégration dans le modèle COM). Il ne contient aucun code concernant une quelconque interface graphique. C'est donc le "noyau dur" de la mise en œuvre, "allégé" de tout ce qui ne concerne pas strictement le fonctionnement de PELEAS.

Dans la terminologie Microsoft, un tel serveur, fournissant un accès à plusieurs coclasses, est appelé un *modèle objet* (*object model*), car il permet de modéliser un domaine d'activité à l'aide d'un ensemble d'objets COM.

DONNÉES TECHNIQUES	
LIGNES DE CODE :	**33.081 lignes**
TAILLE DE L'EXÉCUTABLE :	**604 Ko**
EMPREINTE MÉMOIRE MAXIMALE :	**2 Mo**

5.2.1 Les coclasses de LIGHTPELEAS

Le modèle objet fourni par LIGHTPELEAS contient, pour chaque entité constitutive d'un potentiel sémantique, une coclasse représentant cette entité, plus une coclasse représentant une collection de ces entités. Le modèle contient en plus une coclasse spéciale appelée `Engine` qui représente le modèle PELEAS lui-même et son moteur d'inférences.

[1] Proposition de Langage Utilisable pour les Spécifications Structurées

Seule la coclasse `Engine` permet d'engendrer des objets COM au vrai sens du terme. Toutes les autres coclasses engendrent des *propriétés étendues*. La différence entre un objet COM et une propriété étendue réside dans le fait qu'un client d'un serveur COM peut demander explicitement la création d'une instance d'un objet COM et la détruire à son gré, tandis que la création et la destruction d'une propriété étendue sont contrôlées par un autre objet COM (ou une propriété étendue) : le client peut uniquement obtenir une référence sur une propriété étendue et lui appliquer les méthodes de ses interfaces. Du point de vue de la POO, les propriétés étendues sont des objets dont le constructeur et le destructeur sont privés, non accessibles à l'extérieur. Cette mise en œuvre était nécessaire, car toutes les coclasses autres que `Engine` représentent des données manipulées par le moteur d'inférences. En faire des objets COM normaux aurait signifié qu'un client aurait été capable soit d'instancier une donnée sans nécessairement la relier aux autres de manière appropriée (*a priori* seul le moteur d'inférences sait comment les différentes données interagissent), soit de provoquer la destruction d'une donnée sans en informer le moteur d'inférences. Ces deux possibilités ne sont pas acceptables pour la cohérence de l'ensemble.

La figure 5.4 schématise les relations entre les différentes coclasses du modèle objet LIGHTPELEAS. La coclasse `Engine` apparaît sur fond gris pour signaler que c'est la seule qui permet d'instancier des objets COM. Toutes les autres sont des classes de propriétés étendues. Tous les traits de ce schéma représentent des relations de composition (et non d'héritage).

Pour la suite de cette section, nous allons adopter une convention sur le profil des propriétés et des méthodes des différentes coclasses. En effet, la tradition de COM veut qu'un profil soit présenté comme il le serait pour une utilisation en Visual Basic, ce qui est pourtant assez éloigné du profil réel déclaré dans l'IDL du serveur. La convention que nous avons choisie se situe à mi-chemin entre le Visual Basic et l'IDL :

— Au lieu d'annoncer le type de retour obligatoire `HRESULT`, nous donnerons le type de retour effectif d'une fonction ou d'une propriété, mais présenté comme en C++.

— Les références sur des objets ou des interfaces seront présentées en utilisant la notation étoilée du C++ (qui n'apparaît pas en Visual Basic).

— Les éventuels paramètres facultatifs seront notés entre crochets, avec leur type effectif.

Ainsi, pour la propriété `Label`, nous donnerons le profil :

```
BSTR Label
```

à la place des deux profils complémentaires :

```
HRESULT get_Label([out, retval] BSTR* pVal);
HRESULT put_Label([in] BSTR newVal);
```

et pour la propriété `Domains` (en lecture seule) :

```
IDomains* Domains
```

à la place de :

```
HRESULT get_Domains([out, retval] IDomains* pVal);
```

Pour une méthode, nous écrirons par exemple :

```
GenerateIndex([BSTR indexFile])
```

à la place de :

```
HRESULT GenerateIndex([in, optional] VARIANT indexFile);
```

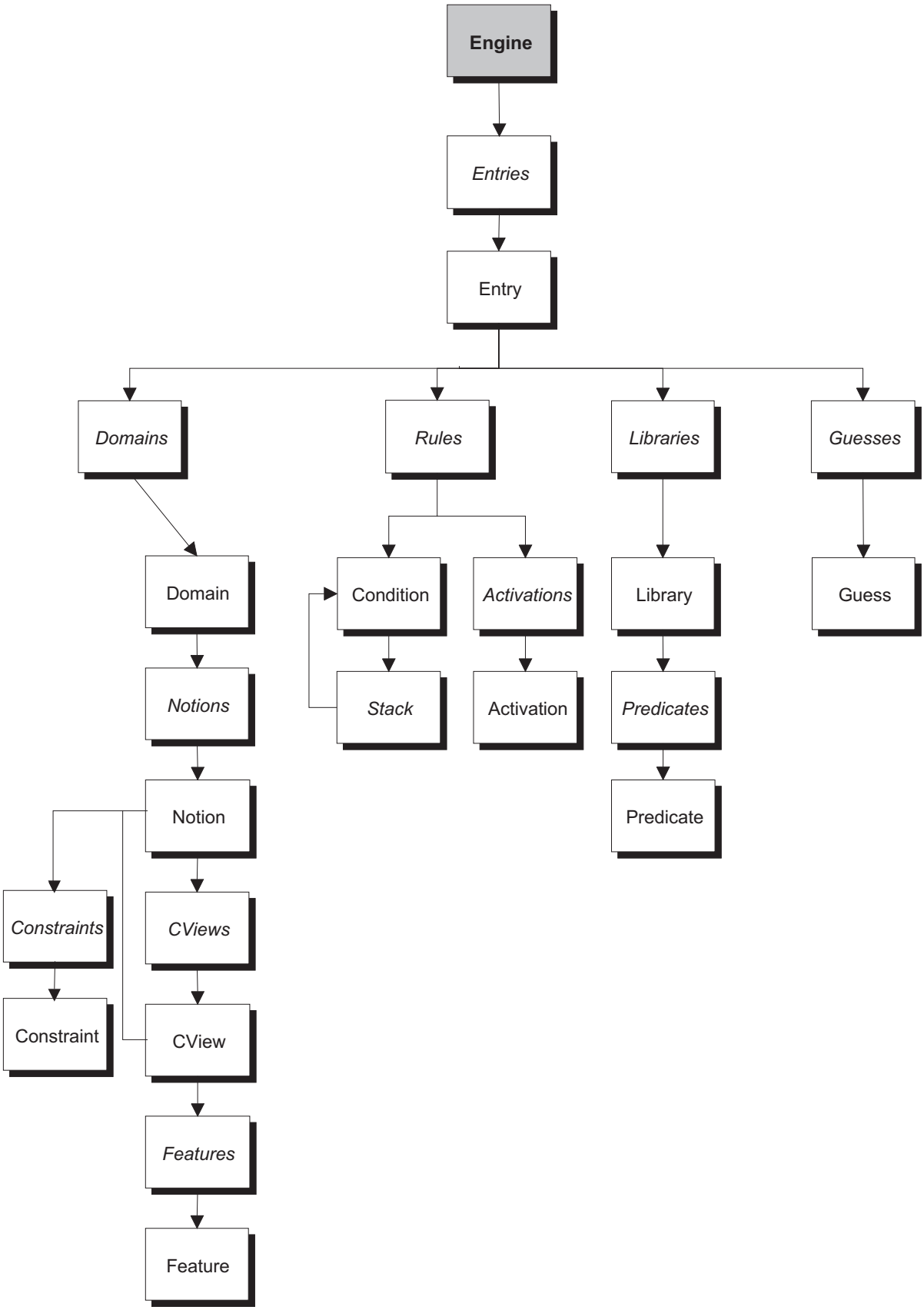


FIG. 5.4: Le modèle objets de LIGHTPELEAS

Les coclasses qui figurent en italiques dans la figure 5.4 sont des collections respectant l'usage en vigueur dans Visual Basic : elles contiennent (possèdent des références sur) un nombre arbitraire d'instances d'autres coclasses (cf. §B.2.4).

5.2.1.1 La coclasse Engine

La coclasse `Engine` engendre des instances qui sont des vrais objets COM, et c'est la seule dans le modèle objet LIGHTPELEAS. Un client de LIGHTPELEAS doit donc toujours passer par l'intermédiaire d'une instance de `Engine` afin d'accéder à ses services. Chaque création d'une instance de `Engine` correspond à la mise en service d'un nouveau serveur LIGHTPELEAS possédant ses propres données. Ainsi, le même client peut créer deux instances de `Engine` et manipuler le même potentiel sémantique à travers ces deux instances sans que les modifications apportées par une instance ne se répercutent pour l'autre.

Lors de la création d'une instance de `Engine` par un client, si le processus échoue à un moment quelconque de son déroulement, toutes les ressources allouées sont restituées au système et une exception est levée.

Spécification formelle : cf. annexe D.1.1

Interface IEngine

La coclasse `Engine` exporte l'interface duale `IEngine` contenant les propriétés et méthodes suivantes :

— `BSTR EntriesDirectory` : propriété en lecture / écriture qui permet de connaître et de modifier le répertoire de travail de cette instance du serveur. Lors de la modification de cette propriété, un test est effectué pour vérifier que le nouveau répertoire existe bien. En cas d'échec, une exception est levée.

— `BSTR IndexFile` : propriété en lecture / écriture contenant le chemin du fichier d'index utilisé par cette instance du serveur afin d'optimiser la sauvegarde des données. Ce fichier d'index, correspondant à un lexique, contient l'ensemble des entrées de ce lexique et leurs emplacements respectifs sur le disque ou sur le réseau. Un test de validité du nouveau nom de fichier est effectué lors de la modification. En cas d'échec, une exception est levée.

— `GenerateIndex([BSTR indexFile])` : méthode permettant d'engendrer un nouveau fichier d'index à partir des entrées lexicales chargées par cette instance du serveur. Le paramètre `indexFile` est optionnel : par défaut, le fichier d'index engendré porte le même nom que le contenu de la propriété `IndexFile`. En cas de problème lors de la génération du nouvel index, ou si l'argument `indexFile` passé n'est pas de type `BSTR`, une exception est levée.

— `IEntries* Entries` : propriété en lecture seule qui renvoie une référence sur l'interface `IEntries` de l'instance de la collection `Entries` utilisée par cette instance de `Engine` pour gérer les entrées lexicales qu'elle a en mémoire.

5.2.1.2 La coclasse Entries

Cette coclasse est, dans la terminologie Microsoft, une *collection* (cf. §B.2.4). C'est-à-dire qu'elle se comporte comme un conteneur par rapport à un ensemble d'instances, ici de la coclasse `Entry`. Elle expose donc dans son interface duale `IEntries` les propriétés et méthodes que nous présentons au §B.2.4 (nous ne mentionnerons pas la propriété `_NewEnum` puisqu'elle n'est pas utilisable directement). Elle expose en outre des méthodes destinées à la lecture d'une entrée dans un fichier, ainsi que pour gérer le Presse-papiers du système d'exploitation.

Spécification formelle : cf. annexe D.1.2

Interface IEntries

— `long Count` : propriété en lecture seule renvoyant le nombre d'entrées dans la collection.

— `IEntry* Item(VARIANT Index)` : propriété en lecture seule qui renvoie une référence sur l'élément dont l'indice est spécifié. Le paramètre `Index` peut être soit une chaîne de caractères (`BSTR`) représentant le nom de l'élément, soit un entier (long ou court) représentant sa position dans la collection. Tout autre type lève une exception. S'il s'agit d'un paramètre numérique inférieur à 1 ou supérieur au nombre d'éléments de la collection, ou s'il s'agit d'un paramètre alphanumérique ne correspondant au nom d'aucun élément de la collection, une exception est également levée.

— `Add(BSTR EntryName)` : méthode qui crée un nouvel élément de type `Entry` portant le nom `EntryName` et l'ajoute à la collection. Si un élément porte déjà ce nom dans la collection, une exception est levée.

— `Remove(VARIANT Index)` : méthode qui supprime l'élément d'indice spécifié dans la collection. Les mêmes restrictions sur le type et la valeur du paramètre `Index` s'appliquent que pour la propriété `Item`.

— `RemoveAll` : méthode qui supprime tous les éléments de la collection.

— `Paste` : méthode qui ajoute à la collection le contenu du Presse-papiers. Si le Presse-papiers est vide ou si les données qu'il contient ne sont pas de type `Entry`, une exception est levée.

— `Open(BSTR FileName)` : méthode qui ouvre le fichier spécifié et ajoute à la collection les données qu'il contient. Si le fichier n'existe pas, ou s'il ne contient pas la description d'une entrée au format PDL, une exception est levée.

— `SaveAll` : méthode qui enregistre toutes les entrées de la collection dans des fichiers respectifs. Toutes les entrées doivent avoir reçu un nom de fichier associé au préalable.

— `Retrieve(BSTR EntryName)` : méthode semblable à `Open` mais en se basant sur le fichier d'index. Si l'entrée spécifiée par `EntryName` n'apparaît pas dans le fichier d'index, une exception est levée.

5.2.1.3 La coclasse Entry

La coclasse `Entry` permet de gérer une entrée lexicale. Elle sert de point d'accès pour retrouver toutes les informations relatives à une entrée lexicale.

Spécification formelle : cf. annexe D.1.3

Interface IEntry

— `[BSTR] Label([BSTR newVal])` : propriété en lecture / écriture qui permet de consulter ou modifier l'étiquette de cette entrée. En écriture, si une autre entrée porte déjà le nom spécifié par `newVal`, une exception est levée.

— `IDomains* Domains` : propriété en lecture seule qui renvoie une référence sur la collection des domaines associés à cette entrée.

— `Copy` : méthode qui place sur le Presse-papiers une copie des informations contenues dans cette entrée.

— `Cut` : méthode similaire à `Copy` mais qui en outre supprime cette entrée (toute référence sur cette entrée devient donc invalide).

— `ILibraries* Libraries` : propriété en lecture seule qui renvoie une référence sur la collection des bibliothèques de prédicats associées à cette entrée.

— `IRules* Rules` : propriété en lecture seule qui renvoie une référence sur la collection des règles contextuelles associées à cette entrée.

— `IGuesses* Guesses` : propriété en lecture seule qui renvoie une référence sur la collection des propositions d'interprétation de l'entrée en contexte (après évaluation).

— `BSTR Path` : propriété en lecture seule qui affiche sous forme lisible le chemin allant de l'entrée jusqu'au nœud actuel. Ici, `Path` est donc équivalent à `Label`.

— `Save(BSTR FileName)` : méthode qui enregistre l'entrée dans le fichier spécifié (et associe ce nom de fichier à l'entrée jusqu'à ce qu'un nouvel appel de `Save` change le nom du fichier). L'entrée est également référencée dans le fichier d'index.

— `Evaluate` : méthode qui lance le processus d'évaluation du sens en contexte.

— `BOOL IsEvaluated` : propriété en lecture seule qui indique si l'entrée a déjà été évaluée.

— `pelNodeType Type` : propriété en lecture seule qui renvoie un code identifiant le type du nœud courant, soit `pelENTRY` ici.

5.2.1.4 La coclasse Domains

Cette coclasse est également une collection. Etant de surcroît une collection de nœuds d'une structure descriptive, elle partage avec les coclasses `Notions`, `CViews` et `Features` une interface quasi-identique. C'est pourquoi nous détaillerons entièrement la classe `Domains` uniquement.

Spécification formelle : cf. annexe D.1.4

Interface IDomains

— `long Count` : propriété en lecture seule renvoyant le nombre d'éléments dans la collection.

— `IDomain* Item(VARIANT Index)` : propriété en lecture seule qui renvoie une référence sur l'élément dont l'indice est spécifié par le paramètre `Index` qui peut être une chaîne de caractères (`BSTR`) ou un entier (long ou court). Tout autre type de paramètre, ainsi que des valeurs de `Index` correspondant à un élément qui ne figure pas dans la collection, provoquent la levée d'une exception.

— `Add(BSTR DomainName)` : méthode qui crée un nouvel élément de type `Domain` nommé d'après `DomainName`, et l'ajoute à la collection. Si un élément porte déjà ce nom, une exception est levée.

— `Remove(VARIANT Index)` : méthode qui supprime un élément donné de la collection. Les mêmes restrictions que dans `Item` s'appliquent au paramètre `Index`.

— `RemoveAll` : méthode qui supprime tous les éléments de la collection.

— `Paste` : méthode qui ajoute à la collection les données situées sur le Presse-papiers, à condition qu'il ne soit pas vide et qu'un élément de type `Domain` y figure. Sinon, une exception est levée.

— `IEntry* Father` : propriété en lecture seule qui renvoie une référence sur l'objet `Entry` qui contient cette collection.

5.2.1.5 La coclasse Domain

Cette coclasse représente un domaine de la structure descriptive. Elle partage avec les coclasses `Notion`, `CView` et `Feature`, qui représentent également des nœuds de structure descriptive, une interface quasi identique. Nous ne détaillerons donc que la coclasse `Domain`.

Spécification formelle : cf. annexe D.1.5

Interface IDomain

— `[BSTR] Label([BSTR newVal])` : propriété en lecture / écriture qui permet de consulter ou modifier l'étiquette de ce nœud. En écriture, si un autre nœud porte déjà le nom spécifié par `newVal`, une exception est levée.

— `[pelActivity] Activity([pelActivity newVal])` : propriété en lecture / écriture qui permet de consulter ou modifier le taux d'activité de ce nœud.

— `INotions* Notions` : propriété en lecture seule qui renvoie une référence sur la collection des notions contenue par ce nœud.

— `Copy` : méthode qui place sur le Presse-papiers une copie des informations contenues dans ce nœud.

— `Cut` : méthode similaire à `Copy` mais qui en outre supprime ce nœud (toute référence sur ce nœud devient donc invalide).

— `IConstraints* Constraints` : propriété en lecture seule qui renvoie une référence sur la liste des contraintes sémantiques existant entre les nœuds fils de ce nœud.

— `BSTR Path` : propriété en lecture seule qui affiche sous forme lisible le chemin allant de l'entrée jusqu'au nœud actuel, avec les étiquettes séparées par des points.

— `pelNodeType Type` : propriété en lecture seule qui renvoie un code identifiant le type du nœud courant, soit `pelDOMAIN` ici.

— `IDomains* Father` : propriété en lecture seule qui renvoie une référence sur la collection qui contient ce nœud.

5.2.1.6 La coclasse Notions

La coclasse `Notions` présente un comportement similaire à celui de la coclasse `Domains`. Son interface `INotions` expose les éléments suivants :

— `long Count` : propriété en lecture seule renvoyant le nombre d'éléments dans la collection.

— `INotion* Item(VARIANT Index)` : propriété en lecture seule qui renvoie une référence sur l'élément dont l'indice est spécifié par le paramètre `Index` qui peut être une chaîne de caractères (`BSTR`) ou un entier (long ou court). Tout autre type de paramètre, ainsi que des valeurs de `Index` correspondant à un élément qui ne figure pas dans la collection, provoquent la levée d'une exception.

— `Add(BSTR NotionName)` : méthode qui crée un nouvel élément de type `Notion` nommé d'après `NotionName`, et l'ajoute à la collection. Si un élément porte déjà ce nom, une exception est levée.

— `Remove(VARIANT Index)` : méthode qui supprime un élément donné de la collection. Les mêmes restrictions que dans `Item` s'appliquent au paramètre `Index`.

— `RemoveAll` : méthode qui supprime tous les éléments de la collection.

— `Paste` : méthode qui ajoute à la collection les données situées sur le Presse-papiers, à condition qu'il ne soit pas vide et qu'un élément de type `Notion` y figure. Sinon, une exception est levée.

— `IDomain* Father` : propriété en lecture seule qui renvoie une référence sur l'objet `Domain` qui contient cette collection.

5.2.1.7 La coclasse Notion

Cette coclasse présente un comportement similaire à celui de la coclasse `Domain`. Son interface `INotion` expose les éléments suivants :

— `[BSTR] Label([BSTR newVal])` : propriété en lecture / écriture qui permet de consulter ou modifier l'étiquette de ce nœud. En écriture, si un autre nœud porte déjà le nom spécifié par `newVal`, une exception est levée.

— `[pelActivity] Activity([pelActivity newVal])` : propriété en lecture / écriture qui permet de consulter ou modifier le taux d'activité de ce nœud.

— `ICViews* CViews` : propriété en lecture seule qui renvoie une référence sur la collection des vues contenue par ce nœud.

— `Copy` : méthode qui place sur le Presse-papiers une copie des informations contenues dans ce nœud.

— `Cut` : méthode similaire à `Copy` mais qui en outre supprime ce nœud (toute référence sur ce nœud devient donc invalide).

— `IConstraints* Constraints` : propriété en lecture seule qui renvoie une référence sur la liste des contraintes sémantiques existant entre les nœuds fils de ce nœud.

— `BSTR Path` : propriété en lecture seule qui affiche sous forme lisible le chemin allant de l'entrée jusqu'au nœud actuel, avec les étiquettes séparées par des points.

— `pelNodeType Type` : propriété en lecture seule qui renvoie un code identifiant le type du nœud courant, soit `pelNOTION` ici.

— `INotions* Father` : propriété en lecture seule qui renvoie une référence sur la collection qui contient ce nœud.

5.2.1.8 La coclasse CViews

La coclasse `CViews` présente un comportement similaire à celui de la coclasse `Domains`. Son interface `ICViews` expose les éléments suivants :

— `long Count` : propriété en lecture seule renvoyant le nombre d'éléments dans la collection.

— `ICView* Item(VARIANT Index)` : propriété en lecture seule qui renvoie une référence sur l'élément dont l'indice est spécifié par le paramètre `Index` qui peut être une chaîne de caractères (`BSTR`) ou un entier (long ou court). Tout autre type de paramètre, ainsi que des valeurs de `Index` correspondant à un élément qui ne figure pas dans la collection, provoquent la levée d'une exception.

— `Add(BSTR ViewName)` : méthode qui crée un nouvel élément de type `CView` nommé d'après `ViewName`, et l'ajoute à la collection. Si un élément porte déjà ce nom, une exception est levée.

— `Remove(VARIANT Index)` : méthode qui supprime un élément donné de la collection. Les mêmes restrictions que dans `Item` s'appliquent au paramètre `Index`.

— `RemoveAll` : méthode qui supprime tous les éléments de la collection.

— `Paste` : méthode qui ajoute à la collection les données situées sur le Presse-papiers, à condition qu'il ne soit pas vide et qu'un élément de type `CView` y figure. Sinon, une exception est levée.

— `INotion* Father` : propriété en lecture seule qui renvoie une référence sur l'objet `Notion` qui contient cette collection.

5.2.1.9 La coclasse CView

Cette coclasse présente un comportement similaire à celui de la coclasse `Domain`. Son interface `ICView` expose les éléments suivants :

— `[BSTR] Label([BSTR newVal])` : propriété en lecture / écriture qui permet de consulter ou modifier l'étiquette de ce nœud. En écriture, si un autre nœud porte déjà le nom spécifié par `newVal`, une exception est levée.

— `[pelActivity] Activity([pelActivity newVal])` : propriété en lecture / écriture qui permet de consulter ou modifier le taux d'activité de ce nœud.

— `IFeatures* Features` : propriété en lecture seule qui renvoie une référence sur la collection des traits contenue par ce nœud.

— `Copy` : méthode qui place sur le Presse-papiers une copie des informations contenues dans ce nœud.

— `Cut` : méthode similaire à `Copy` mais qui en outre supprime ce nœud (toute référence sur ce nœud devient donc invalide).

— `IConstraints* Constraints` : propriété en lecture seule qui renvoie une référence sur la liste des contraintes sémantiques existant entre les nœuds fils de ce nœud.

— `BSTR Path` : propriété en lecture seule qui affiche sous forme lisible le chemin allant de l'entrée jusqu'au nœud actuel, avec les étiquettes séparées par des points.

— `pelNodeType Type` : propriété en lecture seule qui renvoie un code identifiant le type du nœud courant, soit `pelCVIEW` ici.

— `ICViews* Father` : propriété en lecture seule qui renvoie une référence sur la collection qui contient ce nœud.

5.2.1.10 La coclasse Features

La coclasse `Features` présente un comportement similaire à celui de la coclasse `Domains`. Son interface `IFeatures` expose les éléments suivants :

— `long Count` : propriété en lecture seule renvoyant le nombre d'éléments dans la collection.

— `IFeature* Item(VARIANT Index)` : propriété en lecture seule qui renvoie une référence sur l'élément dont l'indice est spécifié par le paramètre `Index` qui peut être une chaîne de caractères (`BSTR`) ou un entier (long ou court). Tout autre type de paramètre, ainsi que des valeurs de `Index` correspondant à un élément qui ne figure pas dans la collection, provoquent la levée d'une exception.

— `Add(BSTR FeatureName)` : méthode qui crée un nouvel élément de type `Feature` nommé d'après `FeatureName`, et l'ajoute à la collection. Si un élément porte déjà ce nom, une exception est levée.

— `Remove(VARIANT Index)` : méthode qui supprime un élément donné de la collection. Les mêmes restrictions que dans `Item` s'appliquent au paramètre `Index`.

— `RemoveAll` : méthode qui supprime tous les éléments de la collection.

— `Paste` : méthode qui ajoute à la collection les données situées sur le Presse-papiers, à condition qu'il ne soit pas vide et qu'un élément de type `Feature` y figure. Sinon, une exception est levée.

— `ICView* Father` : propriété en lecture seule qui renvoie une référence sur l'objet `CView` qui contient cette collection.

5.2.1.11 La coclasse Feature

Cette coclasse présente un comportement similaire à celui de la coclasse `Domain`, mais elle possède des opérations en moins, étant donné qu'un trait ne peut avoir de fils. Son interface `IFeature` expose les éléments suivants :

— `[BSTR] Label([BSTR newVal])` : propriété en lecture / écriture qui permet de consulter ou modifier l'étiquette de ce nœud. En écriture, si un autre nœud porte déjà le nom spécifié par `newVal`, une exception est levée.

— `[pelActivity] Activity([pelActivity newVal])` : propriété en lecture / écriture qui permet de consulter ou modifier le taux d'activité de ce nœud.

— `Copy` : méthode qui place sur le Presse-papiers une copie des informations contenues dans ce nœud.

— `Cut` : méthode similaire à `Copy` mais qui en outre supprime ce nœud (toute référence sur ce nœud devient donc invalide).

— `BSTR Path` : propriété en lecture seule qui affiche sous forme lisible le chemin allant de l'entrée jusqu'au nœud actuel, avec les étiquettes séparées par des points.

— `pelNodeType Type` : propriété en lecture seule qui renvoie un code identifiant le type du nœud courant, soit `pelFEATURE` ici.

— `IFeatures* Father` : propriété en lecture seule qui renvoie une référence sur la collection qui contient ce nœud.

5.2.1.12 La coclasse Constraints

Il s'agit d'une collection destinée à contenir des objets de type `Constraint`. De même que les collections que nous avons précédemment, elle contient les opérations nécessaires à son énumération (`Count` et `Item`), à l'ajout d'élements (`Add`) et au retrait d'éléments (`Remove` et `RemoveAll`).

Son interface `IConstraints` expose les éléments suivants :

— `long Count` : propriété en lecture seule renvoyant le nombre d'éléments dans la collection.

— `IConstraint* Item(long Index)` : propriété en lecture seule qui renvoie une référence sur l'élément dont l'indice est spécifié par le paramètre numérique `Index`. Un indice hors limites provoque la levée d'une exception.

— `Add(BSTR InitialNode, pelConstraint Arrow, BSTR FinalNode)` : méthode qui crée un nouvelle contrainte sémantique du type spécifié par le paramètre `Arrow` entre les nœuds de la collection qui contient cette instance de `IConstraints` et dont les étiquettes sont spécifiées par `InitialNode` et `FinalNode`. Si de tels nœuds n'existent pas, ou bien s'il existe déjà une contrainte entre ces deux nœuds, une exception est levée.

— `Remove(long Index)` : méthode qui supprime un élément donné de la collection. Les mêmes restrictions que dans `Item` s'appliquent au paramètre `Index`.

— `RemoveAll` : méthode qui supprime tous les éléments de la collection.

Le type `pelConstraint` est un type énuméré servant à coder les différents types de contraintes sémantiques. Il est spécifié formellement au §D.1.6.

5.2.1.13 La coclasse Constraint

Cette coclasse permet de représenter les contraintes sémantiques présentes entre certains nœuds d'une même fratrie. Elle possède une interface des plus simples.

Spécification formelle : cf. annexe D.1.7

Interface IConstraint

— `[BSTR] InitialNode([BSTR newVal])` : propriété en lecture / écriture qui permet de consulter ou de modifier le nœud initial de la contrainte sémantique. En écriture, si le nœud spécifié par `newVal` n'existe pas, ou si la modification entraîne la création d'un conflit avec une autre contrainte sémantique, une exception est levée.

— `[pelConstraint] Arrow([pelConstraint newVal])` : propriété en lecture / écriture qui permet de consulter ou de modifier le type de la contrainte sémantique.

— `[BSTR] FinalNode([BSTR newVal])` : propriété en lecture / écriture qui permet de consulter ou de modifier le nœud final de la contrainte sémantique. Les mêmes restrictions sur `newVal` s'appliquent que pour `InitialNode`.

5.2.1.14 La coclasse Libraries

Cette coclasse représente la collection des bibliothèques de prédicats associées à une entrée lexicale (cf. §5.4). Elle possède les opérations classiques d'une collection, plus trois opérations dédiées au chargement et au déchargement des bibliothèques. Leur sémantique est si simple qu'elles ne justifient pas de présenter ici leur spécification formelle.

— `long Count` : propriété en lecture seule renvoyant le nombre d'éléments dans la collection.

— `ILibrary* Item(long Index)` : propriété en lecture seule qui renvoie une référence sur l'élément dont l'indice est spécifié par le paramètre numérique `Index`. Un indice hors limites provoque la levée d'une exception.

— `Add(BSTR LibPath)` : méthode qui crée un nouvelle association entre l'entrée lexicale et une bibliothèque de prédicats dont le chemin d'accès est spécifié par `LibPath`. Si la DLL ainsi désignée n'existe pas, ou si elle ne possède pas la signature (cf. §5.4) d'une bibliothèque LIGHT-PELEAS, une exception est levée.

— `Remove(long Index)` : méthode qui supprime un élément donné de la collection. Les mêmes restrictions que dans `Item` s'appliquent au paramètre `Index`.

— `RemoveAll` : méthode qui supprime tous les éléments de la collection.

— `Load` : méthode qui provoque le chargement et le *mappage* en mémoire de toutes les bibliothèques de la collection. Si les bibliothèques sont déjà chargées, cette méthode n'a aucun effet.

— `UnLoad` : méthode qui signale au système d'exploitation que les bibliothèques ne sont plus utilisées par cette instance de LIGHTPELEAS, et lui laisse le soin de les décharger si besoin est. Si les bibliothèques n'ont pas été préalablement chargées, cette méthode n'a aucun effet.

— `BOOL AreLoaded` : propriété en lecture seule qui indique si les bibliothèques sont chargées en mémoire.

5.2.1.15 La coclasse Library

Cette coclasse n'est disponible dans le modèle objet de LIGHTPELEAS qu'à titre indicatif, afin de préserver la cohérence du schéma qui a présidé à sa conception. En fait, il est peu probable que cette coclasse ait jamais à être utilisée directement. Elle se contente de fournir des indications techniques concernant le système d'exploitation, et un moyen de consulter les prédicats qui sont présents dans une bibliothèque.

L'interface `ILibrary` est composée des éléments suivants :

— `IPredicates* Predicates` : propriété en lecture seule qui renvoie une référence sur la collection des prédicats contenus dans cette bibliothèque.

— `BSTR Path` : propriété en lecture seule qui renvoie le chemin d'accès complet au fichier .DLL contenant cette bibliothèque.

— `long Instance` : propriété en lecture seule qui renvoie l'identificateur associé par WINDOWS 95/NT4 à l'instance de cette bibliothèque qui est utilisée par l'instance courante de LIGHTPELEAS (dans les langages qui disposent des types *ad hoc* pour WINDOWS 95/NT4, il faut transtyper la valeur de retour en `HINSTANCE`).

5.2.1.16 La coclasse Predicates

Cette collection ne comporte que les éléments nécessaires à son énumération car son rôle est purement consultatif : représenter la liste des prédicats contenus dans une bibliothèque.

L'interface `Predicates` est donc réduite à :

— `long Count` : propriété en lecture seule renvoyant le nombre d'éléments dans la collection.

— `IPredicate* Item(VARIANT Index)` : propriété en lecture seule qui renvoie une référence sur l'élément dont l'indice est spécifié. Le paramètre `Index` peut être soit une chaîne de caractères (`BSTR`) représentant le nom de l'élément, soit un entier (long ou court) représentant sa position dans la collection. Tout autre type lève une exception. S'il s'agit d'un paramètre numérique inférieur à 1 ou supérieur au nombre d'éléments de la collection, ou s'il s'agit d'un paramètre alphanumérique ne correspondant au nom d'aucun élément de la collection, une exception est également levée.

5.2.1.17 La coclasse Predicate

La seule opération qu'on peut effectuer sur un predicat est de consulter son nom. C'est pourquoi l'interface `IPredicate` est réduite à :

— `BSTR Name` : propriété en lecture seule qui renvoie le nom du prédicat.

5.2.1.18 La coclasse Rules

Cette collection représente l'ensemble des règles contextuelles attachées à une entrée lexicale. Elle possède une interface `IRules` similaire aux autres collections que nous avons vues jusqu'ici, avec des éléments de consultation et d'ajout/retrait.

— `long Count` : propriété en lecture seule renvoyant le nombre d'éléments dans la collection.

— `IRule* Item(VARIANT Index)` : propriété en lecture seule qui renvoie une référence sur l'élément dont l'indice est spécifié par le paramètre `Index` qui peut être une chaîne de caractères (`BSTR`) ou un entier (long ou court). Tout autre type de paramètre, ainsi que des valeurs de `Index` correspondant à un élément qui ne figure pas dans la collection, provoquent la levée d'une exception.

— `Add(BSTR Name)` : méthode qui crée un nouvel élément de type `Rule` nommé d'après `Name`, et l'ajoute à la collection. Si un élément porte déjà ce nom, une exception est levée.

— `Remove(VARIANT Index)` : méthode qui supprime un élément donné de la collection. Les mêmes restrictions que dans `Item` s'appliquent au paramètre `Index`.

— `RemoveAll` : méthode qui supprime tous les éléments de la collection.

— `Paste` : méthode qui ajoute à la collection les données situées sur le Presse-papiers, à condition qu'il ne soit pas vide et qu'un élément de type `Rule` y figure. Sinon, une exception est levée.

5.2.1.19 La coclasse Rule

Une règle contextuelle étant composée d'un nom, d'une prémisse et d'une liste d'activations, la coclasse `Rule` sert uniquement à encapsuler ces trois éléments dans un seul objet. Son interface `IRule` est donc naturellement composée de :

— `[BSTR] Name([BSTR newVal])` : propriété en lecture / écriture qui permet de consulter ou de modifier le nom d'une règle. En écriture, si une autre règle porte déjà le nom spécifié par le paramètre `newVal`, une exception est levée.

— `ICondition* Premise` : propriété en lecture seule qui renvoie une référence sur la formule logique constituant la prémisse de cette règle.

— `IActivations* Activations` : propriété en lecture seule qui renvoie une référence sur la liste des activations qui forment la conclusion de cette règle.

— `BSTR Text` : propriété en lecture seule qui renvoie dans une chaîne de caractères la version lisible de cette règle suivant le format

$$Nom : Premisse \Rightarrow \{Activations\}$$

— `Copy` : méthode qui place sur le Presse-papiers une copie des informations contenues dans cette règle.

— `Cut` : méthode similaire à `Copy` mais qui en outre supprime cette règle (toute référence sur cette règle devient donc invalide).

5.2.1.20 La coclasse Condition

La coclasse `Condition` représente une condition contextuelle, c'est-à-dire une formule logique à évaluer sur le contexte et qui est située en partie gauche d'une règle contextuelle.

Spécification formelle : cf. annexe D.1.8

Interface ICondition

— `BSTR Text` : propriété en lecture seule qui fournit dans une chaîne de caractères une version lisible de la formule qui constitue cette condition.

— `IStack* Stack` : propriété en lecture seule qui renvoie une référence sur la pile de construction de la condition (cf. §5.2.1.21).

— `ValidateTop` : méthode qui valide le sommet de la pile de construction en tant que contenu de cette condition.

— `pelCondType Type` : propriété en lecture seule qui renvoie un code correspondant au type de cette condition.

— `ICondition* LComponent` : propriété en lecture seule qui renvoie une référence sur

  – la partie gauche de la formule s'il s'agit d'une conjonction ou d'une disjonction ;
  – la sous-formule niée s'il s'agit d'une négation ;
  – une formule vide si cette formule est vide ou s'il s'agit d'un prédicat.

— `ICondition* RComponent` : propriété en lecture seule qui renvoie une référence sur

  – la partie droite de la formule s'il s'agit d'une conjonction ou d'une disjonction ;
  – une formule vide si cette formule est vide ou s'il s'agit d'un prédicat ou d'une négation.

— `ValidateFromClipboard` : méthode similaire à `ValidateTop`, mais qui valide le contenu du Presse-papiers plutôt que le sommet de la pile. Si le Presse-papiers est vide ou ne contient pas un élément de type `Condition`, une exception est levée.

— `Unvalidate` : méthode qui supprime la formule de cette condition (sans vider la pile).

— `MoveContentsToStack` : méthode qui transfère le contenu de la condition au sommet de sa pile (la formule est donc vide, et la pile contient un nouvel élément).

5.2.1.21 La coclasse Stack

La coclasse `Stack` a un fonctionnement assez spécial, puisque c'est une collection de `Condition`, et qu'une `Condition` possède une `Stack`. Cette organisation est nécessaire pour modéliser la récursivité de la structure syntaxique des formules logiques.

Spécification formelle : cf. annexe D.1.9

Interface IStack

— `long Size` : propriété en lecture seule qui renvoie le nombre d'éléments présents dans la pile.

— `ICondition* Item(long Index)` : propriété en lecture seule qui renvoie une référence sur l'élément de la pile d'indice spécifié. Le sommet de la pile est l'élément d'indice 1. Si la valeur de `Index` est inférieure à 1 ou supérieure à la taille de la pile, une exception est levée.

— `ICondition* Top` : propriété en lecture seule qui renvoie une référence sur l'élément au sommet de la pile (identique à `Item(1)`).

— `Push(pelCondType CondType, [BSTR Name])` : méthode qui ajoute un élément au sommet de la pile. Si `CondType` vaut :

  – `pelPREDICATE`, il faut fournir dans le paramètre `Name` le nom du prédicat à ajouter.
  – `pelAND`, les deux éléments au sommet de la pile sont remplacés par leur conjonction.
  – `pelOR`, les deux éléments au sommet de la pile sont remplacés par leur disjonction.
  – `pelNOT`, le sommet de la pile est remplacé par sa négation.

  Toute autre configuration provoque la levée d'une exception.

— `InsertFromClipboard(long Index)` : méthode qui insère dans la pile, à l'endroit spécifié par `Index`, une condition située sur le Presse-papiers. Si `Index` est hors limites, ou si le Presse-papiers est vide, ou bien ne contient pas de `Condition`, une exception est levée.

— `Pop` : méthode qui supprime le sommet de la pile. Si la pile est vide, une exception est levée.

— `PopAll` : méthode qui vide la pile.

— `Remove(long Index)` : méthode qui supprime une condition de la pile. Si `Index` est hors limites, une exception est levée.

— `UpInStack(long Index)` : méthode qui fait remonter d'un cran un élément de la pile. Cette méthode n'a aucun effet si `Index = 1`, et lève une exception si `Index` est hors limites.

— `DownInStack(long Index)` : méthode qui fait descendre d'un cran un élément de la pile. Cette méthode n'a aucun effet si `Index = Size`, et lève une exception si `Index` est hors limites.

— `Swap(long Index1, long Index2)` : méthode qui échange deux éléments dans la pile. Si `Index1` ou `Index2` sont hors limites, une exception est levée.

5.2.1.22 La coclasse Activations

Il s'agit d'une collection d'éléments de type `Activation`, qui joue le rôle de conclusion dans une règle contextuelle.

— `long Count` : propriété en lecture seule qui renvoie le nombre d'éléments dans la collection.

— `IActivation* Item(long Index)` : propriété en lecture seule qui renvoie une référence sur l'élément spécifié. Si `Index` est hors limites, une exception est levée.

— `Add(VARIANT NodePath, pelActivity Value)` : méthode qui crée une nouvelle `Activation` et l'ajoute à la collection. Le paramètre `NodePath` peut être soit une chaîne de caractères (`BSTR`) contenant un chemin composé des noms des nœuds qu'il faut traverser pour atteindre le nœud à activer (par exemple, `"domaine.notion.vue.trait"`), soit une référence sur le nœud à activer (donc de type `IDomain*`, `INotion*`, `ICView*` ou `IFeature*`). S'il existe déjà une activation du nœud spécifié par `NodePath` dans la collection, une exception est levée. De même si ce nœud n'existe pas.

— `Remove(long Index)` : méthode qui supprime une activation de la liste. Si `Index` est hors limites, une exception est levée.

— `RemoveAll` : méthode qui vide la collection.

5.2.1.23 La coclasse Activation

— `[BSTR] Node([VARIANT NodePath])` : propriété en lecture / écriture qui permet de consulter ou de modifier le nœud cible de l'activation. Les conditions sur `NodePath` sont les mêmes que pour `Activations.Add`. En lecture, la valeur renvoyée est toujours une chaîne de caractères exprimant le chemin d'accès au nœud de façon lisible.

— `[pelActivity] Value([pelActivity NewValue])` : propriété en lecture / écriture qui permet de consulter ou de modifier la valeur qui doit être attribuée au nœud lorsque l'activation est validée.

5.2.1.24 La coclasse Guesses

Il s'agit d'une collection d'éléments de type `Guess`, comportant les éléments d'interface habituels d'une collection non modifiable.

— `long Count` : propriété en lecture seule qui renvoie le nombre d'éléments dans la collection.

— `IGuess* Item(long Index)` : propriété en lecture seule qui renvoie une référence sur l'élément d'indice spécifié. Si l'indice est hors limites, une exception est levée.

5.2.1.25 La coclasse Guess

Un élément de type `Guess` représente une proposition d'interprétation de la part de LIGHTPELEAS, c'est-à-dire une vue conceptuelle fortement activée, accompagnée d'une estimation numérique de sa participation dans la construction du sens de l'entrée lexicale en contexte.

— `ICView* Candidate` : propriété en lecture seule qui renvoie une référence sur la vue conceptuelle candidate à l'interprétation.

— `double Hint` : propriété en lecture seule qui renvoie une estimation numérique de sa participation à la construction du sens de l'entrée lexicale. Il s'agit de l'estimateur défini §4.3.4.

5.2.2 Format de stockage des entrées lexicales

5.2.2.1 Organisation des enregistrements

Chaque entrée lexicale est enregistrée dans un fichier séparé, dont le chemin est défini lors de l'appel de la méthode `Entry.Save(FileName)`. Les données qui constituent l'entrée lexicale y sont enregistrées au format PDL, que nous décrivons à la sous-section suivante. Notons qu'un fichier au format PDL doit impérativement commencer par une ligne de signature contenant uniquement `%%PDL`.

De plus, les entrées lexicales sont regroupées en *lexiques*. Un lexique correspond en fait à un fichier d'index contenant une liste d'entrées lexicales, ainsi que les noms des fichiers dans lesquels elles sont enregistrées. Bien entendu, une même entrée lexicale peut figurer dans plusieurs lexiques. Un lexique est engendré par la méthode `Engine.GenerateIndex`.

Les fichiers de toutes les entrées lexicales constituant un lexique doivent être rangés dans un seul répertoire du système de fichiers. C'est le chemin de ce répertoire qui est consulté / modifié par la propriété `Engine.EntriesDirectory`.

5.2.2.2 Le format PDL

Nous avons choisi d'utiliser un format de stockage des données sous forme de fichiers texte directement modifiables depuis n'importe quel éditeur de texte afin de permettre la constitution des lexiques même en l'absence de l'éditeur graphique MELISANDE (cf. §5.3).

A l'intérieur d'un tel fichier, les données constituant une entrée lexicale sont écrites en respectant un format que nous avons nommé PDL (pour *Peleas Descriptive Language*). Il s'agit d'un langage relativement simple, dont nous donnons la grammaire ci-dessous[1] :

Entrée → ENTRY(*Nœud*) = BEGIN
ListeDomaines
LIBRARY
ListeBibli
RULES
ListeRègles
END.

Nœud → (**lettre_**)$^+$

ListeDomaines → *Domaine* | *Domaine ListeDomaines*

Domaine → GHOST ;

[1] **lettre** indique n'importe quelle lettre, majuscule, minuscule et/ou accentuée, et **chemin MS-DOS** représente un nom de fichier absolu au format MS-DOS, c'est-à-dire \ comme séparateur de répertoires.

		| *Nœud* ;
		| *Nœud* =
		Nœud ;
		| *Nœud* = BEGIN
		ListeNotions
		END ;
ListeNotions	→	*Notion* | *Notion ListeNotions*
Notion	→	GHOST ;
		| *Nœud* ;
		| *Nœud* =
		Nœud ;
		| *Nœud* = BEGIN
		ListeVues
		END ;
ListeVues	→	*Vue* | *Vue ListeVues*
Vue	→	GHOST ;
		| *Nœud* ;
		| *Nœud* =
		Nœud ;
		| *Nœud* = BEGIN
		ListeTraits
		END ;
ListeTraits	→	*Trait* | *Trait ListeTraits*
Trait	→	*Noeud* ;
ListeBibli	→	*Bibli* | *Bibli ListeBibli*
Bibli	→	**chemin MS-DOS** ;
ListeRègles	→	*Règle* | *Règle ListeRègles*
Règle	→	*NomRègle* : *Formule* => {
		ListeActiv
		}
NomRègle	→	$(\textbf{lettre}\|[0..9])^+$
Formule	→	*NomPrédicat* | NOT (*Formule*)
		(*Formule*) | F_2
F_2	→	*Formule* AND *Formule* | (F_2) AND (F_2)
		Formule AND (F_2) | (F_2) AND *Formule*
		Formule OR *Formule* | (F_2) OR (F_2)
		Formule OR (F_2) | (F_2) OR *Formule*

ListeActiv → *Activation* | *Activation ListeActiv*

Activation → (*CheminNœud Activité*)

CheminNœud → *Nœud*
| *Nœud.Nœud*
| *Nœud.Nœud.Nœud*
| *Nœud.Nœud.Nœud.Nœud*

Activité → Salient | Valid | Ignored | Negated

Bien entendu, LIGHTPELEAS est équipé d'un interpréteur pour le langage PDL, qui lui permet de construire les structures correspondant à un fichier. Cet interpréteur étant LALR prédictif, LIGHTPELEAS est capable de fournir des messages à l'utilisateur pour indiquer les éventuelles erreurs de syntaxe des fichiers PDL, et comment les résoudre *a priori*.

Pour un exemple de fichier au format PDL, se reporter aux entrées lexicales figurant en annexe (cf. §A.1).

5.3 L'éditeur graphique MELISANDE

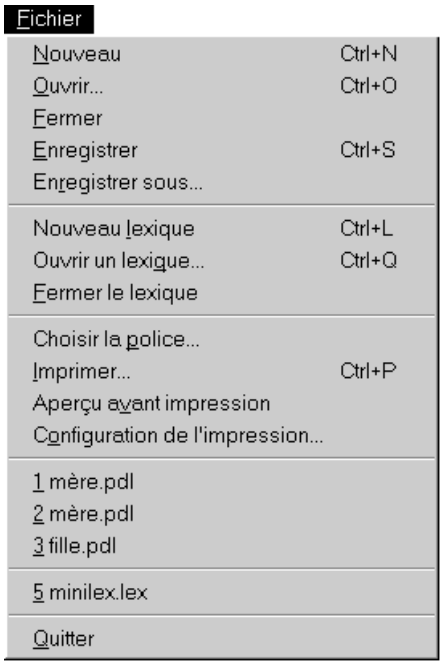
Comme nous l'avons dit plus haut, il est possible de constituer un lexique manuellement en éditant les fichiers texte contenant les entrées lexicales, ou bien en utilisant directement le modèle objet LIGHTPELEAS. Cependant, il nous a semblé plus pratique et plus agréable de disposer d'un éditeur interactif de lexiques PELEAS. C'est le rôle du logiciel MELISANDE, qui est une application 32 bits pour WINDOWS 95/NT4.

DONNÉES TECHNIQUES	
LIGNES DE CODE :	**8.057 lignes**
TAILLE DE L'EXÉCUTABLE :	**414 Ko**
EMPREINTE MÉMOIRE MAXIMALE :	**1,5 Mo**

MELISANDE ne constitue en fait qu'une interface graphique au-dessus de LIGHTPELEAS. Une instance de MELISANDE charge donc toujours LIGHTPELEAS dans son espace mémoire préalablement à son exécution. Les données techniques ci-dessus sont donc à lire ainsi : l'empreinte mémoire maximale de MELISANDE est de 3,5 Mo *y compris* LIGHTPELEAS.

5.3.1 Gestion des fichiers

La gestion des fichiers (lexiques et entrées lexicales) sous MELISANDE se fait grâce au menu Fichier donné ci-après.



On reconnaît les opérations habituelles de création, d'ouverture et de fermeture de fichiers, ainsi que les commandes liées à l'impression. On remarquera cependant que la gestion des fichiers peut se faire à deux niveaux simultanément :

— la première section du menu concerne les entrées lexicales considérées comme des fichiers individuels (une entrée lexicale pour un fichier au format PDL) ;

— la deuxième section concerne les "lexiques" au sens de LIGHTPELEAS, c'est-à-dire des collections d'entrées lexicales. Techniquement, un lexique se ramène à un fichier d'index tel qu'engendré par la méthode `Engine.GenerateIndex` de LIGHTPELEAS (cf. §5.2.1.1).

Du fait de cette double gestion des fichiers, ce menu dispose de deux sections séparées pour le rappel des derniers fichiers chargés : la quatrième section permet de rappeler les dernières entrées lexicales, et la cinquième section permet de rappeler les derniers lexiques.

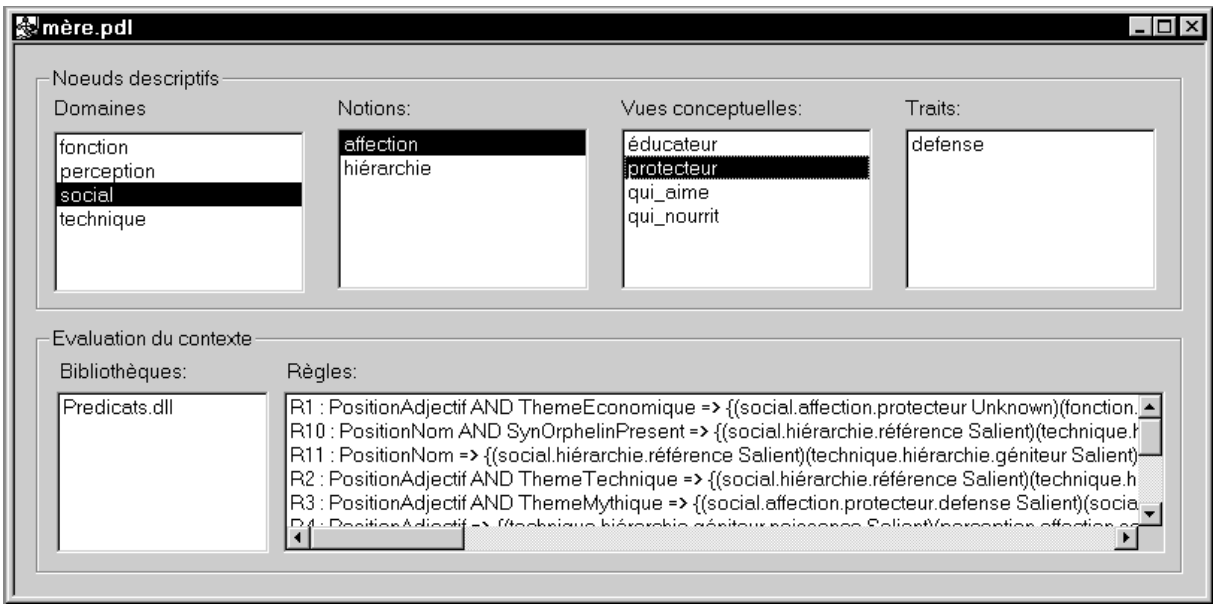
Cette organisation permet de gérer les entrées lexicales indépendamment de tout lexique. En fait, LIGHTPELEAS utilise alors un fichier d'index par défaut nommé `Peleas.lex` et situé dans le répertoire contenant MELISANDE.

L'impression d'une entrée lexicale produit un document contenant la description de cette entrée au format PDL, agrémentée de quelques enrichissements graphiques.

Il est possible de changer la police de caractères par défaut pour l'impression grâce à la commande Choisir la police.... Cependant, la police par défaut est relativement bien adaptée.

5.3.2 Edition d'une entrée lexicale

Chaque entrée lexicale est éditée au sein d'une fenêtre séparée :

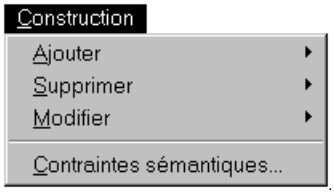


Cette fenêtre contient de boîte de listes permettant d'accéder aux différents nœuds constituant la structure descriptive d'une entrée lexicale.

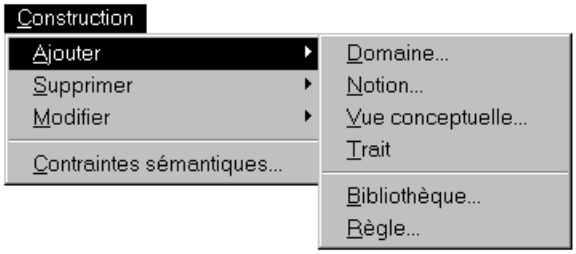
En ce qui concerne les éléments de la zone Nœuds descriptifs, les nœuds appartenant à un niveau donné n'apparaissent dans la boîte de listes correspondante que lorsque leur nœud père a été sélectionné dans la boîte située immédiatement à gauche. Ainsi, sélectionner une notion fait apparaître la liste des vues de cette notion dans la boîte Vues conceptuelles, mais tant qu'aucune vue n'a été sélectionnée, la boîte Traits reste vide.

5.3.2.1 Commandes d'édition

La gestion des éléments constitutifs d'une entrée lexicale se fait grâce au menu Construction :



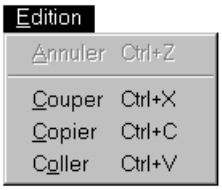
Ce menu permet d'ajouter, de modifier ou de supprimer un élément de la structure descriptive. Pour effectuer une de ces actions, il faut dérouler un des trois sous-menus correspondant, puis sélectionner un type d'élément :



Seuls les types d'éléments cohérents avec l'état de la fenêtre d'édition sont accessibles : ainsi, si aucune vue n'est sélectionnée, le type Vue conceptuelle ne sera pas accessible dans le sous-menu Supprimer.

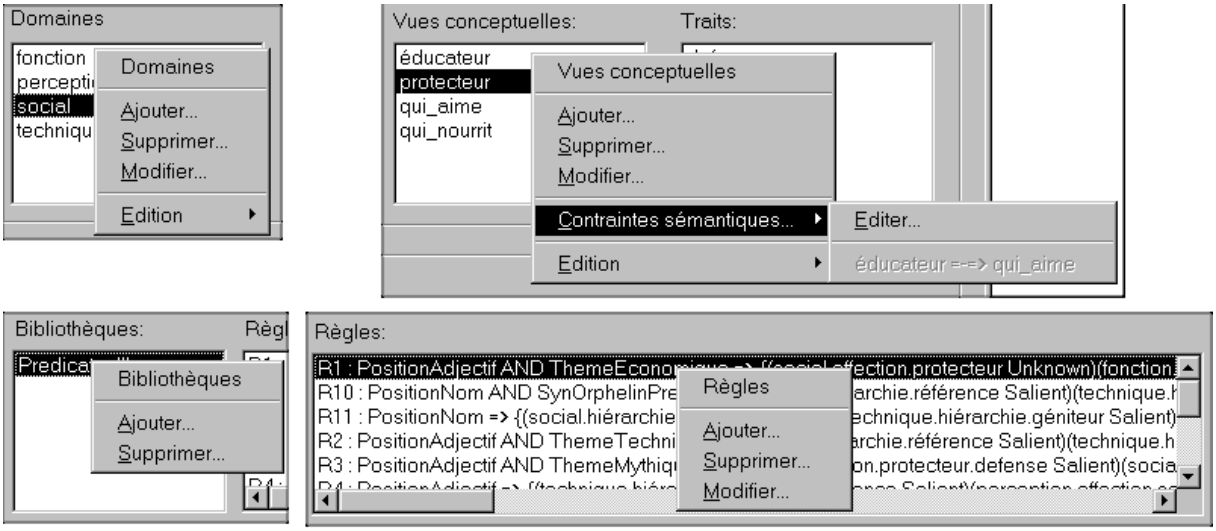
Le menu Construction permet aussi d'atteindre la boîte de dialogue d'édition des contraintes sémantiques pour une liste de vues conceptuelles ou une liste de traits. Pour que l'option Contraintes sémantiques soit accessible, il faut que le nœud père de cette liste ait été préalablement sélectionné (i.e. une notion ou une vue respectivement).

Le menu Edition permet d'utiliser les commandes habituelles du Presse-papiers : couper, copier, coller.



Tout nœud descriptif (domaine, notion, vue ou trait) peut être copié ou coupé. Pour le coller, il faut sélectionner un nœud de niveau supérieur (ou n'importe quel autre domaine pour un domaine) et activer la commande Coller.

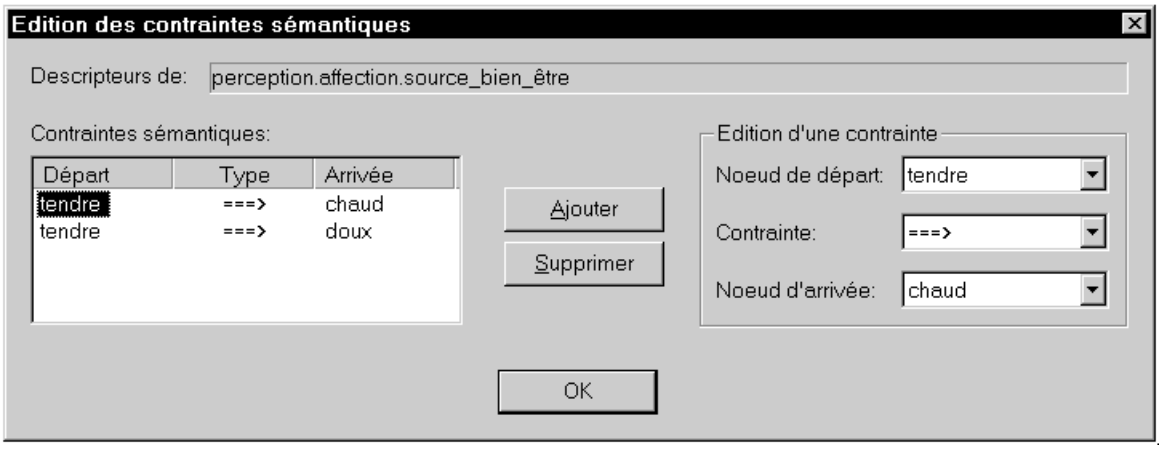
L'ensemble des commandes des menus Edition et Construction sont accessibles par des *menus contextuels*, c'est-à-dire des menus auto-adaptatifs que l'on déroule par un clic droit sur un élément graphique particulier. On peut ainsi obtenir quatre types de menus contextuels :



Ces menus contextuels, tout comme les commandes des menus Edition et Construction, assurent la cohérence entre les commandes disponibles et l'état de la fenêtre d'édition.

5.3.2.2 Contraintes sémantiques

Sélectionner la commande Contraintes sémantiques pour une vue ou un trait provoque l'ouverture de la fenêtre d'édition des contraintes :



Pour ajouter une nouvelle contrainte sémantique, il faut choisir un nœud de départ, un type de contrainte et un nœud d'arrivée dans la zone Edition d'une contrainte, puis cliquer sur le bouton Ajouter.

Pour supprimer une contrainte existante, il faut sélectionner son nœud de départ dans la zone Contraintes sémantiques, puis cliquer sur le bouton Supprimer

Pour modifier une contrainte, il faut la sélectionner comme précédemment. Ses caractéristiques s'affichent dans la zone Edition d'une contrainte afin de les modifier. Il faut alors supprimer la contrainte avant d'ajouter les nouvelles caractéristiques.

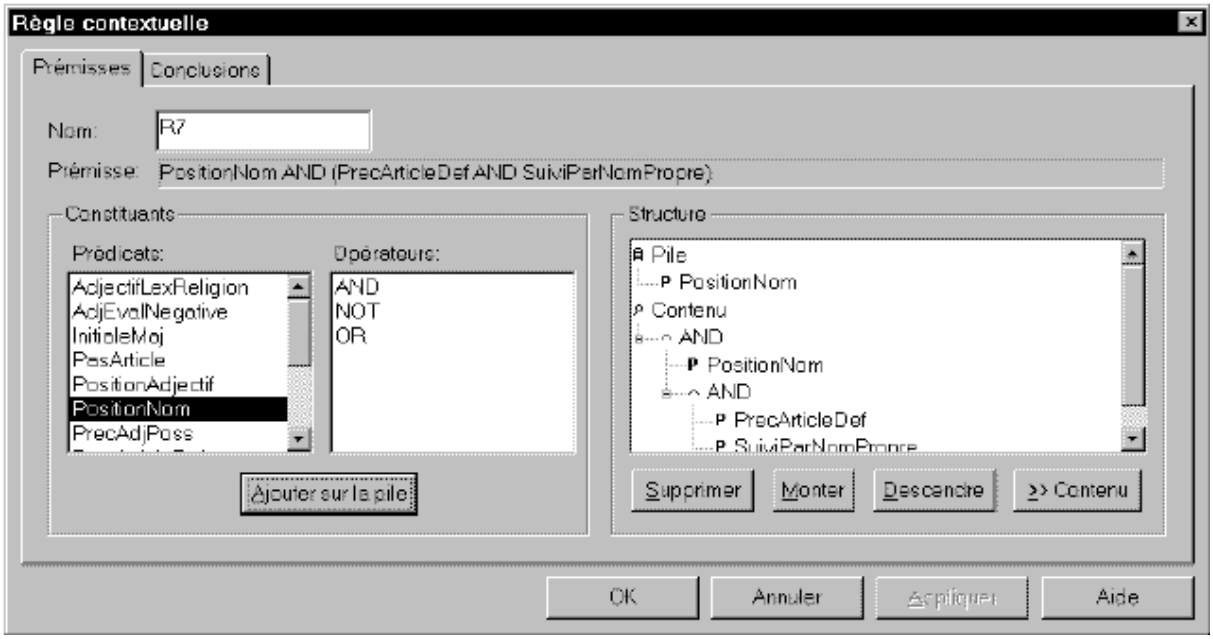
5.3.2.3 Règles contextuelles

L'ajout et la modification d'une règle contextuelle se font par l'intermédiaire de la même boîte de dialogue (dans le cas de l'ajout, il n'y a aucune donnée lors de l'ouverture de la boîte).

L'édition d'une règle contextuelle se fait en deux temps, correspondant à deux onglets dans la boîte d'édition :

1. édition de la partie gauche de la règle (onglet Prémisses) ;
2. puis édition de la partie droite (onglet Conclusions).

Edition des prémisses

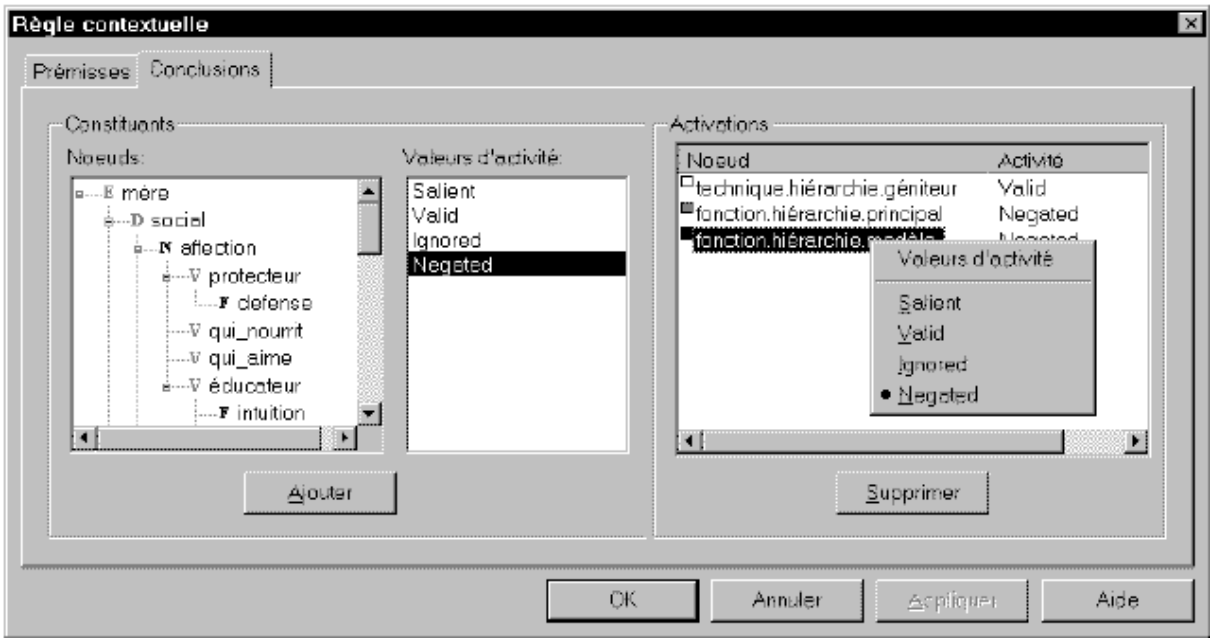


La construction d'une formule suit un procédé de notation polonaise inverse. Ainsi, pour construire la formule $a \wedge b$, il faut sélectionner le prédicat b et l'ajouter sur la pile, puis faire de même pour le prédicat a. On sélectionne ensuite l'opérateur AND qu'on ajoute sur la pile. Le contenu de la pile est dépilé deux fois, et la conjonction des deux éléments dépilés est empilée. La pile, qui contenait donc $(abx\ldots)$ contiendra alors $((a \wedge b)x\ldots)$.

Lorsque la pile contient la formule finale (quelle que soit sa profondeur), il faut ensuite la valider en tant que contenu de la prémisse grâce au bouton » Contenu. Lorsque le contenu n'est pas vide, une version linéaire de la formule est affichée dans le champ Prémisse.

Une règle doit impérativement avoir reçu un nom et une prémisse avant de pouvoir passer à l'onglet Conclusions.

Edition des conclusions



Pour ajouter une nouvelle activation dans la partie droite de la règle, il faut tout d'abord sélectionner une valeur d'activité dans la liste Valeurs d'activité, puis un nœud de la structure descriptive dans la liste Nœuds, et enfin cliquer sur le bouton Ajouter.

Pour changer la valeur d'activité d'une activation déjà existante, on fait apparaître un menu contextuel par un clic droit sur l'activation souhaitée dans la zone Activations. Il ne reste alors plus qu'à choisir la nouvelle valeur d'activité dans ce menu.

5.4 Développement d'une bibliothèque de prédicats

Pour l'évaluation des règles contextuelles attachées à une entrée lexicale, le programme LIGHTPELEAS a recours à des prédicats qui le renseignent sur l'état des représentations internes construites par le système. Chaque règle contextuelle comporte une partie gauche, appelée prémisse, qui doit être évaluée à VRAI pour que la règle soit déclenchée. Une prémisse est constituée de noms de prédicats reliés par des connecteurs logiques : AND, OR et NOT. L'évaluation de la prémisse implique donc qu'on puisse connaître d'abord la valeur de vérité de chaque prédicat.

5.4.1 Format des prédicats

Les prédicats utilisés par LIGHTPELEAS sont normalisés. Ils doivent tous adopter le format le plus simple qui soit : ils ne prennent aucun paramètre en entrée et fournissent en sortie une valeur appartenant, bien entendu, à l'ensemble $\{\text{VRAI}, \text{FAUX}\}$. Cette absence de paramètre est un choix qui résulte de notre souci d'indépendance de LIGHTPELEAS vis-à-vis du système de traitement automatique des langues naturelles avec lequel il interagit. En effet, autoriser des paramètres en entrée d'un prédicat signifie que le contenu du prédicat n'est pas suffisant en lui-même pour produire l'évaluation du contexte qu'il est sensé fournir. La présence d'un paramètre indique que LIGHTPELEAS fournit une information supplémentaire dont le prédicat a besoin pour être évalué : soit cette information concerne l'état d'une représentation interne de LIGHTPELEAS, soit elle concerne l'état d'une représentation interne du système de

traitement automatique des langues naturelles. Dans toute cette partie, nous noterons P-représentation une représentation interne de LIGHTPELEAS, c'est-à-dire un potentiel sémantique quel que soit l'état d'avancement de son évaluation. Nous noterons également S-représentation une représentation interne du système hôte de LIGHTPELEAS : une S-représentation peut-être une représentation morphologique, syntaxique, sémantique ou pragmatique (un schéma, un modèle de personnage...)

On peut tout de suite écarter l'option d'un paramètre permettant à LIGHTPELEAS de renseigner un prédicat sur l'état d'une P-représentation. En effet, LIGHTPELEAS est actuellement conçu comme un outil prenant en entrée un contexte sous forme de S-représentations, et fournissant en sortie la liste des noms des S-représentations sémantiques qui sont des candidats valides comme résultat de l'interprétation d'une expression polysémique. Le contexte sur lequel LIGHTPELEAS réalise ses inférences ne contient donc aucune P-représentation : par conséquent, il n'y a pas lieu que LIGHTPELEAS transmette des informations sur ses P-représentations à l'un de ses prédicats.

De plus, la consultation des S-représentations contextuelles n'est réalisée qu'à l'intérieur des prédicats qui composent les prémisses des règles contextuelles. Les S-représentations ne sont pas connues à l'extérieur des prédicats, et donc LIGHTPELEAS ne peut pas non plus transmettre d'information sur une S-représentation à un prédicat. C'est cette étanchéité entre les S-représentations et les P-représentations qui garantit l'indépendance de LIGHTPELEAS par rapport à son système de traitement automatique des langues naturelles hôte.

Puisque le contexte ne doit pas contenir de P-représentation, et que LIGHTPELEAS ne manipule aucune S-représentation, il ne peut y avoir d'échange d'information depuis LIGHTPELEAS vers un prédicat. Cela signifie donc que chaque prédicat est autosuffisant pour fournir l'information qu'on lui demande. Un corollaire est alors que les prédicats sont capables de demander au système de traitement automatique des langues naturelles de leur fournir des S-représentations, et que le format interne de ces représentations est connu par les prédicats.

5.4.2 Mise en œuvre des prédicats

Le choix d'une mise en œuvre des prédicats est conditionné par deux contraintes. La contrainte principale est, étant donné ce qui vient d'être dit, que le code exécutable des prédicats ne doit pas faire partie de l'application LIGHTPELEAS. Ceci provient de la nécessaire indépendance de LIGHTPELEAS avec son système hôte. La deuxième contrainte est moins importante quant au choix d'une mise ne œuvre : le module contenant le code exécutable des prédicats ne fait pas nécessairement partie du système de traitement automatique des langues naturelles hôte, mais doit cependant être capable de connaître ses S-représentations.

La première contrainte nous a conduit à envisager la mise en œuvre des prédicats comme un module exécutable indépendant de LIGHTPELEAS. Ils doivent cependant interagir en ceci que LIGHTPELEAS doit pouvoir accéder au code d'un prédicat de ce module, et surtout lancer son exécution. Les systèmes d'exploitation WINDOWS 95/NT4 proposent quatre moyens pour réaliser ce genre d'interactions :

— L'utilisation d'une bibliothèque de liens dynamiques, ou DLL (cf. annexe C) contenant du code exécutable qui peut être invoqué par une application.

— L'établissement d'une relation client/serveur entre deux applications par l'intermédiaire du protocole standard d'échange dynamique des données DDE (*Dynamic Data Exchange*).

— La publication de fonctions par une application en utilisant le service Automation (cf. §B.2).

— Et enfin, la publication par une application d'une classe d'objets, au sens de la Programmation Orientée Objets, en ayant recours à l'ensemble des services d'ActiveX.

Nous avons retenu la première solution, qui nous a semblé la plus simple à mettre en œuvre et la plus économique en terme de ressources du système d'exploitation. En effet, le protocole DDE ne permet d'échanger que des données brutes ; la publication de code exécutable par ce moyen impose donc aux programmeurs de gérer eux-mêmes la conversion entre les prédicats sous forme de données et une forme qui soit du code exécutable. Les deux dernières solutions, qui font appel à la technologie ActiveX, imposent des contraintes de programmation assez lourdes alors que, dans le cas qui nous intéresse, une DLL peut rendre exactement les mêmes services avec un nombre de contraintes considérablement plus réduit.

Par conséquent, nous proposons de mettre en œuvre les prédicats sous forme de liens dynamiques contenus dans des DLL. En pratique, les prédicats se présenteront comme des fonctions exécutables réunies dans des modules DLL, qui tiendront lieu de "bibliothèques de prédicats".

5.4.3 Spécificités d'une DLL "bibliothèque de prédicats"

Les spécificités d'une DLL mettant en œuvre une bibliothèque de prédicats telle que LIGHTPELEAS le conçoit portent sur deux points :

— Elle doit comporter un mécanisme particulier de signature afin de la différencier des autres DLL utilisées par WINDOWS 95/NT4.

— Elle doit permettre de renseigner ses clients sur le nombre et les noms des prédicats qu'elle contient, ce que ne permettent pas les DLL classiques.

Pour répondre à ces deux exigences simultanément, nous avons choisi de définir un jeu de fonctions qu'une DLL bibliothèque de prédicats PELEAS doit obligatoirement exporter, de façon similaire au point d'entrée (cf. annexe C). La différence avec le point d'entrée d'une DLL est que si le concepteur de la bibliothèque de prédicats omet ces fonctions, il n'existe aucun mécanisme pour les engendrer automatiquement (puisqu'elles ne font pas partie du standard WINDOWS 95/NT4), et la DLL n'est pas reconnue par LIGHTPELEAS.

Ces fonctions sont au nombre de deux :

1. la fonction `CountPredicates` renvoie le nombre de prédicats contenu dans la bibliothèque ;
2. la fonction `PredicateName` permet d'obtenir le nom d'un prédicat à partir d'un index compris entre 0 et le nombre de prédicats moins un.

Ces fonctions ont pour profils :

— `UINT CountPredicates()` qui renvoie le nombre de prédicats sous forme d'un entier non signé ;

— `BOOL PredicateName(UINT nIndex, LPSTR szBuffer, UINT nBufSize)` qui stocke dans `szBuffer` le nom du prédicat numéro `nIndex`. Ce buffer doit avoir été alloué préalablement à l'appel et doit donc être désalloué par l'appelant. Elle renvoie `TRUE` si tout s'est bien passé, et `FALSE` sinon.

Ces deux fonctions doivent être exportées par une DLL pour que LIGHTPELEAS la reconnaisse en tant que bibliothèques de prédicats, et forment ainsi une signature.

5.4.4 Format des prédicats dans la DLL

Eu égard à ce que nous avons dit au §5.4.1, les prédicats exportés par une DLL bibliothèque de prédicats ne prennent aucun argument et renvoient une valeur booléenne.

Toutes les fonctions exportées par la DLL, hormis son point d'entrée et les fonctions de signature présentées au paragraphe précédent, doivent donc avoir le profil : `BOOL Predicat();`

Afin de rester cohérent au niveau de l'énumération des noms de prédicats contenus dans la bibliothèque, tous les prédicats exportés par la DLL doivent être accessibles par la fonction `PredicateName`, et rien qu'eux (c'est-à-dire que les fonctions `CountPredicates` et `PredicateName` ne sont pas énumérées).

5.5 L'analyseur de corpus BARD

La constitution d'un lexique pour le modèle PELEAS est une tâche qui demeure longue et fastidieuse, même en utilisant l'éditeur graphique MELISANDE (cf. §5.3), car elle nécessite un travail intense d'expertise linguistique.

Nous avons décrit au §4.4 une méthode permettant l'extraction automatique de données à partir d'un corpus large afin d'alimenter le potentiel sémantique d'une entrée lexicale.

Malheureusement, n'ayant pas eu à notre disposition pendant la mise en œuvre de PELEAS les outils électroniques de représentation des connaissances requis par cet algorithme (il n'existe pas de dictionnaire électronique hiérarchique pour le français comparable, par exemple, au LDOCE[1], ni de thésaurus électronique en français, ni de base de relations lexicales sémantiques), nous n'avons pu mettre en œuvre qu'une partie de cet algorithme.

Le logiciel BARD décrit ci-après correspond à la mise en œuvre des étapes 1 et 2 de l'algorithme (respectivement le découpage du corpus en sous-corpus, et l'extraction d'étiquettes lexicales pour les *équivalents* du mot cherché), i.e. les seules programmables directement à partir de corpus, sans avoir recours à des bases de connaissances supplémentaires.

5.5.1 Hypothèses utilisées

On part ici du principe que le sens d'un mot dépend de son contexte linguistique et de sa mise en situation. L'analyse statistique du lexique d'un corpus nécessite donc la constitution d'un sous-corpus de travail (i.e. une suite de contextes pertinents) en éliminant systématiquement certaines parties des textes composant le corpus de départ.

5.5.1.1 Sélection des contextes

La sélection du sous-corpus de travail se fait habituellement de deux façons :

— par *fenêtrage*, i.e. en sélectionnant des tranches d'étendue constante centrées autour d'un mot pôle ;

— ou par *fragmentation* des textes, en sélectionnant le contenu d'unités structurelles précises, d'étendues inégales (chapitres, paragraphes, etc.)

[1] Longman's Dictionary of Contemporary English.

Afin de faire intervenir la notion d'isotopie locale dans la sélection des contextes pertinents, nous avons décidé de nous baser sur les observations de Rastier [99] : la propagation des traits sémantiques s'amenuise lors du franchissement des frontières d'une phrase, et se perpétue rarement au-delà des frontières de paragraphe. C'est pourquoi notre critère de sélection d'un contexte pertinent est à mi-chemin entre les deux critères cités ci-dessus : pour une occurrence d'un mot pôle donné, le contexte pertinent est constitué de la phrase qui contient cette occurrence, ainsi que de la phrase précédente et de la phrase suivante *modulo les frontières de paragraphe* : ainsi, si la phrase contenant l'occurrence du mot pôle est la première de son paragraphe, le contexte pertinent ne contient que cette phrase et la phrase suivante.

Au terme de cette sélection, on dispose du sous-corpus de travail. On peut alors comptabiliser les fréquences d'apparition de chaque forme fléchie dans le sous-corpus de travail, à l'exception éventuelle de certaines formes particulières marquées comme non pertinentes *a priori* (dans notre cas, il s'agira la plupart du temps des articles, pronoms, conjonctions, certains adverbes et adjectifs...)

5.5.1.2 Lemmatisation

Puisque nous nous intéressons ici à des équivalences sémantiques, nous souhaitons faire abstraction des informations contenues dans la morphologie des mots figurant dans le sous-corpus de travail. C'est pourquoi nous procédons maintenant à une phase de *lemmatisation* des formes comptabilisées. Cela signifie qu'on va rassembler les formes fléchies en les groupant par *lemmes* (formes non fléchies, ou canoniques). De ce fait, on comptabilisera plutôt les fréquences d'apparition des formes canoniques dans le sous-corpus de travail.

Cette lemmatisation sera assurée par le contrôle ActiveX AMORPH (cf. §5.6). Etant donné la simplicité de cet outil, on peut rencontrer deux problèmes :

— la forme fléchie peut-être inconnue (ne figure pas dans le dictionnaire utilisé par AMORPH) ;

— une même forme fléchie peut correspondre à plusieurs lemmes.

Or, si l'on interrompt la lemmatisation à chaque fois qu'un de ces deux problèmes survient pour demander à l'utilisateur de trancher, le temps consacré à la lemmatisation devient prohibitif.

C'est pourquoi nous avons décidé d'introduire les deux règles suivantes :

1. Toute forme fléchie inconnue sera considérée comme étant sa propre forme canonique.
2. Toute forme canonique correspondant à une forme fléchie sera comptabilisée autant de fois que la forme fléchie.

La première règle risque d'introduire du silence : différentes formes fléchies d'un même lemme inconnu seront comptabilisées séparément, et leur score d'écart réduit pourra passer en dessous du seuil de pertinence. Un lemme potentiellement intéressant risquera donc de ne pas être pris en compte au final.

Parallèlement, la deuxième règle peut introduire du bruit : si une forme fléchie correspond à plusieurs lemmes, ils seront tous comptabilisés et par conséquent, des lemmes qui ne sont pas (ou peu) présents dans le corpus risquent d'être pris en compte.

Cependant, en l'état actuel des choses, la constitution d'une entrée lexicale PELEAS repose encore en majeure partie sur l'intervention humaine, l'expertise linguistique. Nous considérerons donc que ces perturbations pourront être rattrapées lors du travail d'expertise ultérieur, et nous avons préféré privilégier la rapidité de traitement des corpus.

5.5.1.3 Calcul du score d'écart réduit

La détermination d'un équivalent potentiel se fonde sur l'étude de la répartition des mots présents dans un corpus d'étendue N (en nombre de mots). Il s'agit d'un test de probabilité dont l'hypothèse nulle est une répartition aléatoire des occurrences de ce mot (selon une loi normale hypergéométrique). Il y a alors lieu de rejeter l'hypothèse nulle lorsque la répartition observée pour un mot s'écarte de manière significative du modèle, c'est-à-dire :

— lorsque ce mot est irrégulièrement réparti, avec une fréquence anormalement haute ou basse ;

— lorsque ce mot est réparti de façon anormalement régulière.

Dans le cadre de l'hypothèse nulle, la répartition d'un mot ne doit pas être affectée par l'échantillon de corpus sur lequel il est observé : il devrait donc y avoir la même fréquence d'apparition de ce mot sur le corpus de référence que sur le sous-corpus de travail.

Ainsi, si nous notons $p = n/N$ le rapport d'étendue entre le sous-corpus de travail et le corpus de référence, et $f(m)$ la fréquence d'apparition du mot m dans le corpus de référence, alors la fréquence théorique d'apparition de m dans le sous-corpus de travail est

$$f_m = f(m) \times p$$

Notons maintenant f^*_m la fréquence observée de m dans le sous-corpus de travail. Puisque nous avons supposé que les mots ont une fréquence d'apparition qui suit une loi hypergéométrique, tout écart entre f_m et f^*_m inférieur à l'écart type sera considéré comme non significatif. Rappelons ici que l'écart type σ se mesure par rapport à la probabilité p d'apparition de m grâce à la variance $\sigma^2 = f(m) \times p \times (1-p)$.

Nous allons donc *réduire* l'écart entre fréquence observée et fréquence théorique par rapport à l'écart type pour calculer le score d'écart réduit

$$r = \frac{f^*_m - f_m}{\sqrt{f_m \times (1-p)}}$$

Grâce à une table gaussienne des percentiles, on pourra considérer qu'un score d'écart réduit $r \geq 3$ est significatif, puisqu'il correspond à un risque d'erreur inférieur à 2,7‰ à rejeter l'hypothèse nulle.

5.5.2 Le prototype BARD

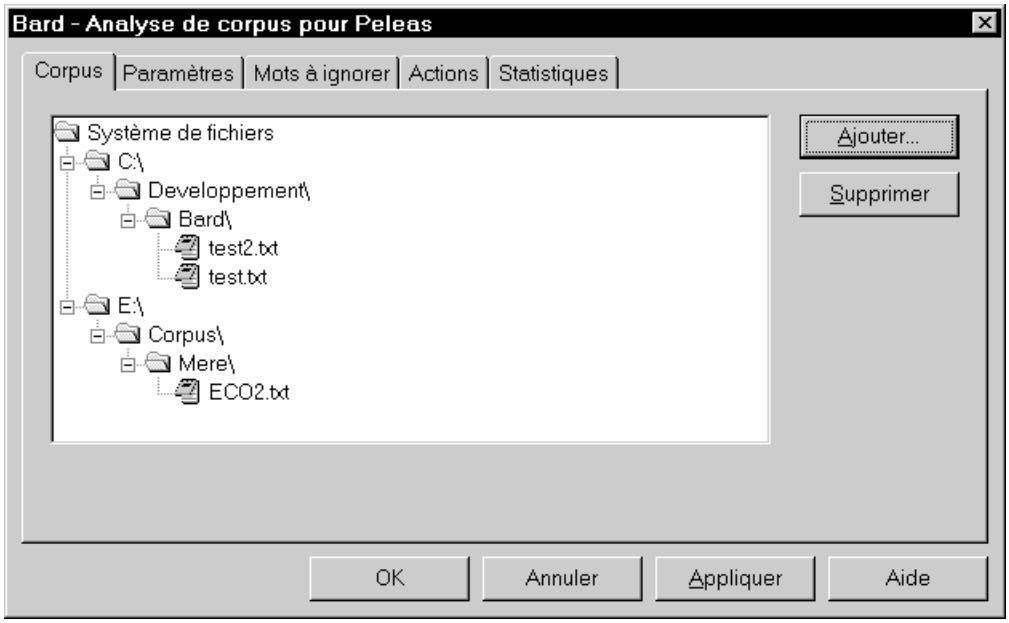
Le prototype réalisé est une application 32 bits pour WINDOWS 95/NT4, prenant en entrée des fichiers texte ASCII non formatés et fournissant en sortie des fichiers d'entrées lexicales PELEAS au format PDL. La construction des entrées lexicales est faite par l'intermédiaire du noyau LIGHTPELEAS, qui doit donc impérativement avoir été enregistré dans le système d'exploitation préalablement à toute utilisation de BARD.

DONNÉES TECHNIQUES	
LIGNES DE CODE :	**8.555 lignes**
TAILLE DE L'EXÉCUTABLE :	**107 Ko**
EMPREINTE MÉMOIRE MAXIMALE :	**1,4 Mo**

L'application BARD fait appel aux contrôles ActiveX LIGHTPELEAS et AMORPH pour fonctionner : son empreinte mémoire maximale *y compris* ces deux contrôles peut donc s'élever à 5,2 Mo.

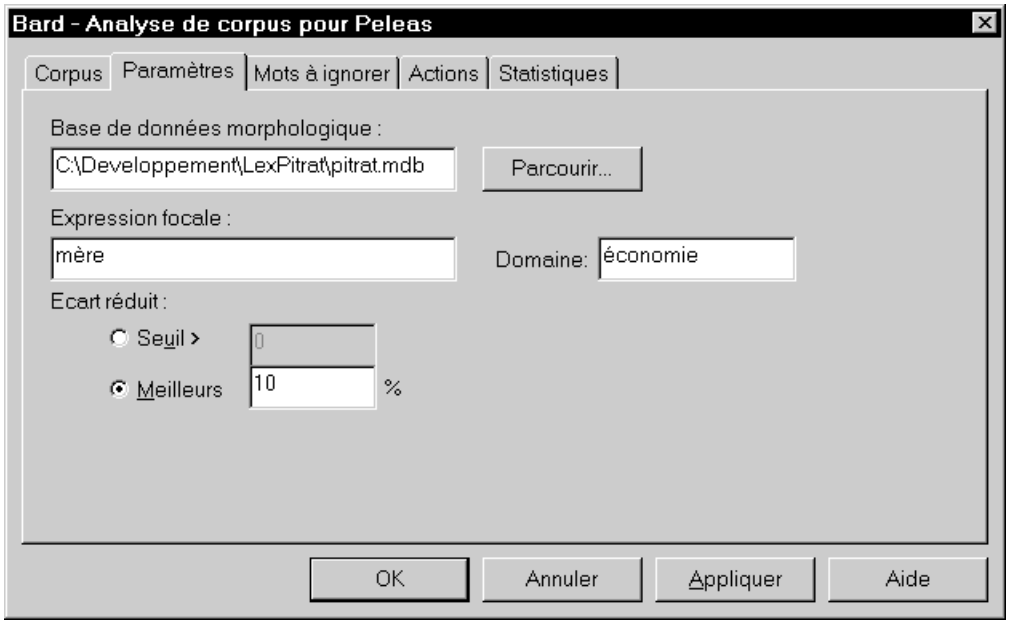
L'application BARD se présente sous la forme d'une boîte de dialogue contenant cinq onglets : Corpus, Paramètres, sf Mots à ignorer, Actions et Statistiques. Les trois premiers onglets permettent de spécifier les paramètres de l'extraction de données, l'onglet Actions détermine les traitements à effectuer sur le corpus, et l'onglet Statistiques présente les termes retenus.

Onglet Corpus



Cet onglet permet de fixer la liste des fichiers texte ASCII constituant le corpus à analyser. On peut modifier cette liste grâce aux boutons Ajouter et Supprimer.

Onglet Paramètres



Cet onglet permet de fixer les paramètres principaux de l'analyse :

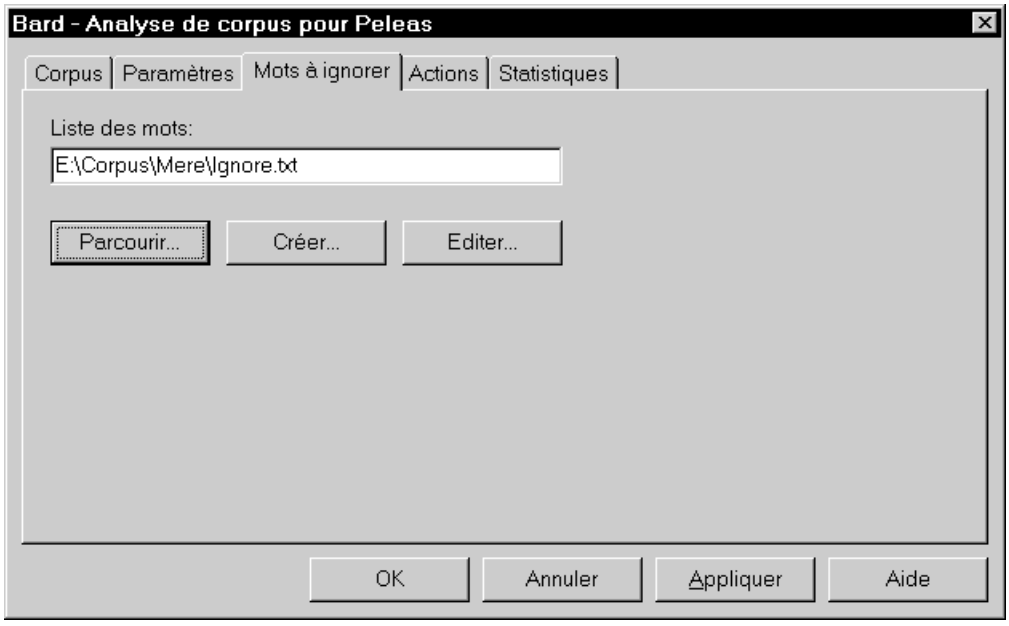
— Le champ Base de données morphologiques permet de choisir le lexique de type Pitrat utilisé pour l'analyse morphologique des mots du corpus par le contrôle ActiveX AMORPH (cf. §5.6. Le bouton Parcourir utilise l'explorateur de fichiers standard de WINDOWS 95/NT4 pour localiser un nouveau lexique.

— Le champ Expression focale doit contenir le mot ou l'expression dont BARD recherche les équivalents dans le corpus. Il s'agit donc de l'entrée lexicale dont on cherche à construire la structure descriptive.

— Le champ Domaine, s'il est rempli, contient l'étiquette du domaine d'activité sur lequel porte le corpus. Si ce champ est laissé vide, BARD considérera qu'il s'agit du domaine *GHOST*.

— Les champs sous Ecart réduit permettent de choisir de quelle façon les équivalents e_j de l'expression focale w seront sélectionnés en fonction de leur score d'écart réduit : soit en fixant un seuil minimal σ (donc il faut que $r > \sigma$ pour que e_j soit sélectionné par BARD), soit en ne conservant qu'un pourcentage π des meilleurs scores d'écart réduit. Ces deux options sont mutuellement exclusives.

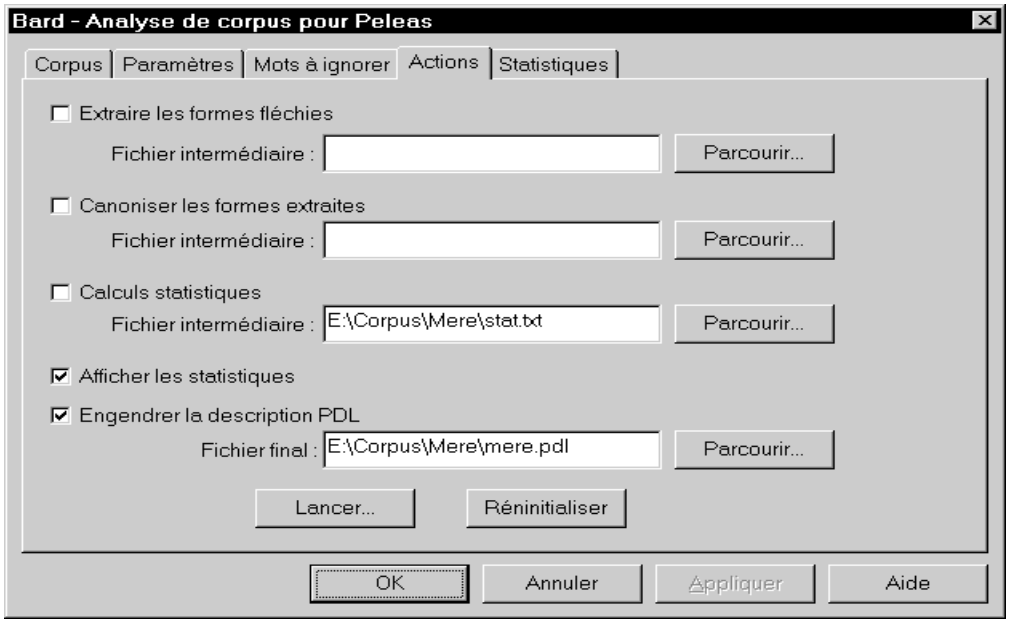
Onglet Mots à ignorer



Cet onglet ne doit pas être obligatoirement rempli avant de lancer l'analyse. Il permet de spécifier un fichier texte ASCII qui contient une liste de mots que BARD doit ignorer lors de l'extraction des formes fléchies. Ceci permet, par exemple, d'éviter de prendre en compte les articles, ou certaines abréviations si l'utilisateur pense que ces mots ne sont *a priori* pas pertinents comme constituants de la structure descriptive de l'expression focale.

Le bouton Parcourir permet d'utilisateur l'explorateur de fichiers pour choisir un nouveau fichier, tandis que les boutons Créer et Editer permettent d'appeler le Bloc-Notes de WINDOWS 95/NT4 sans sortir de BARD, afin de créer et respectivement de modifier le fichier spécifié dans le champ Liste des mots.

Onglet Actions

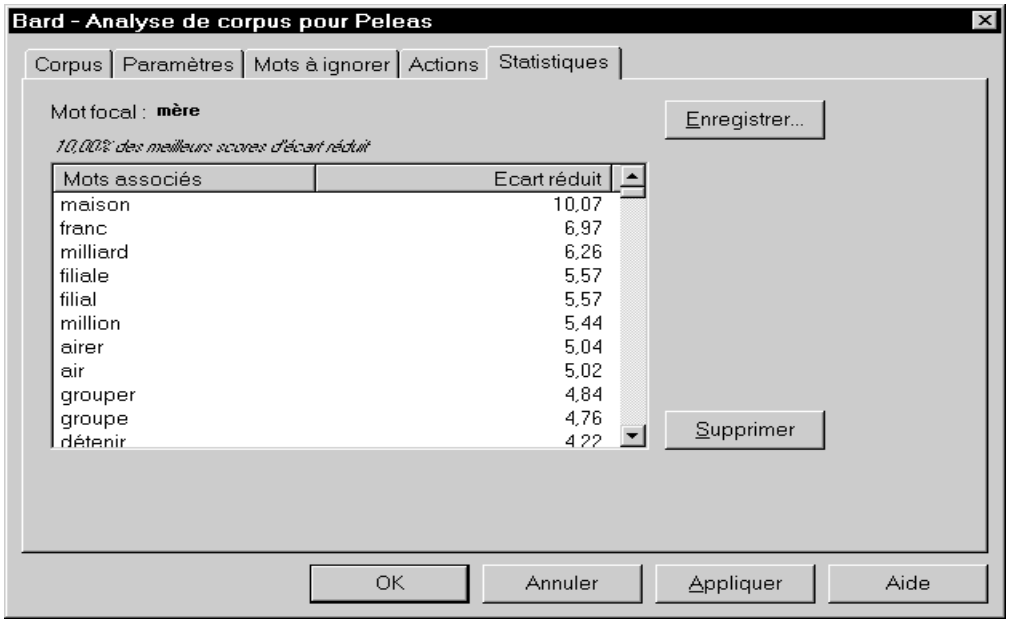


Cet onglet contient un certain nombre de cases à cocher correspondant aux différentes actions que BARD peut effectuer.

Pour chaque action autre que Afficher les statistiques, il faut spécifier un fichier d'entrée (le fichier d'entrée pour l'action Extraire les formes fléchies est le corpus en lui-même) situé en amont, et un fichier de sortie en aval.

Avant d'exécuter les actions prescrites, BARD s'assure de la cohérence des paramètres qui lui ont été fournis : tous les paramètres nécessaires doivent être présents, et lorsqu'un paramètre a changé (comme l'expression focale, par exemple), certaines actions doivent nécessairement être reprises.

Onglet Statistiques



Cet onglet permet de consulter les résultats statistiques à l'issue de l'analyse de corpus.

On y trouve rappelés l'expression focale, la méthode de sélection des résultats ainsi que le seuil ou le pourcentage associé. En dessous figurent les mots sélectionnés avec en regard leur score d'écart réduit.

Il est possible de supprimer un mot de cette liste (probablement introduit en raison du bruit engendré par la règle 2 exposée au §5.5.1.2) grâce au bouton Supprimer.

Le bouton Enregistrer permet d'engendrer une description PDL contenant les mots sélectionnés, comme présenté au §4.4.3 (identique au fait de sélectionner Engendrer la description PDL dans l'onglet Actions).

Si le fichier PDL spécifié n'existe pas, il est créé de toutes pièces. Sinon, BARD ajoute les informations qu'il vient d'extraire à celles déjà présentes dans le fichier.

5.6 L'analyseur morphologique AMORPH

Comme nous l'avons dit précédemment (cf. §5.5), l'analyseur de corpus BARD effectue ses calculs d'information mutuelle entre les formes canoniques des lexies figurant dans les corpora, et non pas entre les lexies elles-mêmes. Or, lors du développement de BARD, le seul logiciel de lemmatisation disponible pour les environnements WINDOWS 95/NT4 était WINBRILL v1.0, basé sur l'étiqueteur de Brill [13]. Or cette version ne mettait pas encore en œuvre la partie lemmatisation. Contraint par des critères de rapidité de développement, nous avons décidé de développer un prototype (assez fruste) de lemmatiseur pour WINDOWS 95/NT4.

De cette décision est né le logiciel AMORPH, qui, comme son nom l'indique, est un analyseur morphologique. C'est un analyseur de type Pitrat muni, pour le moment, d'une base de données morphologiques concernant plus de 22.000 formes canoniques, 167 conjugaisons et 100 jeux de déclinaisons. Tout comme LIGHTPELEAS, l'analyseur AMORPH est un contrôle ActiveX. Il est donc utilisable par n'importe quelle application ayant recours à ce standard.

DONNÉES TECHNIQUES	
LIGNES DE CODE :	**4.125 lignes**
TAILLE DE L'EXÉCUTABLE :	**49 Ko**
EMPREINTE MÉMOIRE MAXIMALE :	**1,8 Mo**

5.6.1 Spécifications du composant AMORPH

AMORPH est un analyseur morphologique de style Pitrat. La base de données morphologique qui en constitue le cœur est présentée à l'annexe E.

Comme nous l'avons annoncé précédemment, le composant logiciel AMORPH est un composant ActiveX qui fait appel au moteur du SGBD/R Access afin d'exécuter les requêtes d'analyse morphologique (cf. §E.3). Cela se fait également à travers le protocole DCOM, car Access est lui-même un composant ActiveX qui permet aux composants extérieurs de manipuler ses données par l'intermédiaire des *Data Access Objects* (il s'agit d'une famille d'interfaces dédiées à la manipulation des bases de données, cf. objets de données, p. 353).

Les flots de données mis en jeu sont schématisés sur la figure 5.5 :

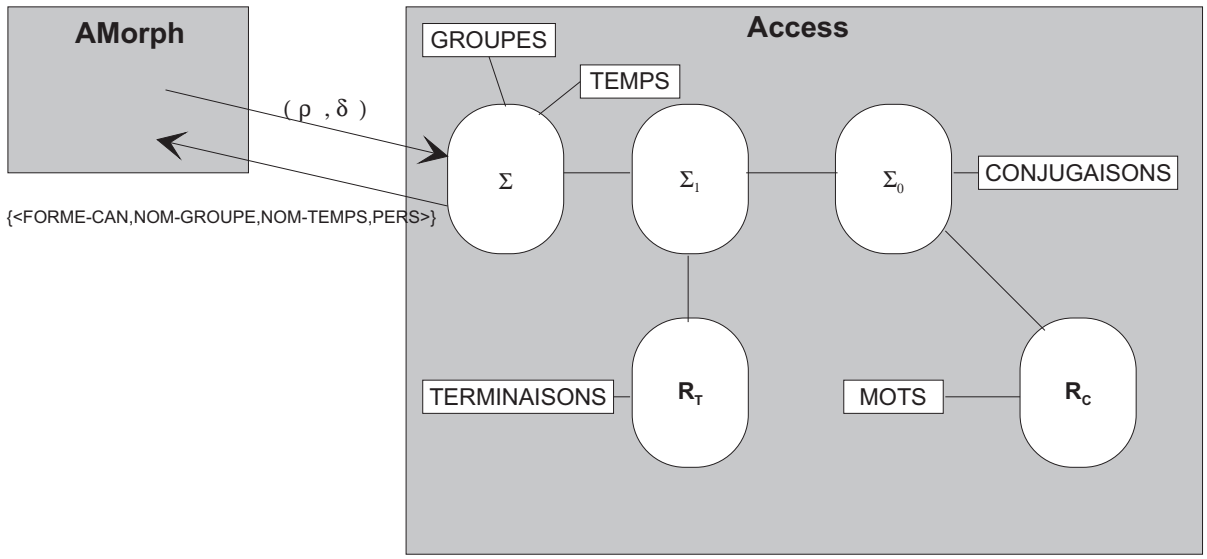


FIG. 5.5: Echanges de données entre AMORPH et Access

Le composant AMORPH exporte trois interfaces (D)COM auprès du système d'exploitation, correspondant à des classes d'objets nous permettant de modéliser le processus d'analyse morphologique : il s'agit des interfaces `IAnaLex`, `IFlexedForms` et `IFlexedForm`. Chacune d'elle est assurée par une unique coclasse, respectivement `AnaLex`, `FlexedForms` et `FlexedForm`.

Sémantiquement, ces trois coclasses s'interprètent de la façon suivante :

— une instance d'`AnaLex` représente le "moteur" de l'analyseur morphologique AMORPH ; c'est donc cette coclasse qui tient lieu, pour les autres applications, de l'agent analyseur morphologique : c'est à elle que les clients soumettent leurs requêtes ;

— une instance de `FlexedForms` représente la collection des analyses possibles d'une forme fléchie ; étant donné qu'elle incarne le "résultat" de l'analyse morphologique, elle ne peut être indépendante d'une instance d'`AnaLex`, c'est pourquoi nous en ferons une *propriété étendue* d'`AnaLex` ;

— une instance de `FlexedForm` représente une analyse possible d'une forme fléchie, c'est-à-dire un élément d'une collection `FlexedForms` ; ce sera donc une propriété étendue de `FlexedForms`.

Les spécifications formelles correspondant à ces trois coclasses se trouvent en annexe (cf. §D.2).

5.6.1.1 Interface IAnaLex

Rappelons tout d'abord que seule la coclasse `AnaLex` est instanciable par une application cliente d'AMORPH. Son instanciation implique l'instanciation des coclasses `FlexedForms` et `FlexedForm` suivant les résultats des analyses morphologiques.

— Propriété `LexicalDB` :

  – En lecture, renvoie une chaîne de caractères qui contient le chemin de la base de données couramment utilisée.

– En écriture, elle permet de changer de base de données morphologiques en indiquant le chemin d'un nouveau fichier de base de données.

Par défaut, cette propriété a la valeur `"pitrat.mdb"` qui contient la base de données que nous avons décrite dans les paragraphes précédents. *Attention : changer de base de données implique que la nouvelle base contienne les mêmes schémas de tables et les mêmes requêtes que la base pitrat.mdb.*

— Méthode `Load` : sans paramètre, charge en mémoire le contenu de la base de données dont le chemin est indiquée par la propriété `LexicalDB`. Appelée avec un paramètre de type chaîne de caractères, elle charge en mémoire la base de données dont le chemin est indiqué par ce paramètre, et change la valeur de la propriété `LexicalDB`.

— Propriété `Loaded` : en lecture seule, renvoie un booléen indiquant si la base de donnée dont le chemin est donné par la propriété `LexicalDB` est chargée en mémoire (et donc utilisable) ou non.

— Méthode `Analyze` : attend un paramètre de type chaîne de caractères qui contient la forme fléchie d'un mot à analyser. Cette méthode lance les requêtes d'analyse contenues dans la base de données courante et construit les objets résultats de type `FlexedForms` et `FlexedForm`.

— Propriété `Known` : en lecture seule, renvoie un booléen indiquant si le dernier appel de la méthode `Analyze` a produit des analyses de la forme fléchie.

— Propriété `FlexedForms` : renvoie une référence de type `FlexedForms` à la collection des analyses produites par le dernier appel de la méthode `Analyze`.

5.6.1.2 Interface IFlexedForms

— Propriété `Count` : en lecture seule, renvoie le nombre d'analyses contenues dans cette collection d'éléments.

— Propriété `Item` : en lecture seule, attend un paramètre de type entier. Cette propriété renvoie une référence de type `FlexedForm` sur l'élément dont l'indice dans la collection est indiqué par le paramètre. Si l'indice est hors limites, la consultation de cette propriété lève une exception utilisateur.

5.6.1.3 Interface IFlexedForm

— Propriété `CanonicalForm` : en lecture seule, renvoie une chaîne de caractères qui contient la forme canonique correspondant à cette analyse de la forme fléchie passée en paramètre de la méthode `IAnaLex::Analyze`.

— Propriété `Group` : en lecture seule, renvoie une chaîne de caractères qui contient le nom lisible du groupe de conjugaison correspondant à cette analyse.

— Propriété `Tense` : en lecture seule, renvoie une chaîne de caractères qui contient le nom lisible du jeu de terminaison (ou *temps* par abus de langage) correspondant à cette analyse.

— Propriété `Person` : en lecture seule, renvoie un entier qui est le numéro de déclinaison (ou *personne* par abus de langage) correspondant à cette analyse.

5.6.2 Remarques d'ordre technique

Uniquement à titre d'élément de comparaison, nous tenons à préciser que les temps de réponse du composant AMORPH s'échelonnent entre 0,1 seconde et 0,7 seconde environ pour toutes les formes fléchies reconnaissables dans le lexique que nous avons utilisé (le temps le plus court est réalisé pour un mot invariable, et le plus long pour un mot qui n'appartient pas au lexique). Ces temps ont été mesurés sur un Pentium 100 Mhz, utilisant le système d'exploitation WINDOWS NT4 et disposant de 48 Mo de mémoire vive, dans le cas défavorable où Access n'était pas précédemment chargé en mémoire.

5.7 Récapitulation

Au final, nous disposons donc d'un atelier logiciel comprenant :

— un module de traitement des ambiguïtés lexicales, LIGHTPELEAS, qui met en œuvre le modèle PELEAS et s'appuie sur une base de données lexicales (la liste des entrées) ;

— un éditeur graphique, MELISANDE, qui permet de maintenir la base des entrées lexicales ;

— un analyseur de corpus, BARD, qui est un outil d'aide à la création d'entrées lexicales. Celui-ci utilise en outre un analyseur morphologique, AMORPH, à des fins de lemmatisation des corpus qu'il traite.

L'utilisation de LIGHTPELEAS par un agent extérieur (humain ou artificiel) se présente ainsi :

— En entrée : l'agent demande à LIGHTPELEAS d'évaluer le potentiel sémantique d'une entrée lexicale. Les informations contextuelles sont fournies à LIGHTPELEAS par l'intermédiaire d'une ou plusieurs base(s) de prédicats. Dans le cas d'un agent humain, les prédicats peuvent être interactifs (questions posées à l'utilisateur), tandis que dans le cas d'un agent artificiel (système de traitement automatique des langues naturelles), il s'agira plutôt de tests sur les représentations internes manipulées par l'agent pour modéliser le contexte.

— En sortie : LIGHTPELEAS fournit le potentiel valorisé de l'entrée lexicale s'il a pu unifier les règles contextuelles avec le contexte d'évaluation. Si l'agent extérieur est capable de manipuler des structures non atomiques, le résultat de LIGHTPELEAS est un ensemble de vues conceptuelles fortement activées (saillantes ou niées) réunies dans la collection `Guesses`. Mais si l'agent ne peut manipuler que des représentations atomiques, alors il pourra consulter la valeur de l'estimateur $\widehat{v}$ associé à chaque vue conceptuelle de `Guesses`. La vue conceptuelle dont l'estimateur $\widehat{v}$ est le plus fort est celle qui a vraisemblablement la plus grande importance dans la construction du sens de cette occurrence de l'entrée lexicale. Sélectionner cette vue reviendra donc à une perte d'information limitée au maximum.

Chapitre 6

Bilan et perspectives

> « La chasse a passionné l'humanité depuis toujours, et avec juste raison. Ce que beaucoup n'ont pas compris, cependant, c'est que la chasse peut être exaltante même si la chose poursuivie n'est qu'une idée, un concept, une théorie. A mesure que s'est développée la conscience, il est devenu évident que c'est de toutes la poursuite la plus importante, celle dont l'issue conditionne la survie ou l'échec de toute l'humanité. »
>
> **Frank Herbert**, *Destination Vide*

6.1 Récapitulation

Durant ce travail de recherche, nous nous sommes livré à un travail d'enquête, remontant des indices les plus flagrants à des mécanismes cachés. L'objectif que nous avions était celui-ci : à partir de l'observation des ambiguïtés lexicales si courantes en langue, comment parvenir à constituer un modèle opérationnel et automatisable capable de traiter avec robustesse le genre d'ambiguïtés qui met habituellement les analyseurs sémantiques en défaut.

Une première exploration des ambiguïtés lexicales nous a tout d'abord conduit à nous méfier des modèles traditionnels inspirés de la linguistique chomskyenne qui fait pourtant figure de référence en traitement automatique des langues naturelles. Les deux approches classiques qui en découlent (toute ambiguïté lexicale est soit une *homonymie*, soit une *indétermination*) semblent en effet trop réducteurs pour expliquer un grand nombre d'ambiguïtés qu'on rencontre courramment. Cela nous a donc conduit à nous tourner vers une troisième alternative : considérer les ambiguïtés comme résultant d'un phénomène de *polysémie* (de multiplication des significations potentielles).

L'étape suivante a été de rechercher les caractérisations linguistiques de la polysémie, et d'observer comment elle était prise en compte par le traitement automatique des langues naturelles. Cela a permis d'établir les points suivants :

1. Il existe au moins trois types de polysémie, mettant en jeu des processus de plus en plus complexes et des quantités d'information toujours plus grandes pour être traités : la *polysémie fonctionnelle*, la *polysémie d'acception* et la *polysémie d'usage*.

2. Généralement, seule la polysémie fonctionnelle est traitée par les systèmes automatiques. Cela est dû, selon nous, aux modèles théoriques sous-jacents qui tendent à confondre lexique et connaissances, et qui attribuent un rôle prééminent à la syntaxe dans le fonctionnement de la langue.

3. La polysémie d'acception a été peu étudiée (comparativement) en linguistique, et encore moins en traitement automatique des langues naturelles.

4. La polysémie d'usage, qui résulte de la co-participation de plusieurs nuances de sens subtilement différentes dans une même signification, n'est jamais placée en tant qu'objet d'étude autonome alors qu'elle engendre pourtant des ambiguïtés extrêmement nombreuses et courantes.

C'est cette forme de polysémie complexe et subtile que nous nous sommes attaché à décrire et à modéliser dans cet ouvrage, et que nous avons tout d'abord cherché à caractériser en construisant une typologie de ses manifestations : *transferts d'usages*, *cumuls d'emplois*, *mots jokers* et *jeux de langage*.

Pour cela, nous avons fait appel aux grands modèles sémantiques classiques pour nous fournir une base de travail, qu'ils viennent de la linguistique ou de l'informatique. Par l'observation de ces modèles, nous avons été amené à rejeter tour à tour le paradigme référentiel et le paradigme inférentiel, abondamment développés en linguistique, et qui servent de base à l'élaboration de quasiment tous les modèles théoriques actuels du traitement automatique des langues naturelles. Nos critères de rejet ont été principalement que :

— ces paradigmes induisent des postulats philosophiques trop forts vis-à-vis d'une application technologique ;

— notamment, ils reposent sur les *concepts* comme objets de base, or nous n'avons pas trouvé de fondement théorique capable de cerner ce qu'est exactement un concept ni comment il fonctionne ;

— de plus, la notion de concept alourdit énormément le traitement de la variabilité linguistique, et va même parfois jusqu'à se contredire elle-même, alors que la notion de *signifié* semble convenir beaucoup mieux ;

— et surtout, ces paradigmes entrent en contradiction avec l'observation de la polysémie d'usage puisqu'ils n'admettent pas que plusieurs interprétations différentes participent simultanément à la signification d'un mot polysémique.

Seul le paradigme de la sémantique différentielle nous a semblé convenir, notamment parce qu'il se concentre plus sur le matériau de la langue : il délaisse les concepts au profit des signifiés, et surtout, il remplace la modélisation de la *compréhension* (qui suppose un sujet comprenant) par celui de l'*interprétation*, phénomène intralinguistique d'explicitation, qu'on conçoit beaucoup plus facilement automatisable.

Cependant, cette théorie fournit trop peu de critères objectifs systématiques que nous pourrions facilement programmer. Nous lui avons donc adjoint une étude empirique consistant en un modèle fonctionnel d'une activité d'interprétation particulière : la *traduction*. L'aspect didactique de cette activité nous a permis de déterminer ces critères qui nous faisaient jusqu'ici défaut.

Le modèle informatique que nous avons conçu à partir de ces données, PELEAS, se range dans la catégorie des *lexiques dynamiques* : seul un noyau minimal de sens est stocké de façon statique, tandis que la majeure partie du processus d'interprétation est conduite de façon dynamique par des règles d'inférences.

Ce modèle s'inspire d'un modèle antérieur, EDGAR, spécialement conçu pour traiter la polysémie des mots courants. Les deux modèles ont donc de nombreuses caractéristiques communes :

— ce sont tous deux des modèles hybrides appliquant des techniques de propagation d'activité connexionistes à un réseau de connaissances symboliques ;

— ils utilisent des techniques de coercion et de taxation d'activité lors de sa propagation dans le réseau ;

— ils sont munis d'un mécanisme de raisonnement par défaut ;

— le processus d'interprétation est *amorcé* par les informations contextuelles, plutôt que *complété* par elles ;

— et finalement, ils fournissent tous deux des réponses *ensemblistes* comme interprétations.

Cependant, PELEAS se démarque de son parent par un certain nombre de propriétés fondamentales :

— Il présente une structuration des données *hiérarchique* (contrairement à la structure plate d'EDGAR), ce qui lui permet d'offrir une profondeur variable dans le raisonnement sémantique sur les descripteurs d'une entrée lexicale.

— Le jeu de contraintes sémantiques étendu de PELEAS le munit d'un processus de relaxation de contraintes plus souple.

— La différence fondamentale entre les deux modèles se situe au niveau pratique de la constitution des potentiels sémantiques : **tous les sommets d'un potentiel sémantique sont des lexicalisations reliées à l'entrée lexicale uniquement par co-occurrence lexicale ou par paraphrase**, tandis que la constitution d'un potentiel sémantique d'EDGAR demande un travail important d'abstraction conceptuelle. Nous nous en sommes tenus, quant à nous, à notre programme de départ : évacuer toute tentation de recours aux concepts pour ne tenir compte que des signifiés et de leurs réalisations.

— PELEAS se soucie plus de son intégration dans un système classique de traitement automatique des langues naturelles, en ordonnant ses réponses ensemblistes à l'aide d'estimations numériques offrant un large degré de liberté à l'analyseur sémantique pour traiter ces réponses de manière atomiste (par d'éventuels retour-arrières) ou ensembliste (en parallèle).

La mise en œuvre informatique de PELEAS s'est révélée robuste, compacte et efficace. Ceci répond d'une part à nos objectifs fondamentaux qui étaient de créer un modèle *robuste* de traitement automatique des polysémies complexes. Mais cela démontre en outre qu'il est possible d'écrire des logiciels complexes d'intelligence artificielle et de les faire tourner sur des *micro-systèmes*. Il est de plus tout à fait possible, et même souhaitable, de tirer parti des nouvelles fonctionnalités modulaires des systèmes d'exploitation pour distribuer le traitement des informations à travers un réseau de micro-ordinateurs (donc peu coûteux). Cela montre qu'il est possible de s'affranchir des systèmes informatiques lourds et donc d'entrevoir une "descente" des applications d'intelligence artificielle depuis les milieux de recherche rares et privilégiés dans un environnement grand public.

En conclusion, nous retiendrons ceci. Le modèle PELEAS de traitement automatique des formes de polysémie complexe, malgré la pauvreté de ses constituants, possède un pouvoir d'interprétation très

important. Ce pouvoir est assûrément dû au fait que toute notion de conscience ou de compréhension a été évacuée de ce modèle. Il s'agit donc d'un modèle purement dénotationnel, qui utilise la langue comme système de représentation interne. Cela conduit au fait que toute "compréhension" se situe en fait dans l'esprit de l'observateur humain de ce modèle. PELEAS ne comprend ni n'interprète : il se contente de ne pas enfermer la langue dans un système formel rigide, et de la laisser faire émerger ses propres interprétations.

Le modèle PELEAS repose *en interne* (c'est-à-dire pour la partie propagation d'activité) sur le paradigme de la sémantique différentielle, et *en externe* (i.e. pour l'identification du contexte) sur le paradigme de la sémantique inférentielle. Il faut toutefois remarquer que l'aspect inférentiel est ici ramené à sa signification première : PELEAS conçoit le contexte comme contenant des *indices* dont il faut retrouver les causes par induction (alors que traditionnellement, les indices sont assimilés à des signes dont on déduit des conséquences). L'exploration du contexte par PELEAS est donc un processus de *diagnostic* : on observe des *symptômes* dans le contexte et on essaye d'en *induire* les causes dans le potentiel sémantique.

C'est un modèle dont la puissance est inversement proportionnelle à la précision des informations contextuelles : dans le cas d'une situation idéale (information claire et déterminée), PELEAS se comporte comme n'importe quel système de restriction de la sélection, mais plus la situation se dégrade et plus il fait appel à son "expérience de la langue" pour combler le vide sémantique. Il montre une habileté toute particulière pour l'interprétation des jeux de mots, phrases à tiroirs et autres doubles sens.

C'est pourquoi nous aimerions, pour finir, décrire le modèle PELEAS comme **un agent sans humour capable d'interpréter un jeu de mots sans le comprendre**.

6.2 Apports du/au génie logiciel

Comme argumenté par Choppy dans [20], l'utilisation des spécifications algébriques pour la conception de notre modèle objet nous a permis d'approfondir notre compréhension du problème. Cela nous a conduit à remanier plusieurs fois certains détails de formalisation, tant au niveau de PELEAS qu'à celui de l'architecture de notre solution logicielle.

En introduisant les spécifications algébriques, le génie logiciel apporte des outils pour assurer la qualité de la modularité, de l'encapsulation et de l'abstraction des éléments logiciels. Ces outils se sont avérés pour nous très utiles, surtout si l'on considère que le code source de l'ensemble des logiciels produits contient plus de 400 classes (au sens de la programmation orientée objets), chacune interagissant avec en moyenne une dizaine d'autres classes.

Notre démarche de conception nous a en nous apporté un éclairage original en ce qui concerne l'utilisation conjointe des spécifications algébriques et du modèle DCOM d'ActiveX. Une première boucle itérative entre spécification algébrique et rédaction d'interface IDL permet d'obtenir rapidement un bon niveau d'abstraction (éléments purement fonctionnels *vs* solutions de mise en œuvre). De plus, leurs formalismes sont suffisamment proches pour que l'on puisse mettre à profit les outils de développement habituellement associés à ActiveX pour produire très rapidement des prototypes partiels des agents[1].

Tout cela nous laisse penser que l'ajout d'outils de spécification algébrique[2] dans les plate-formes de développement actuelles (qui comportent déjà des outils de prototypage par spécification semi-formelle

[1] Comme exposé par Choppy [20], le prototypage est une phase essentielle de la validation.
[2] Du type ASSPEGIQUE.

de type UML), représenterait une aide considérable pour l'aide à la conception de logiciels de moyennes et grandes tailles.

6.3 Perspectives

Cette étude nous a permis de découvrir un grand nombre de champs de recherche possibles que nous n'avons qu'entrevus ou effleurés, mais qui méritent une attention soutenue.

Nous citerons plus particulièrement quatre points qui nous ont marqué. Les deux premiers concernent l'évolution du modèle PELEAS lui-même, et les deux suivants concernent ses applications possibles.

1. Nous avions vu au§4.4 à propos de l'acquisition des connaissances permettant de constituer un lexique, qu'une hypothèse fondamentale de notre raisonnement était que les éléments lexicaux récurrents dans les paraphrases d'une entrée lexicale pouvaient constituer des sommets de son potentiel sémantique.

   Cette hypothèse a été en partie déçue empiriquement lors de l'évaluation du modèle, puisque nous avons constaté que les mots retenus par le logiciel BARD sont plutôt utiles pour former des conditions contextuelles de co-occurrence ("si le mot m apparaît dans le contexte alors...") que pour servir de constituants au potentiel sémantique.

   C'est une raison supplémentaire d'étudier en soi-même l'automatisation de l'acquisition des données pour constituer des potentiels sémantiques.

2. Au moment de la rédaction de cette conclusion, nous sommes en train de formaliser une méthode permettant de calculer des interactions entre deux configurations de potentiels sémantiques pour deux mots co-occurrents. Nous espérons par-là obtenir un processus d'abstraction et de synthèse permettant de retrouver des *signifiés virtuels*, c'est-à-dire qui doivent exister dans le système fonctionnel de la langue, mais qu'on ne peut observer qu'indirectement parce qu'ils ne sont pas (ou plus) lexicalisés.

3. Notre travail commun avec des traducteurs nous a fait comprendre que le modèle PELEAS pouvait avoir une application intéressante en Traduction Assistée par Ordinateur. L'idée principale, suggérée par les traducteurs, est la suivante : former des potentiels sémantiques où l'entrée lexicale appartient à la langue source et les autres sommets appartiennent à la langue cible, repérés par des *traductions attestées*, et les conditions contextuelles se situent également dans la langue source.

   Couplé à un processus de génération, on pourrait alors obtenir des sorties explicatives de la forme : *Le mot m peut être traduit par :*

   — t_1 *car ...*

   — *...*

   — t_n *car ...*

   A charge ensuite pour les traducteurs de construire une traduction tenant compte des proposition t_1 à t_n. Ces propositions pourraient constituer une aide à la traduction d'idiomatismes, notamment, où les aspects culturels idiomatiques sont détectés en langue source mais leurs conséquences sont exprimés en langue cible.

4. Et finalement, suite aux remarques d'un relecteur portant sur un de nos articles [121] il apparaît que la structure générale des potentiels sémantiques pourrait être réutilisée pour la modélisation des connaissances en incertain, c'est-à-dire avec des mondes alternatifs dont on ne connaît pas la probabilité de réalisation. Pour ce genre d'application, la bijection présentée en annexe au §F s'avère particulièrement utile, puisqu'elle montre qu'on peut représenter linéairement en machine des structures de connaissances hiérarchiques qui supportent *implicitement* la notion de contexte ou de *vista*, ce qui laisse entendre qu'il est possible d'améliorer notablement le traitement algorithmique des explosions combinatoires propres à ce genre de représentations.

Bibliographie

[1] Adriaens (G.) et Small (S.). – Word Expert Parser Revisited in a Cognitive Science Perspective. In Small et al. [115], pp. 13–43.

[2] Aristote. – *De l'interprétation*. – Paris, Vrin, 1965. Trad. J. Tricot.

[3] Bassi Acuña (A.). – *Un modèle dynamique de la compréhension de textes intégrant l'acquisition de connaissances*. – Thèse de doctorat, Université Paris XI, Orsay, 1995.

[4] Benveniste (E.). – *Problèmes de linguistique générale I*. – Paris, Gallimard, 1966.

[5] Benveniste (E.). – *Problèmes de linguistique générale II*. – Paris, Gallimard, 1974.

[6] Berthonneau (A.-M.) et Kleiber (G.). – Pour une nouvelle approche de l'imparfait : l'imparfait, un temps anaphorique méronymique. *Langages*, 112, 1993, pp. 55–73.

[7] Beust (P.). – Un modèle oppositionnel de la signification. *In : Actes de RéciTAL'98*. – Le Mans, 1998.

[8] Bonhomme (P.). – Apport de la statistique lexicale dans la recherche d'information. *In : Actes de ISKO-France 97*. – Lille, 1997.

[9] Bourdieu (P.). – *Ce que parler veut dire*. – Paris, Fayard, 1982.

[10] Brachman (R.). – *On the Epistemological status of semantic Networks*, pp. 3–50. – New York, Findler Academic Press, 1979.

[11] Bréal (M.). – *Essai de sémantique (science des significations)*. – Paris, Hachette, 1897.

[12] Briffaut (X.), Chibout (K.), Sabah (G.) et Vapillon (J.). – A Linguistic Engineering Environment based on Lexical Functionnal Grammar and Conceptual Graphs. *In : Actes de LFG'97*. – University of California, San Diego, 1997.

[13] Brill (E.). – A simple rule-based part of speech tagger. *In : Proceedings of the Third Conference on Applied Natural Language Processing*. – Trente, Italie, 1992.

[14] Briscoe (E.) et Copestake (A.). – Some extensions as Lexical Rules. *In : Proceedings of the IJCAI-Workshop on Computational Approaches to Non-Literal Language, Metaphor, Metonymy, Idiom, Speech Acts and Implicature*. pp. 12–20. – Boulder, 1991.

[15] Brockschmidt (K.). – *Inside OLE (2nd edition)*. – Redmond (WA), Microsoft Press, 1995.

[16] Brugère (S.). – *Etude d'un système de traitement de la polysémie d'acceptions*. – Mémoire de DEA de Sciences Cognitives, LIMSI, Université Paris XI, Orsay, 1990.

[17] Chenciner (A.). – Systèmes dynamiques différentiables. *Encylopœdia Universalis*, 21, 1992, pp. 1044–1080.

[18] Chomsky (N.). – *Syntactic Structures*. – La Haye, Mouton, 1957.

[19] Choppy (C.). – *ASSPEGIQUE User's Manual*. – Rapport interne n 452, LRI, Unité associée au CNRS UA 410, Université Paris XI, 1988.

[20] Choppy (C.). – *Spécifications algébriques : Prototypage et Validation*. – Rapport scientifique présenté pour l'obtention d'une habilitation à diriger des recherches n 891, LRI, Unité associée au CNRS UA 410, Université Paris XI, 1994.

[21] Choquer (M.-A.), Gaudel (M.-C.), Holvoet (Y.) et Mauboussin (A.). – *A Concrete Syntax for PLUSS*. – METEOR Report, 1986. Task 10 and 11.

[22] Church (K.) et al. – Using statistics in lexical analysis. *In : Lexical Acquisition Exploiting On-line Resources to Build a Lexicon*, éd. par Zernik (U.), pp. 65–90. – Laurence Erlbaum Associates, 1991.

[23] Copestake (A.) et Briscoe (E.). – Lexical operations in a unification-based framework. *In : Lexical Semantics and Knowledge Representation*, éd. par Pustejovsky (J.) et Bergler (S.). – New-York, Springer, 1992.

[24] Culioli (A.). – Valeurs modales et opérations énonciatives. *Le Français moderne*, 46(4) , 1978, pp. 300–317.

[25] Dahlgreen (K.), McDowell (J.) et Stabler (E. P.). – Knowledge representation for commonsense reasonning with text. *Computational Linguistics*, 15(3) , 1989, pp. 149–170.

[26] de Saussure (F.). – *Cours de linguistique générale*. – Paris, Payot, 1972.

[27] Debili (F.). – Une méthode d'apprentissage de relations lexicales sémantiques orientées vers la résolution d'ambiguités lexicales et structurales. *In : Actes RFIA*. – France, 1984.

[28] Dollan (W.). – Word Sense Ambiguation : Clustering Related Senses. *In : Proceedings of COLING'94*. – Kyoto, Japon, 1994.

[29] Ducrot (O.). – L'imparfait en français. *Linguistische Berichte*, 60, 1979, pp. 1–23.

[30] Dyer (M.). – *In Depth Understanding : A Computer Model of Integrated Processing for Narrative Comprehension*. – Cambridge (MA), MIT Press, 1983.

[31] Evens (M.), Litowitz (B.), Markowitz (J.), Smith (R.) et Werner (O.). – *Lexical - Semantic relations : a comparative survey*. – Edmonton, Canada, Linguistic Research Inc., 1983.

[32] Fass (D.). – MET* : A method for discriminating metonymy and metaphor by computer. *Computational Linguistics*, 17(1) , 1991, pp. 49–90.

[33] Ferrari (S.). – *Méthode et outils informatiques pour le traitement des métaphores dans les documents écrits*. – Thèse de doctorat, LIMSI, Université Paris XI, Orsay, 1997.

[34] Ferrari (S.) et Prince (V.). – Création et extension automatique de dictionnaires terminologiques multi-lingues spécialisés à partir de corpus monolingues. *In : Actes de la Conférence internationale sur le traitement automatique des langues et ses applications industrielles TAL+AI 96*, pp. 79–85. – Moncton (NB), Canada, 1996.

[35] Fodor (J. A.). – *The Language of Thought*. – Cambridge (MA), Harvard University Press, 1975.

[36] François (D.) et François (F.). – L'ambiguïté linguistique. *Word*, 23, 1968.

[37] Frege (G.). – *Ecrits logiques et philosophiques*. – Paris, Seuil, 1892.

[38] Fuchs (C.). – *Les ambiguïtés du français*. – Paris, Ophrys, 1996.

[39] Fuchs (C.), Gosselin (L.), Joulin (J.), Konfé (A.), Raysz (J.-P.) et Victorri (B.). – *Réalisation d'un analyseur automatique de certaines formes aspectuelles du français*. – Rapport du laboratoire ELSAP-CNRS, Université de Caen, 1989. Convention CNET-SEPT.

[40] Fuchs (C.) et Victorri (B.) (édité par). – *Continuity in Linguistic Semantics*. – Amsterdam, John Benjamin Publishing Co., 1994, *Linguisticae investigationes supplementa*, volume 19.

[41] Garfinkel (H.). – *Studies in Ethnomethodology*. – Englewood Cliffs (NJ), Prentice Hall, 1967.

[42] Gaudel (M.-C.). – Towards structured algebraic specifications. *In : Procedures of the ESPRIT Technical Week*, pp. 493–510. – North Holland, 1985. Esprit'85 Status Report.

[43] Ghiglione (R.) et Blanchet (A.). – *Analyse de contenu et contenus d'analyses*. – Paris, Dunod, 1991.

[44] Gosselin (L.). – *Sémantique de la temporalité en français*. – Louvain-la-Neuve, Duculot, 1996.

[45] Gosselin (L.), Konfé (A.) et Raysz (J.-P.). – Un analyseur sémantique automatique d'adverbes aspectuels du français. *In : Actes du quatrième colloque de l'ARC*, pp. 257–271. – Paris, INRIA, 1990.

[46] Greimas (A.-J.). – *Sémantique structurale*. – Paris, Larousse, 1966.

[47] Halliday (M. A. K.). – *An Introduction to Functional Grammar*. – Londres, Edward Arnold, 1985.

[48] Halliday (M. A. K.) et Martin (J. R.). – *Readings in Systemic Linguistics*. – Londres, Batsford Academy, 1981.

[49] Herbert (F.). – *Destination : vide*. – Paris, Robert Laffont, 1981.

[50] Hirst (G.). – Resolving Lexical Ambiguity Computationally with Spreading Activation and Polaroid Words. In Small et al. [115], pp. 73–107.

[51] Hopfield (J.). – Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *In : Proceedings of the National Academy of Sciences*, pp. 2554–2558. – U.S.A., 1982.

[52] Hopfield (J.). – Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons. *In : Proceedings of the National Academy of Sciences*, pp. 3088–3092. – U.S.A., 1984.

[53] Jodouin (J.-F.). – Présentation des modèles connexionistes. *Intellectica*, 9(10) , 1990, pp. 9–39.

[54] Johnson-Laird (P.). – La représentation mentale de la signification. *RISS*, 115, 1988, pp. 53–69.

[55] Kaplan (R.) et Bresnan (J.). – LFG : a formal system for grammatical representation. *In : The Mental Representation of grammatical relations*, éd. par Bresnan (J.). – Cambridge (MA), MIT Press, 1982.

[56] Katz (J.) et Fodor (J.). – The structure of a semantic theory. *Language*, 38, 1963, pp. 170–210.

[57] Kayser (D.). – Truth and the interpretation of natural language : A non monotonic approach to variable depth. *In : ECAI-90 Proceedings*, pp. 392–394. – Stockholm, Suède, 1990.

[58] Kayser (D.). – La relation mot-concept. – Conférence donnée au LAFORIA, Université Paris VI, 1991.

[59] Kleiber (G.). – Contexte, interprétation et mémoire : approche standard vs approche cognitive. *Langue Française*, 103, 1994, pp. 9–22.

[60] Konfé (A.), Victorri (B.) et Raysz (J.-P.). – Extraction de règles dans les réseaux connexionistes récurrents. *In : Actes des Journées Internationales Neuro-Nîmes 90*. – Nîmes, 1990. EC2.

[61] Lakoff (G.). – *Women, Fire, and Dangerous Things : What Categories Reveal About the Mind*. – Chicago, University of Chicago Press, 1987.

[62] Langacker (R. W.). – An introduction to cognitive grammar. *Cognitive Science*, 10(1) , 1986.

[63] Launay (M.). – Effet de sens : produit de quoi ? *Langages*, 82, 1986, pp. 13–39.

[64] Le Goffic (P.). – *Ambiguïté linguistique et activité de langage*. – Thèse de doctorat d'état, Université Paris VII, 1981.

[65] Le Goffic (P.). – *Comme*, adverbe connecteur intégratif : éléments pour une description. *Travaux linguistiques du* CERLICO, 4, 1991, pp. 11–31.

[66] Lehuen (J.). – *Un modèle de dialogue dynamique et générique intégrant l'acquisition de la compétence linguistique*. – Thèse de doctorat, Université de Caen, 1997.

[67] Li (S.) et Economopoulos (P.). – *ActiveX/COM Control Programming*. – Birmingham, U.K., Wrox Press Ltd., 1997.

[68] Lyons (J.). – *Sémantique linguistique*. – Paris, Larousse, 1980.

[69] Mann (W.). – The anatomy of a systemic choice. *Discourse Processes*, 8(1) , 1985.

[70] Martin (R.). – *Pour une logique du sens*. – Paris, P.U.F., 1983.

[71] Martinet (A.). – Neutralisation et syncrétisme. *La Linguistique*, 4(1) , 1968, pp. 1–20.

[72] Martinet (A.). – *Eléments de linguistique générale*. – Paris, Armand Colin, 1980.

[73] McRoy (S.). – Using Multiple Knowledge Sources for Word Sense Discrimination. *Computational Linguistics*, 18(1) , 1992, pp. 1–30.

[74] Melçuk (I.) et Zolkovskij (A. K.). – Towards a functionning "meaning-text" model of language. *Linguistics*, 57, 1970, pp. 10–47.

[75] Miller (G. A.), Beckwith (R.), Fellbaum (C.), Gross (D.) et Miller (K.). – Introduction to WordNet : An on-line lexical database. *In : Five Papers on WordNet*, pp. 1–9. – 1993.

[76] Morlet (C.). – Variétés différentiables. *Encylopœdia Universalis*, 23, 1992, pp. 344–351.

[77] Morris (C.). – *Foundations of the theory of signs*, pp. 77–138. – University of Chicago Press, 1933.

[78] Nicolle (A.) et Beust (P.). – Anadia, un analogue machine de la mémoire. *In : Workshop "Les modèles de représentation : quelles alternatives ?"*. Association Ferdinand Gonseth. – Neufchâtel, 1997.

[79] Ogden (C. K.) et Richards (I. A.). – *The Meaning of Meaning*. – Londres, Routledge and Kegan Paul, 1923.

[80] Piaget (J.). – *Logique et Connaissance scientifique*. – Paris, Gallimard, 1967, *Pléiade*.

[81] Pottier (B.). – *Systématique des éléments de relation*. – Paris, Klincksiek, 1962.

[82] Pottier (B.). – Recherches sur l'analyse sémantique en linguistique et en traduction automatique. – Publication de la Faculté des Lettres et Sciences Humaines de Nancy, Série A, 1963.

[83] Pottier (B.). – *Linguistique générale. Théorie et description*. – Paris, Klincksiek, 1974.

[84] Prince (V.). – La pidginisation du français par le jargon américain de l'informatique. *In : Actes du 2ème colloque du GEPE*, pp. 41–50. – Strasbourg, 1986.

[85] Prince (V.). – *Sémantique récurrente dans les langues naturelles et actualisation expérimentale*. – Thèse de doctorat, Université Paris VII, Paris, 1986.

[86] Prince (V.). – GLACE : Un système d'aide à la compréhension d'éléments lexicaux inducteurs d'ambiguïté. *In : Actes de RFIA-91*. – Lyon, 1991.

[87] Prince (V.). – *Notes sur l'évaluation de la réponse dans TEDDI : introduction d'une relation d'équivalence sur la synonymie relative*. – Rapport interne n 91-20, LIMSI-CNRS, Université Paris XI, 1991.

[88] Prince (V.). – *A discrete approach based on logic simulating continuity in lexical semantics*, pp. 189–204. – Volume 19 in Fuchs et Victorri [40].

[89] Prince (V.). – Interpreting Common Words in Context : A Symbolic Approach. *In : Proceedings of the European Conference on Artificial Intelligence*. – Amsterdam, 1994.

[90] Prince (V.). – La polysémie des noms courants : vers une modélisation par conditions suffisantes. *In : Actes des Journées de Sémantique Lexicale Brestoises*. – ENST Bretagne, 1996.

[91] Prince (V.). – *Vers une informatique cognitive dans les organisations. Le rôle central du langage.* – Paris, Masson, 1996.

[92] Prince (V.). – An Intelligent Lexicon for Contextual Word Sense Discrimination. *Applied Intelligence*, 7, 1997, pp. 125–146.

[93] Prince (V.) et Bally-Ipsas (R.). – *Un algorithme pour le transfert de règles pragmatiques dans le processus complexe GLACE*. – Rapport interne du LIMSI n 91-17, Orsay, Université Paris XI, 1991.

[94] Pustejovsky (J.). – The generative lexicon. *Computational Linguistics*, 17, 1991, pp. 409–441.

[95] Pustejovsky (J.) et Boguraev (B.). – Lexical knowledge representation and natural language processing. *Artificial Intelligence*, 63, 1993, pp. 193–223.

[96] Putnam (H.). – *Representations and Reality*. – Cambridge (MA), MIT Press, 1988.

[97] Quillian (R.). – Semantic memory. *In : Semantic Information Processing*, éd. par Minsky (M.), pp. 227–270. – Cambridge (MA), MIT Press, 1968.

[98] Rady (M.). – *L'ambiguïté du langage naturel est-elle la source du non-déterminisme des procédures de traitement ?* – Thèse de doctorat d'état, Université Paris VI, Paris, 1983.

[99] Rastier (F.). – *Sémantique interprétative*. – Paris, P.U.F., 1987.

[100] Rastier (F.). – *Sémantique et recherches cognitives*. – Paris, P.U.F., 1991.

[101] Rastier (F.), Cavazza (M.) et Abeillé (A.). – *Sémantique pour l'analyse. De la linguistique à l'informatique*. – Paris, Masson, 1994.

[102] Rich (E.), Barnett (J.), Wittenburg (K.) et Wroblewski (D.). – Ambiguity procrastination. *In : AAAI Proceedings*, pp. 571–576.

[103] Rieger (B. B.). – Semantic relevance and aspect dependency in a given subject domain. *In : Proceedings of COLING'84*, pp. 298–301.

[104] Robba (I.). – Raisonnement par analogie et validité des résultats. *In : ECCOS, Colloque européen de sciences cognitives*. – Orsay, 1992.

[105] Rosch (E.) et Lloyd (B.). – *Cognition and Categorization*. – Hillsdale, Erlbaum, 1978.

[106] Sabah (G.). – *L'I.A. et le langage*. – Paris, Hermès, 1989. 2 vol.

[107] Sabah (G.). – Caramel : A computational model of natural language understanding using a parallel implementation. *In : Proceedings of ECAI'90*. – Stockholm, Suède, 1990.

[108] Sabah (G.). – Caramel : a flexible model for interaction between cognitive processes underlying natural language understanding. *In : Proceedings of COLING'90*. – Helsinki, Finlande, 1990.

[109] Sabah (G.). – Natural Language Understanding and Consciousness. *In : Actes de AISB'95*. – Sheffield, R. U., 1995. Workshop on "Reaching for mind".

[110] Sabah (G.). – The fundamental role of pragmatics in Natural Language Understanding systems. *In : Actes de IEEE International Conference on Tools with Artificial Intelligence*, pp. 258–265. – Boston, 1997.

[111] Sabah (G.). – The "Sketchboard" : A Dynamic Interpretative Memory and its Use for Spoken Language. *In : Actes de Eurospeech'97*, pp. 617–620. – Rhodes, 1997.

[112] Sapir (E.). – *Linguistique*. – Paris, Editions de Minuit, 1968.

[113] Schank (R.). – Conceptual dependency : A theory of natural language understanding. *Cognitive Psychology*, 3(4) , 1973, pp. 552–630.

[114] Small (S.). – *Word Expert Parsing : a theory of distributed word-based natural language understanding*. – Thèse de doctorat, University of Maryland, Dept of Computer Science, 1980.

[115] Small (S.), Cottrell (G.) et Tanenhaus (M.) (édité par). – *Lexical Ambiguity Resolution : Perspectives from Psycholinguistics, Neuropsychology, and Artificial Intelligence*. – San Mateo (CA), Morgan Kaufmann Publishers Inc., 1988.

[116] Sowa (J.). – *Conceptual Structures : processing in mind and machine*. – Reading (MA), Addison-Wesley, 1984.

[117] Sperber (D.) et Wilson (D.). – *La pertinence*. – Paris, Editions de Minuit, 1989.

[118] Stallard (D.). – The Logical Analysis of Lexical Ambiguity. *In : Proceedings of the 25th Annual Meeting of the ACL*, pp. 179–185. – Stanford University, California, 1987.

[119] Surcin (S.). – *Didactique du Génie Logiciel : l'apprentissage du modèle à objets*. – Mémoire de DEA de Sciences Cognitives, LIMSI, Université Paris XI, Orsay, 1994.

[120] Surcin (S.). – A Micro-Semantic Model for Coping With Common Words Ambiguity. *In : Proceedings of SPECOM-96 (International Conference on Speech and Computer)*. – St-Petersbourg, Russie, 1996.

[121] Surcin (S.). – A Semi-Symbolic Cognitive Model of Usage Polysemy. *In : Proceedings of ECCM-98 (European Conference on Cognitive Modelling)*. – Nottingham, Royaume-Uni, 1998.

[122] Surcin (S.). – Handling Lexical Ambiguity in Cases of Usage Polysemy. *In : Proceedings of CLUK-98 (Colloquium of the Association for Computational Linguistics in the United Kingdom)*. – Sunderland, Royaume-Uni, 1998.

[123] Swinney (D.). – Lexical access during sentence comprehension : (Re)Consideration of context effects. *Journal of Verbal Learning and Verbal Behavior*, 18(6) , 1979, pp. 645–659.

[124] Tesnière (L.). – *Eléments de syntaxe structurale*. – Paris, Klincksiek, 1969.

[125] Thom (R.). – *Stabilité structurelle et morphogénèse*. – Paris, Ediscience, 1972.

[126] Thom (R.). – *Modèles mathématiques de la morphogénèse*. – Paris, Christian Bourgeois, 1974.

[127] Thom (R.). – *Esquisse d'une sémiophysique*. – Paris, Interéditions, 1988.

[128] Vanderwende (L.). – Ambiguity in the acquisition of lexical information. *In : Proceedings of the AAAI 1995 Spring Symposium Series*, pp. 174–179. – Working notes of the symposium on representation and acquisition of lexical knowledge.

[129] Victorri (B.). – Modéliser la polysémie. *t. a. informations*, 29, 1988, pp. 21–42.

[130] Victorri (B.) et Fuchs (C.). – *La polysémie, construction dynamique du sens*. – Paris, Hermès, 1996.

[131] Vignaux (G.). – Catégorisation et schématisation. Des arguments au discours. *In : Sémantique et cognition*, éd. par Dubois (D.). – Paris, Editions du CNRS, 1991.

[132] Vygotsky (L. S.). – *Thought and Language*. – Cambridge (MA), MIT Press, 1962.

[133] Werner (O.). – Cultural knowledge, language and world view. *In : Cognition, a Multiple View*, éd. par Garvin (P.). – New York, Spartan, 1970.

[134] Weydt (H.). – Le concept d'ambiguïté en grammaire transformationnelle-générative et en linguistique fonctionnelle. *La Linguistique*, 2(8) , 1972, pp. 41–72.

[135] Wilks (Y.). – A preferential pattern-seeking semantics for natural language inference. *Artificial Intelligence*, 6, 1975, pp. 53–74.

[136] Wilks (Y.). – Making preferences more active. *Artificial Intelligence*, 11, 1978, pp. 197–223.

[137] Winograd (T.). – Towards a procedural understanding of semantics. *Revue internationale de philosophie*, 30(117-118) , 1976, pp. 261–303.

[138] Woods (W. A.). – Semantics and quantification in natural language question answering. *Advances in Computers*, 17, 1977, pp. 1–87.

[139] Woods (W. A.). – Procedural semantics as a theory of meaning. *In : Elements of Discourse Understanding*, éd. par Webber (B.), Joshi (A.) et Sag (I.), pp. 301–334. – Cambridge (MA), Cambridge University Press, 1981.