

Apprentissage incrémental pour la construction de bases lexicales évolutives : application en désambiguïsation d'entités nommées

Thomas Girault

▶ **To cite this version:**

Thomas Girault. Apprentissage incrémental pour la construction de bases lexicales évolutives : application en désambiguïsation d'entités nommées. Traitement du texte et du document. Université Rennes 1, 2010. Français. NNT : . tel-00867236

HAL Id: tel-00867236

https://theses.hal.science/tel-00867236

Submitted on 27 Sep 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre : 4083 ANNÉE 2010

UNIVERSITÉ DE RENNES 1

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Informatique

École doctorale MATISSE

présentée par

Thomas Girault

préparée dans le cadre d'une convention CIFRE entre
Orange Labs et l'unité de recherche UMR 6074 IRISA
Institut de Recherche en Informatique et Systèmes Aléatoires
Composante universitaire : IFSIC

Apprentissage incrémental pour la construction de bases lexicales évolutives : application en désambiguïsation d'entités nommées

Thèse soutenue à Rennes le 18 juin 2010

devant le jury composé de :

Amedeo NAPOLI
Directeur de recherche CNRS, LORIA
Rapporteur

Bernard VICTORRI
Directeur de recherche CNRS, LaTTiCe
Rapporteur

Marie-Odile CORDIER
Professeur à l'Université de Rennes 1, IRISA
Examinateur

Anne VILNAT
Professeur à l'Université Paris-Sud 11, LIMSI
Examinateur

Pascale SÉBILLOT
Professeur à l'INSA de Rennes, IRISA
Directrice de thèse

Olivier COLLIN
Ingénieur de recherche, Orange Labs
Encadrant industriel

Remerciements

Je tiens tout d'abord à remercier Pascale Sébillot, sans qui je n'aurais jamais pu concrétiser ce projet avec autant de richesse et de rigueur. Je voudrais lui exprimer toute ma gratitude pour la pertinence de ses conseils qui m'ont poussé à éclaircir et développer mes intuitions.

Je remercie Olivier Collin pour m'avoir permis de réaliser cette thèse au sein d'Orange Labs et de m'avoir accordé sa confiance dès le début de cette aventure. Par ailleurs, nos nombreux échanges ont été source d'inspiration et d'encouragement pour moi.

Je souhaite remercier tout particulièrement Amedeo Napoli et Bernard Victorri qui m'ont fait l'honneur de rapporter cette thèse ; leurs lectures attentives et leurs remarques constructives m'ont été d'une aide précieuse. Je remercie également Marie-Odile Cordier et Anne Vilnat, examinateurs de cette thèse, d'avoir accepté de jouer ce rôle.

Je tiens à citer l'Association Nationale de la Recherche Technique (ANRT) et Orange Labs qui ont participé au financement de cette thèse par le biais d'une bourse CIFRE.

Merci à tous mes anciens collègues des équipes LangNat et TSI ainsi qu'aux membres de l'équipe TexMex : leurs recherches ont été une source d'inspiration importante dans le développement de mes travaux. Je pense en particulier à Maxime Amblard, Alexis Bondu, Laurent Candillier, Vincent Claveau, Olivier Tardif et Gregory Smits pour leurs conseils et pour les discussions fructueuses qui ont largement contribué à faire progresser ma problématique scientifique. Je dois aussi mentionner Sergei Obiedkov, dont l'expertise a été essentielle pour aboutir aux contributions du deuxième chapitre de cette thèse.

Enfin, merci à tous ceux qui, de près ou de loin, m'ont encouragé et soutenu tout au long de ce travail : mes amis proches et ma famille ont toujours su me soutenir lors des moments difficiles. Merci à ma sœur Stéphanie qui m'a transmis son goût pour la linguistique et qui a su prêter attention à mes questions.

Ce travail est dédié à la mémoire de mon cousin Martin.

Table des matières

Introduction **1**

1 Construction de bases lexicales évolutives à partir de textes **7**
1.1 Représentations du sens et désambiguïsation sémantique automatique 8
1.1.1 Polysémie des unités lexicales 8
1.1.1.1 Notions de sémiotique pour le traitement automatique des langues 8
1.1.1.2 Ambiguïtés lexicales 10
1.1.1.3 Granularité de catégorisation 10
1.1.2 Cas particulier des entités nommées 12
1.1.2.1 Reconnaissance d'entités nommées 12
1.1.2.2 Ambiguïtés des entités nommées 12
1.1.2.3 Vers une une désambiguïsation fine des entités nommées 13
1.1.3 Désambiguïsation lexicale automatique 14
1.1.3.1 Exploitation du contexte 14
1.1.3.2 Principales stratégies de désambiguïsation 15
1.1.4 Discussion 15
1.2 Constitution de bases lexicales 16
1.2.1 Bases lexicales constituées manuellement 16
1.2.1.1 Exemples de bases lexicales généralistes 17
1.2.1.2 Ressources spécialisées dans le traitement des entités nommées 17
1.2.1.3 Limites des bases lexicales expertes 18
1.2.2 Acquisition supervisée de bases lexicales sur corpus 19
1.2.2.1 Acquisition guidée par des unités de sens prédéfinies 20
1.2.2.2 Acquisition guidée par des relations sémantiques prédéfinies 20
1.2.3 Construction automatique de bases lexicales distributionnelles 21
1.2.3.1 Extraction de contextes distributionnels 22
1.2.3.2 Représentation du contexte 23

1.2.3.3 Apprentissage non supervisé pour la structuration d'unités lexicales en classes sémantiquement homogènes . . . 24
1.2.4 Évolutivité des ressources lexicales . . . 26
1.3 Approches incrémentales pour la construction de bases lexicales évolutives . . . 28
1.3.1 Construction à partir de flux de données textuelles . . . 28
1.3.1.1 Notion de flux de données textuelles . . . 29
1.3.1.2 Applications et traitements sur des flux de données textuelles . . . 29
1.3.2 Fusion de connaissances dans les ontologies évolutives . . . 31
1.3.3 Partitionnement incrémental de données fournies en séquence 32
1.3.3.1 Analyse de concepts formels . . . 33
1.3.3.2 *Clustering* conceptuel incrémental . . . 36
1.3.3.3 *Clustering* incrémental exploitant la densité des données . . . 36
1.3.3.4 Discussion . . . 37
1.4 Conclusion . . . 38

2 Constitution d'une base lexicale hiérarchisée par un treillis de Galois 41
2.1 Treillis de Galois et traitement automatique des langues . . . 42
2.1.1 Treillis de relations issues de ressources lexicales existantes . . 43
2.1.2 Analyse de concepts formels pour la fouille de données textuelles 43
2.1.3 L'espace des versions en classification supervisée . . . 45
2.2 Constitution d'un contexte formel à partir de corpus . . . 46
2.2.1 Corpus de la campagne CoNLL-2003 . . . 47
2.2.2 Analyse en dépendances pour la constitution d'un contexte formel . . . 48
2.2.2.1 Extraction des dépendances syntaxiques des entités nommées . . . 49
2.2.2.2 Contexte formel des relations extraites . . . 50
2.2.3 Discussion . . . 51
2.3 Construction incrémentale à partir de données fournies en séquence . 52
2.3.1 Construction incrémentale à partir de contextes formels prédéfinis 54
2.3.1.1 Caractérisation des concepts dans un treillis en construction . . . 57
2.3.1.2 Intégration des attributs associés à un objet . . . 58
2.3.1.3 Exemple . . . 60
2.3.1.4 Construction incrémentale d'un treillis de Galois . . 64
2.3.2 Adaptations aux contextes formels présentés en séquence . . 65
2.3.2.1 Suppression d'un objet . . . 66

2.3.2.2 Exemple de suppression 67
2.3.2.3 Ajout d'une relation quelconque 69
2.3.2.4 Croissance du treillis 71
2.3.3 Bilan 72
2.4 Conclusion 73

3 Analyse des données linguistiques d'un treillis de Galois lexical 75
3.1 Structuration d'unités lexicales polysémiques 76
3.1.1 Treillis d'une relation entre des unités polysémiques 76
3.1.2 Règles d'implication entre niveaux conceptuels 78
3.1.3 Extraction de règles d'association 80
3.2 Polysémie et recouvrements des sens 84
3.2.1 Degré de polysémie des concepts dans un treillis 84
3.2.2 Critère distributionnel pour l'extraction de concepts sémantiquement homogènes 86
3.2.3 Recouvrement et continuité des sens 89
3.2.3.1 Métrique distributionnelle pour la construction d'un espace sémantique continu 89
3.2.3.2 Des cliques aux concepts formels 91
3.3 Représentation géométrique de treillis de Galois 92
3.3.1 Représentation vectorielle des concepts formels 93
3.3.2 Projection des concepts formels vers un espace réduit 95
3.3.3 Cartographie sémantique des concepts formels 97
3.3.4 Discussion 100
3.4 Conclusion 101

4 Exploitation de treillis de Galois en désambiguïsation non supervisée 103
4.1 La désambiguïsation vue comme une sélection d'unités de sens prédéfinies 104
4.1.1 Désambiguïsation par définitions de dictionnaire 105
4.1.2 Mesures de similarité taxonomique 106
4.1.3 Conclusion 107
4.2 Annotation conceptuelle d'unités lexicales 108
4.2.1 Annotation d'un objet en relation avec un attribut 109
4.2.1.1 Espace de recherche associé à un objet en relation avec un attribut 109
4.2.1.2 Règles d'association pour la sélection des concepts . 110
4.2.1.3 Exemple d'annotation conceptuelle 110
4.2.2 Annotation d'entités nommées en relation avec plusieurs attributs 112

4.2.2.1 L'annotation comme une recherche de concepts candidats dans un treillis . . . 112
4.2.2.2 Détermination du prototype d'un ensemble de concepts 113
4.2.2.3 Algorithme de désambiguïsation . . . 114
4.2.3 Annotation conceptuelle : amélioration de la robustesse . . . 116
4.3 Évaluation en cascade d'annotations conceptuelles . . . 117
4.3.1 Protocole d'évaluation en cascade . . . 118
4.3.1.1 Classification supervisée sur un corpus enrichi . . . 118
4.3.1.2 Critère de performance pour l'évaluation des systèmes de reconnaissance d'entités nommées . . . 119
4.3.1.3 Tests statistiques pour la comparaison de performances 122
4.3.2 Exemple d'annotation conceptuelle . . . 123
4.3.3 Résultat de l'évaluation en cascade . . . 125
4.3.4 Problèmes liés à l'évaluation en cascade . . . 127
4.4 Conclusion . . . 128

5 Propagation d'étiquettes sémantiques dans une base lexicale en construction 131
5.1 Classification de concepts par co-apprentissage . . . 133
5.1.1 Co-apprentissage à partir de données fournies en séquence . . . 134
5.1.1.1 Bipartition des données d'apprentissage . . . 134
5.1.1.2 Hypothèse d'indépendance pour des apprentissages sur des vues séparées . . . 135
5.1.1.3 Algorithme de co-apprentissage . . . 136
5.1.2 Co-classification de concepts dans un treillis en construction . . . 137
5.1.2.1 Intégration de relations co-classifiées à un treillis . . . 138
5.1.2.2 Classifieur bayésien naïf pour l'étiquetage d'un concept formel . . . 139
5.1.3 Expérimentations et résultats . . . 141
5.1.3.1 Performances des stratégies supervisées . . . 141
5.1.3.2 Performances des stratégies semi-supervisées . . . 144
5.1.4 Discussion . . . 147
5.2 Propagation d'étiquettes sémantiques dans un treillis . . . 148
5.2.1 Propagation d'étiquettes dans un graphe . . . 149
5.2.2 Adaptation au cadre des treillis . . . 152
5.2.2.1 Cartographie de concepts formels étiquetés . . . 152
5.2.2.2 Influence de la distance sur l'étiquetage des concepts 154
5.2.2.3 Détermination des matrices associées aux concepts . 155
5.2.3 Expérimentations et résultats . . . 157
5.2.4 Discussion . . . 159

5.2.4.1 Vers une prise en charge de l'évolution du treillis pendant la propagation 159
5.2.4.2 Indices structurels pour la détection d'exemples influents 160
5.3 Conclusion et perspectives 161

Conclusion **163**

Bibliographie **185**

Symboles et notations utilisés

Notations générales

$x \to y$	implication logique entre x et y
$P(A)$	probabilité de l'évènement A
$P(A\|B)$	probabilité de l'évènement A sachant B
$\underset{x}{\text{argmax}}\ f(x)$	valeur de x fournissant la valeur maximale de la fonction f
$\underset{x}{\text{argmin}}\ f(x)$	valeur de x fournissant la valeur minimale de la fonction f

Notations relatives aux treillis de Galois

$\mathbb{K}$	contexte formel
$\underline{\mathfrak{T}}_{\mathbb{K}}$	treillis de Galois associé à $\mathbb{K}$
$\mathfrak{T}_{\mathbb{K}}$	ensemble des concepts formels associés à $\mathbb{K}$
$\text{ext}(c)$	extension d'un concept formel c
$\text{int}(c)$	intension d'un concept formel c
$\text{ext}_{\uparrow}(c)$	extension héritée d'un concept formel c
$\text{int}^{\downarrow}(c)$	intension héritée d'un concept formel c
$\text{ext}_{\circlearrowleft}(c)$	extension propre d'un concept formel c
$\text{int}_{\circlearrowleft}(c)$	intension propre d'un concept formel c
$M_{\mathfrak{T}}$	matrice associée à l'ensemble des concepts $\mathfrak{T}$

Notations relatives à l'apprentissage automatique

$\mathcal{M}$	modèle prédictif
L	ensemble de données étiquetées (L pour *labelled*)
U	ensemble de données non étiquetées (U pour *unlabelled*)
$Labels$	ensemble des étiquettes possibles pour un exemple
$conf(A \to B)$	confiance de la règle $A \to B$
$sup(A \to B)$	support de la règle $A \to B$

Introduction

Depuis déjà quelques années, le volume des données échangées sur le réseau Internet s'accroît en permanence. Ces flux[1] de données numériques (actualités, transcriptions radiophoniques, messagerie électronique instantanée – pour ne prendre que des exemples de données textuelles) posent, au-delà des verrous technologiques, de nouvelles problématiques scientifiques au domaine du traitement automatique des langues. C'est la dimension dynamique du lexique observé dans ces données textuelles que nous aborderons dans cette thèse, en particulier par son emploi d'un vocabulaire en perpétuelle évolution, que ce soit au niveau de la création des mots que des sens de ceux existant déjà. Le changement des valeurs sémantiques que peuvent prendre un mot en sortant de son emploi habituel nécessite de nouveaux traitements de l'ambiguïté lexicale. La piste de recherche que nous avons explorée vise à prendre en compte le plus possible cette dynamique lexicale et propose une méthode pour construire des bases lexicales associées à des représentations du sens évolutives : celles-ci se réorganisent automatiquement chaque fois qu'un nouvel usage est détecté.

Depuis le milieu des années 1990, on assiste au développement de ressources lexicales électroniques qui définissent des unités de sens (étiquettes, définitions, concepts) pour décrire la sémantique d'unités lexicales (UL). Les systèmes de désambiguïsation qui exploitent ces bases lexicales (BL) mettent en œuvre des algorithmes capables d'associer des usages d'UL observés en corpus avec des unités de sens. La structure de ces BL peut aller du simple dictionnaire à une organisation beaucoup plus complexe de thésaurus ou d'ontologie (voir notamment les taxonomies définies dans WordNet [Miller, 1995] ou FrameNet [Baker *et al.*, 1998]) modélisant le sens des UL selon plusieurs niveaux de granularité. Certains systèmes de désambiguïsation [Resnik, 1999] doivent alors sélectionner un niveau de granularité sémantique approprié au contexte d'usage pour désambiguïser correctement les occurrences d'UL. Le résultat fourni doit être interprétable par des

1. Un flux de données arrive de manière séquentielle, élément par élément et implique habituellement que l'ensemble des données transmises est trop volumineux pour être stocké [Muthukrishnan, 2005]. Cette étude se focalise essentiellement sur l'aspect temporel du traitement. La question du stockage des données et les mécanismes d'oubli ne sont pas considérés pour le moment.

humains et réutilisable par des programmes informatiques.

La constitution manuelle de telles BL et leur manipulation pour des tâches de désambiguïsation sémantique reste toutefois problématique. Leur construction est une entreprise gigantesque qui devient hors de portée de l'expertise humaine [Pedersen, 2006] face à une quantité grandissante de documents dont le contenu se diversifie. Bien que ces BL atteignent aujourd'hui de grandes tailles, elles ne couvrent pas le vocabulaire employé dans les documents. Elles ne sont donc jamais exhaustives et demandent à être complétées pour garantir des résultats acceptables. Une problématique de désambiguïsation dans des flux nécessite des adaptations fréquentes à de nouveaux contextes : un enrichissement manuel de telles ressources ne semble donc absolument pas envisageable. De plus, les sens répertoriés dans les BL ne correspondent pas nécessairement aux usages rencontrés dans les corpus [Ide et Véronis, 1998].

Ces dernières années, le manque de connaissances expertes a été en partie comblé par des BL constituées automatiquement à partir de corpus grâce à des techniques d'apprentissage artificiel. Plutôt que se référer à un inventaire de sens prédéfinis, certaines approches dites « distributionnelles » [Grefenstette, 1994; Bouaud *et al.*, 1997; Lin et Pantel, 2001] appliquent des méthodes d'apprentissage sur de très grands corpus textuels pour constituer automatiquement des classes d'usages d'UL qui ont des sens proches. Ces méthodes d'apprentissage sont « non supervisées », c'est-à-dire que l'acquisition des classes (ou *clusters*) procède de façon émergente à partir du corpus sans être guidée ni par une ressource lexicale préétablie, ni par un étiquetage sémantique du corpus. Cette émergence est le plus souvent fondée sur l'hypothèse harrisienne [Harris, 1968] selon laquelle des UL apparaissant dans des contextes similaires tendent à avoir des sens proches. Les classes inférées sont donc assimilables à des unités de sens qui composent une BL exploitable en désambiguïsation.

Les unités de sens résultant d'une analyse distributionnelle sont le plus souvent de simples classes qui ne modélisent pas différents niveaux de granularité sémantique. Il apparaît pourtant justifié de désambiguïser une UL avec des sens plus ou moins fins, en fonction de son contexte d'usage. Il serait donc intéressant que les structures qui modélisent des unités de sens dans une BL soient conçues pour prendre en compte différents niveaux de granularité sémantique. Une base lexicale de granularité variable s'accorde d'ailleurs bien avec l'objectif d'évolutivité. Lors d'une analyse distributionnelle, l'ensemble non structuré de *clusters* n'est généralement pas conçu pour évoluer. En effet, l'introduction progressive de nouvelles connaissances est susceptible de perturber les subdivisions de sens existantes en les séparant en des sens plus spécifiques ou en les fusionnant en des sens plus généraux. L'intégration à un modèle de connaissance préexistant serait donc facilitée par une représentation des sens sur plusieurs niveaux de granularité disposant des *clusters* dans une structure hiérarchisée.

De plus, l'intégration de connaissance à des *clusters* existants est aussi limitée par les algorithmes d'apprentissage non supervisé habituellement utilisés en analyse distributionnelle. Ces derniers adoptent généralement un apprentissage *batch* (ou « *one shot* » [Cornuéjols, 2005]) qui exige que toutes les données soient disponibles pour établir les classes distributionnelles d'un corpus. L'apprentissage *batch* ne permet donc pas de traiter les données de manière séquentielle et ni d'intégrer de nouvelles connaissances à une BL distributionnelle déjà figée. L'apparition d'éléments nouveaux perturbe l'organisation des données, nécessitant le plus souvent de procéder à de nouveaux calculs sur l'intégralité des données. D'un point de vue calculatoire, il n'est pas réaliste d'envisager un nouvel apprentissage sur toutes les données lorsque l'on veut prendre en compte de nouvelles connaissances apparaissant dans un flux.

Contrairement à l'apprentissage *batch*, les techniques d'apprentissage incrémental sont capables d'apprendre en continu, sans avoir reçu initialement l'ensemble des données, selon un processus dynamique et historique. Elles seraient donc plus aptes à appréhender certains mécanismes d'évolution du sens. Les phénomènes d'adaptation continue qu'elles mettent en œuvre tiennent compte d'une dynamique complexe qui nous semble corrélée à la question de l'émergence d'unités de sens lors de l'acquisition du lexique par des machines.

Le but de cette thèse est d'étudier la mise au point d'algorithmes et de structures de données permettant d'automatiser la construction d'une base lexicale répertoriant des unités lexicales observées dans des flux de données textuelles. La base lexicale visée possède certaines propriétés fondamentales. À l'instar des techniques d'analyse distributionnelle, cette base lexicale acquise doit être constituée automatiquement à partir d'usages d'unités lexicales observés dans des textes. Elle doit privilégier une structuration hiérarchisée explicitant différents niveaux de finesse dans la description sémantique des unités lexicales. Pour faciliter l'interprétation, nous cherchons également à préciser le contexte d'usage qui explique l'émergence des unités de sens de la base lexicale à partir de textes. Cette base doit aussi permettre une réutilisation de ces unités de sens pour des tâches de désambiguïsation lexicale. Enfin pour modéliser le caractère évolutif du sens, la base lexicale doit être construite à partir de flux en adoptant une démarche d'apprentissage incrémental.

La solution élaborée s'inscrit dans le cadre de l'apprentissage incrémental non supervisé. Parmi les différentes méthodes existantes, nous montrons l'intérêt de l'*analyse de concepts formels* (ACF) [Barbut et Monjardet, 1970; Ganter et Wille, 1999], technique symbolique qui n'a pas encore été utilisée dans un cadre de désambiguïsation lexicale. Le treillis de Galois qui résulte de cette analyse fournit une structuration hiérarchique de concepts inférés à partir d'usages d'UL dans un corpus non étiqueté sémantiquement. Ce treillis est construit incrémentalement, permettant ainsi d'intégrer progressivement et automatiquement de

nouvelles connaissances. Ces propriétés se révèlent particulièrement avantageuses face aux BL réputées peu évolutives. Ce caractère évolutif permet également d'appréhender des UL telles que les entités nommées (EN) dont la sémantique est susceptible d'évoluer au cours du temps : ces UL font référence à des objets du monde dynamiques et instables tels que des personnes, des lieux ou des organisations. Une base lexicale doit également être capable d'adapter sa structuration pour intégrer de nouvelles EN apparaissant dans un flux. Les systèmes actuels de reconnaissance des EN reposent habituellement sur une classification prédéfinie et peu précise. Le choix des EN comme objet d'étude et d'illustration se prête ainsi plutôt bien à la question de l'évolution du sens et à l'intégration de connaissances dans des BL. L'utilisation des treillis nous permet d'étudier les caractéristiques d'une classification fine inférée à partir de corpus, de suivre les évolutions de cette classification et de désambiguïser les EN observées dans des flux de texte. Notons toutefois que le cadre que nous développons ici est tout à fait transposable au traitement d'autres UL et que ce choix des EN est aussi motivé par des questions applicatives : les EN représentent près de 30% des requêtes sur le Web [Artiles *et al.*, 2007] et environ 10% du contenu de textes journalistiques [Coates-Stephens, 1992].

Les contributions principales de cette thèse sont les suivantes. Tout d'abord, nous montrons que la structure de treillis construite par ACF à partir d'un corpus est assimilable à une BL modélisant la polysémie lexicale. La représentation élaborée tient compte des aspects symboliques et graduels du sens en structurant des UL sur plusieurs niveaux de granularité et en assimilant les recouvrements entre unités de sens à des propriétés de continuité. Notre deuxième apport consiste à exploiter cette nouvelle représentation pour une tâche de désambiguïsation lexicale. Dans cette perspective, nous proposons une méthodologie d'annotation conceptuelle non supervisée capable d'étiqueter en contexte les unités lexicales à désambiguïser avec des concepts formels. Enfin, notre troisième contribution concerne les aspects incrémentaux mis en œuvre pour constituer une BL évolutive. À cet égard, nous avons élaboré un algorithme incrémental original pour construire une BL capable de faire évoluer progressivement sa structure en s'adaptant aux données textuelles d'un flux. Cette manipulation de données fournies sous une forme séquentielle nous permet de faire progresser les techniques d'acquisition existantes vers une plus grande capacité d'adaptation aux changements. Par ailleurs, cette construction est couplée à des méthodes de classification semi-supervisée exploitant un petit nombre d'EN étiquetées d'un corpus d'apprentissage. Ces méthodes sont capables d'exploiter la structure d'une BL pour propager l'étiquetage connu d'EN à d'autres unités lexicales employées pour la construction incrémentale de la BL. Ce couplage nous permet de mieux comprendre et modéliser la dynamique d'acquisition du lexique par des machines.

Ce mémoire développe les idées énoncées dans cette introduction en cinq cha-

pitres.

Le premier a pour vocation d'introduire les principaux concepts théoriques auxquels nous nous référons et de situer notre approche de façon contrastive. Il constitue un état de l'art sur la constitution automatique de bases lexicales à partir de textes. De plus, pour poser les bases de notre réflexion sur l'évolution du sens, nous y examinons différents modèles de *clustering* incrémental capables de traiter des flux de données textuelles.

Le deuxième chapitre reprend une méthodologie existante pour construire à partir de corpus une base lexicale hiérarchisée par un treillis de Galois. Les données utilisées dans nos expérimentations sont issues d'une extraction préalable d'unités lexicales rattachées syntaxiquement à des entités nommées dans des corpus. Une analyse de concepts formels sur ces données permet d'inférer un ensemble de concepts regroupant des EN en relation avec des contextes syntaxiques. Un algorithme incrémental original est présenté pour construire un treillis de concepts à partir de données fournies en séquence.

L'objectif du troisième chapitre est de mettre en évidence les propriétés linguistiques de la base lexicale constituée, pour valoriser ses intérêts pour modéliser la polysémie lexicale. Afin de réaliser cet objectif, nous examinons l'aspect symbolique et structurel du treillis au regard d'outils d'analyse de données statistique s'appuyant sur l'aspect fréquentiel des données. Les différents niveaux de granularité du treillis témoignent d'aspects symboliques tandis que les recouvrements entre concepts formels sont assimilables à des propriétés de continuité. Le modèle de base lexicale obtenu combine ainsi des points de vues numériques et symboliques dont la complémentarité a déjà été valorisée autant en apprentissage artificiel [Sébillot, 2006] que pour la représentation de lexiques sémantiques [Victorri et Fuchs, 1996].

Le quatrième chapitre est consacré à l'exploitation d'un treillis de Galois constitué à partir de corpus pour enrichir la représentation de ce dernier par une annotation sémantique. Considérés comme des unités de sens contextualisées, les concepts formels sont exploités en désambiguïsation sémantique non supervisée. Des méthodes de désambiguïsation lexicale connues, utilisant des ressources lexicales électroniques préexistantes, sont adaptées à la structure de treillis de Galois. La méthode d'annotation conceptuelle proposée fournit un enrichissement du corpus dont l'intérêt est quantifiable à travers une tâche de classification supervisée des EN, dans le cadre d'une évaluation en cascade [Candillier *et al.*, 2006]. Cette approche, bien que valorisée par l'expérimentation, n'exploite toutefois pas au mieux les propriétés des treillis. Les différentes granularités de la BL sont en effet mal considérées, surtout lorsque que sa taille (c'est-à-dire le nombre de concepts) est importante. Par ailleurs, en imposant de juger une BL figée, l'architecture séquentielle de la cascade limite fortement la prise en compte de l'incrémentalité de la BL : une séquence de processus lourde doit être relancée pour évaluer

l'influence d'une nouvelle intégration dans la BL sur les performances du système.

C'est pourquoi nous proposons, dans le dernier chapitre, une solution qui vise à coupler la classification supervisée d'EN avec la construction de la base lexicale dans un cadre d'apprentissage incrémental. Ce couplage consiste à enrichir la description des concepts formels d'un treillis en construction avec l'étiquetage des EN issues d'un corpus d'apprentissage. Le treillis enrichi est alors capable de produire directement une double annotation des EN dans le corpus : une annotation conceptuelle (non supervisée) complétée par un étiquetage sémantique (supervisé) « classique ». Notre méthodologie d'enrichissement s'appuie sur deux stratégies d'apprentissage semi-supervisé étiquetant des concepts formels obtenus de manière non supervisée par ACF. La première utilise un algorithme de co-apprentissage [Blum et Mitchell, 1998] (de l'anglais *co-training*) alors que la seconde emploie l'algorithme *label propagation* [Zhu et Ghahramani, 2002] tenant compte de la granularité variable du treillis ainsi que des distances entre concepts formels.

Chapitre 1

Construction de bases lexicales évolutives à partir de textes

Des ressources électroniques de grande taille (dictionnaires informatisés, thésaurus, ontologies) sont nécessaires pour décrire la sémantique des unités lexicales (UL) utilisées dans des textes. Ces bases lexicales sont particulièrement utiles pour les applications du traitement automatique des langues (TAL) (recherche d'information, traduction automatique...) se confrontant à l'ambiguïté lexicale et devant pourtant déterminer le sens considéré d'une UL donnée. Les BL doivent exprimer les différents sens de cette unité — autrement dit, modéliser sa polysémie — afin d'être exploitables pour sa désambiguïsation automatique. Par ailleurs, les BL actuelles, constituées manuellement ou automatiquement, ne sont pas conçues pour intégrer facilement des nouveautés qui se présentent dans des flux de données textuelles. Le traitement en séquence de ces flux impose pourtant de faire évoluer ces BL pour prendre en compte l'apparition de nouvelles UL, l'émergence de sens inédits et le changement de sens déjà établis.

La problématique initiale de cette thèse est de mettre au point des algorithmes et des structures de données pour automatiser la construction d'une BL répertoriant des UL observées dans des flux de données textuelles. Les propriétés souhaitées pour cette BL sont les suivantes. Elle doit d'abord pouvoir faire émerger automatiquement des unités de sens en observant des usages d'unités lexicales dans des corpus. Nous cherchons ensuite à répertorier ces unités lexicales sur plusieurs niveaux de granularité sémantique. Nous devons également expliciter les contextes d'usage d'unités lexicales faisant émerger des unités de sens afin d'exploiter ces sens en désambiguïsation lexicale. Enfin, la BL et ses unités de sens doivent être capables de s'adapter à de nouveaux usages d'UL observés en corpus.

Ce chapitre constitue un état de l'art de la construction automatique de BL au regard des propriétés que nous venons d'énoncer. La première section se rapporte à la modélisation de la polysémie lexicale et aux méthodes de désambiguïsation.

La deuxième dresse un panorama des principales BL existantes et des travaux dédiés à la construction automatique de BL à partir de corpus à l'aide de techniques d'apprentissage artificiel. Dans la troisième section, nous examinons différentes stratégies permettant de constituer automatiquement des BL évolutives ; nous étudions en particulier des méthodes incrémentales d'apprentissage artificiel capables d'adapter des structures de données pour y intégrer de nouvelles connaissances.

1.1 Représentations du sens et désambiguïsation sémantique automatique

L'interprétation d'une UL peut varier en fonction de son contexte d'usage dans un texte. En TAL, l'ambiguïté lexicale est omniprésente et sa résolution exige de modéliser les différentes interprétations possibles d'une UL au sein de BL, puis de sélectionner l'une d'entre elles lors d'une procédure de désambiguïsation. Les BL décrivent d'une manière plus ou moins détaillée les différentes interprétations ou « sens » des mots. En informatique, cette notion de sens peut être appréhendée de diverses façons et donne lieu à de nombreuses représentations.

Cette section amorce la problématique de la représentation du sens dans les BL exploitées en désambiguïsation sémantique automatique. Nous introduisons en premier lieu le problème de l'ambiguïté lexicale en TAL, qui est ensuite examiné à l'aune du cas particulier des entités nommées. Nous étudions enfin brièvement les principales stratégies de désambiguïsation sémantique automatique.

1.1.1 Polysémie des unités lexicales

De nombreuses tâches du TAL exigent d'annoter les mots d'un texte avec des étiquettes correspondant à des « sens ». Cette association entre mot et sens s'inscrit dans la description traditionnelle du signe linguistique [de Saussure, 1995].

1.1.1.1 Notions de sémiotique pour le traitement automatique des langues

On considère généralement que le signe linguistique est composé d'un signifiant, d'un signifié et éventuellement d'un référent. Le « triangle sémiotique » (*cf.* figure 1.1) est habituellement employé pour représenter ces notions :

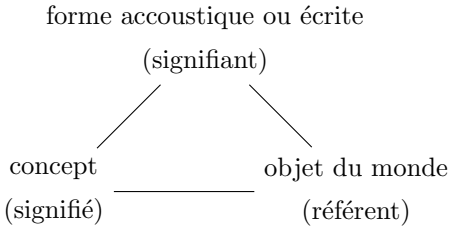


FIG. 1.1 – Représentation du signe linguistique dans un triangle sémiotique

Les définitions des termes « signifiant », « signifié » et « référent » sont l'objet de nombreuses discussions dans les domaines de la linguistique, de la sémiotique et de la représentation des connaissances [Sowa, 2000]. Nous pouvons toutefois préciser ces notions pour notre cadre de TAL.

Le signifiant peut correspondre à une forme acoustique, un geste, un mot ou n'importe quelle autre expression. Dans notre cadre de traitement automatique de textes, les signifiants manipulés par les programmes sont des **unités lexicales** (UL). Les UL correspondent à des formes graphiques telles que des mots (ou *tokens*) ou des petits successions de mots du vocabulaire d'un corpus textuel.

Les programmes informatiques n'ont pas directement accès aux signifiés, c'est-à-dire aux sens ou aux concepts (représentation mentale du sens). En revanche, le TAL a développé de nombreuses modélisations sémantiques pour représenter les signifiés. Les étiquettes sémantiques et les concepts utilisés dans les systèmes de désambiguïsation sont des signifiants : ils correspondent en fait à des symboles informatiques ou des structures de données se substituant aux signifiés.

Encore une fois, un modèle informatique est nécessaire pour représenter les référents. Ceux-ci correspondent à des objets ou des idées identifiables qui possèdent un ancrage dans le monde réel ou imaginaire. Dans un texte, les référents peuvent être représentés par des unités lexicales (c'est d'ailleurs le cas des entités nommées que nous allons étudier plus loin) porteuses d'une sémantique (un sens peut se rapporter à un référent).

L'exemple suivant illustre les notions de signifiant, signifié et objet :

– *Il a prétendu être passé au vert alors qu'il venait de griller le feu.*

Le fait d'avoir placé « vert » dans le contexte de « passer » et de « griller le feu » nous permet de comprendre que « vert » est associé à la fois à un référent (l'objet « feu tricolore » qui se rapporte au signifié « signalisation routière ») et un signifié (la couleur verte symbolise une autorisation de circuler).

1.1.1.2 Ambiguïtés lexicales

L'ambiguïté lexicale est un phénomène linguistique général souvent rencontré par les humains dans leur quotidien mais qui ne gène pas leur compréhension [1]. Même si nous ne parvenons pas toujours à définir exactement chaque mot d'un énoncé, nous sommes en général capables de le comprendre dans sa globalité lorsque le contexte d'énonciation est suffisamment explicite. En revanche en informatique, l'ambiguïté lexicale reste un problème central dans de nombreuses tâches impliquant la manipulation de données en langue naturelle, notamment en recherche d'information, en extraction d'information ou traduction automatique.

Les signes linguistiques peuvent avoir diverses interprétations dans des contextes différents. Lorsque les unités lexicales « passer », « vert », « griller » ou « feu » sont considérées ensemble, il nous est relativement facile de comprendre la situation. En revanche, l'interprétation est plus délicate lorsque les unités lexicales sont prises indépendamment car elles peuvent avoir de multiples sens et se référer à différents objets. L'ambiguïté lexicale peut ainsi porter sur le signifié ou sur le référent.

Dans le cas d'une ambiguïté qui porte sur le signifié, on parle de polysémie lexicale : une unité lexicale polysémique possède plusieurs significations possibles lorsque le contexte n'est pas suffisamment précisé. La plupart des unités lexicales sont polysémiques : il suffit de consulter un dictionnaire pour se rendre compte que chacune de ses entrées lexicales peut être associée à plusieurs définitions. Par exemple « vert » qui symbolisait une autorisation dans l'exemple précédent peut prendre un autre sens dans le même discours :

– *Il s'est mis au vert depuis son accident.*

Dans le cas de l'ambiguïté référentielle (ou polyréférentialité [Poibeau, 2005]), une unité lexicale peut désigner des idées ou des objets/entités différents.

En fonction de son contexte d'usage la sémantique d'une UL doit être modélisée avec des unités de sens plus ou moins fines. Ces sens plus ou moins généraux témoignent d'une granularité variable des interprétations.

1.1.1.3 Granularité de catégorisation

La palette de couleur ci-dessous (*cf.* figure 1.2) illustre le phénomène de *granularité* des sens rencontré en sémantique lexicale. Cette palette peut être discrétisée selon plusieurs échelles : on peut partir de l'ensemble de la palette en délimitant des grandes zones de couleurs, pour construire un référentiel de base : rouge, vert, bleu, jaune, orange, marron... Ces zones peuvent être discrétisées à un niveau plus fin. Les couleurs centrales ou *prototypiques* sont les meilleurs re-

1. Pour Rastier [2004], la polysémie est un artéfact de la linguistique.

présentants des zones. Des couleurs intermédiaires situées aux frontières des zones partagent des propriétés en mélangeant des couleurs prototypiques avec plus ou moins d'intensité. Ces couleurs intermédiaires ont un statut ambigu par rapport au référentiel de base. Par exemple, en mélangeant un bleu et un vert, on peut obtenir un bleu turquoise, un vert d'eau qui réalise un continuum entre le bleu et le vert. Le cyan est pour certains bleu et pour d'autres vert.

En réalité, dans une image, certaines couleurs apparaissent très rarement alors que certaines familles de couleurs sont nettement plus représentées. La palette de couleurs représentative d'une image aura donc tendance à détailler davantage les teintes les plus présentes et que l'on a le plus besoin de différencier.



FIG. 1.2 – Exemple de discrétisation des couleurs

Ce que nous cherchons à montrer avec des catégories de couleurs est comparable à ce qui se passe avec des unités de sens. Comme pour les couleurs, des unités lexicales peuvent être associées à des sens plus ou moins fins. Selon son contexte d'usage, une UL peut avoir un ou plusieurs sens généraux qui s'affinent dans des contextes spécifiques. Certaines unités de sens se recouvrent car elles ont des caractéristiques communes ou peuvent désigner des exemples communs.

En sémantique lexicale [Polguère, 2008], des catégories sémantiques plus ou moins générales (respectivement spécifiques) respectent une relation d'*hyperonymie* (respectivement *hyponymie*). L'hyperonymie est une relation entre une catégorie générale et ses sous-catégories spécifiques appelées hyponymes. Elle est exprimable en extension et en intension. L'extension est une énumération de l'ensemble des membres d'une même catégorie. L'intension est l'ensemble des propriétés partagées par les membres d'une même catégorie. Une même unité sémantique peut avoir plusieurs hyperonymes (par exemple, une église est à la fois un bâtiment et un lieu sacré).

Nous examinons à présent les questions de l'ambiguïté lexicale et de la granularité de catégorisation au regard d'unités lexicales d'un type particulier : les entités nommées (EN).

1.1.2 Cas particulier des entités nommées

La tâche de reconnaissance et de catégorisation des entités nommées (EN) est aujourd'hui primordiale pour de nombreuses applications. L'identification des entités nommées inclut traditionnellement trois types d'expressions : les noms propres, les expressions temporelles et les expressions numériques.

1.1.2.1 Reconnaissance d'entités nommées

Un système de reconnaissance des EN doit résoudre deux problèmes : d'une part le repérage des séquences de mots associées à des EN dans un texte, d'autre part, une tâche de classification des séquences ainsi repérées en fonction d'une classification préétablie. De nombreux travaux ont porté sur cette tâche de reconnaissance des EN dans des textes journalistiques, notamment dans le cadre des conférences MUC [2] [Grishman et Sundheim, 1996; Hirschman et Chinchor, 1998] ou des campagnes d'évaluation CoNLL [3] [Tjong Kim Sang et De Meulder, 2003]. En France, la campagne ESTER [4] proposant l'évaluation de systèmes de transcription d'émissions radiophoniques en langue française, a enrichi les transcriptions par un ensemble d'informations annexes dont le marquage des EN.

Les systèmes de reconnaissance des EN appliqués sur des dépêches journalistiques atteignent généralement d'assez bonnes performances, avec un taux combiné de rappel et de précision supérieur à 0,90. Il faut noter toutefois que pour atteindre de telles performances, la catégorisation effectuée s'appuie sur un jeu d'étiquettes sémantiques prédéfinies dont le grain est généralement assez rudimentaire : par exemple, les campagnes d'évaluation les plus connues se limitent à seulement quatre étiquettes (« personne », « lieu », « organisation » et « autre »). La tâche de reconnaissance et catégorisation d'EN s'oriente donc aujourd'hui vers de nouveaux défis visant à établir une classification plus fine et une désambiguïsation des ces entités.

1.1.2.2 Ambiguïtés des entités nommées

Comme la plupart des UL, les EN sont potentiellement ambiguës. Les exemples suivants illustrent différentes ambiguïtés auxquelles nous sommes confronté.

1. *commentaire de Robert Smith sur le système harmonique de John Harrison*
2. *le chanteur Robert Smith, leader du groupe The Cure*
3. *concert de Yannick Noah au Zénith*
4. *victoire de Yannick Noah à Roland Garros*

2. *Message Understanding Conferences.*
3. *Conference on Natural Language Learning.*
4. Évaluation des systèmes de transcription enrichie d'émissions radiophoniques.

5. *victoire d'Arnold Schwarzenegger en Californie*

Les exemples (1) et (2) illustrent un cas de « polyréférentialité » [Poibeau, 2005], c'est-à-dire d'une même EN représentant des objets différents : l'EN *Robert Smith* désigne deux personnes distinctes (mathématicien et chanteur). Pour les exemples (3) et (4), l'ambiguïté est de type polysémie lexicale, une EN représentant un seul objet pouvant faire référence à plusieurs sens : de manière générale, *Yannick Noah* est une EN de type *personne* mais à un niveau de granularité plus fin, on peut différencier ses rôles de musicien et sportif. Notons que les UL en interaction avec des EN sont elles aussi potentiellement polysémiques : dans les exemples (4) et (5), l'UL *victoire* peut être considérée dans un contexte politique, sportif ou cinématographique (la remise des Oscars).

1.1.2.3 Vers une une désambiguïsation fine des entités nommées

Certaines EN se référant à des humains peuvent être répertoriées par rapport à des classes sémantiques (`personne`, `musicien`, `chanteur`, `sportif`, `politicien`, `acteur`...) illustrées par le diagramme de Venn de la figure 1.3. Des recouvrements entre des classes d'EN peuvent y être observés étant donné que certaines EN possèdent de multiples « facettes ». Par ailleurs, ces recouvrements sont parfois assimilables à différents niveaux de granularité et presque exprimables comme des inclusions.

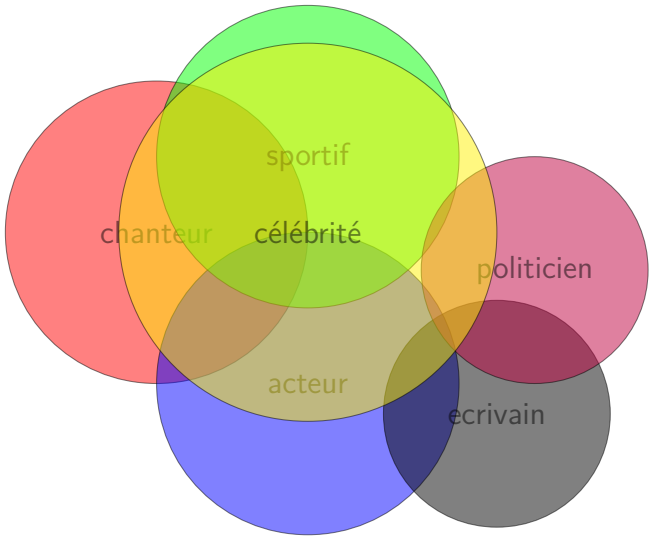


FIG. 1.3 – Catégorisation des exemples 1 à 5

On peut observer actuellement une communauté grandissante autour du problème de la désambiguïsation des référents des EN. En 2007, dans le cadre de la

campagne d'évaluation SemEval, la tâche *Web People Search* [Artiles *et al.*, 2007] proposait de désambiguïser des noms propres de personnes dans un contexte de recherche d'information sur le Web. Étant donnée une requête ambiguë (par exemple « John Smith »), l'objectif était de séparer les documents se rapportant à des personnes différentes et de regrouper les documents se référant à une même personne. Cette problématique de l'ambiguïté référentielle ne sera pas étudiée davantage ici. Nous consacrons maintenant notre étude à l'exploitation d'une modélisation de la polysémie lexicale pour désambiguïser le sens d'unités lexicales en contexte.

1.1.3 Désambiguïsation lexicale automatique

D'après Ide et Véronis [1998], un système de désambiguïsation lexicale (en anglais *word sense disambiguation*) s'attache à associer des mots d'un texte avec une unité de sens qui se différencie d'autres unités de sens potentiellement attribuables à ce mot. En général, cette tâche met en œuvre deux étapes. La première consiste à inventorier tous les sens possibles pour chaque mot du texte à désambiguïser. La seconde étape consiste à assigner un sens approprié à chaque occurrence d'un mot et s'appuie sur les occurrences des autres mots qui apparaissent dans son contexte.

1.1.3.1 Exploitation du contexte

Les systèmes automatiques de désambiguïsation lexicale tentent de résoudre l'ambiguïté d'une unité lexicale en exploitant son contexte d'usage dans un énoncé. Une des principales difficultés de cette tâche tient au fait que toutes les unités lexicales d'un énoncé sont potentiellement polysémiques lorsqu'elles sont considérées indépendamment. Le sens d'un énoncé dépend du sens des unités lexicales qui le composent. Pour le déterminer, les systèmes de désambiguïsation doivent donc parvenir à résoudre une situation d'interblocage : le sens d'un énoncé dépend du sens des unités lexicales qui le composent et les sens de ces unités dépendent eux-mêmes du sens global de l'énoncé. On peut aussi se limiter à la désambiguïsation de certaines unités lexicales. On parle alors de *targeted word sense disambiguation* par opposition aux systèmes qui traitent tous les mots d'un énoncé (*all words word sense disambiguation*). À cet égard, la *reconnaissance des EN* peut être vue comme un cas de *targeted word sense disambiguation.*

Nous présentons maintenant différentes stratégies utilisées pour désambiguïser les UL d'un corpus.

1.1.3.2 Principales stratégies de désambiguïsation

Le traitement de l'ambiguïté lexicale peut être envisagé selon plusieurs familles d'approches de désambiguïsation sémantique. Celles-ci peuvent être différenciées en fonction du type de données nécessaires pour produire un résultat.

Une première approche tente de déterminer le sens des mots en utilisant notamment la connaissance lexicale répertoriée dans des dictionnaires ou des thésaurus. De nombreuses méthodes de désambiguïsation sémantique exploitent de telles ressources : la plupart d'entre elles s'appuient sur des mesures d'affinité entre les entrées lexicales de la ressource et les unités lexicales du corpus. Nous examinons plus précisément ces ressources dans la section 1.2.1 et les méthodes pour les exploiter en désambiguïsation dans le chapitre 3.

Les approches qui exploitent des corpus examinent, elles, les occurrences d'UL et leurs contextes en utilisant des outils d'apprentissage artificiel. Lorsque la désambiguïsation consiste à étiqueter des UL par des sens prédéfinis, les techniques d'apprentissage utilisées sont supervisées. Le système est alors capable d'apprendre à classer des UL d'une séquence en fonction de critères observés dans une fenêtre de texte limitée. L'apprentissage non supervisé intervient lorsque l'on ne dispose d'aucune annotation pour désambiguïser les UL du corpus. Cette stratégie détermine le sens des UL par leurs usages possibles dans les textes, en regroupant les UL sémantiquement proches en fonction de la similarité de leurs contextes d'apparition. Dans ce cadre, nous pouvons à nouveau distinguer plusieurs types de travaux : ceux dont le but est de désambiguïser directement des unités lexicales, et ceux cherchant à construire une ressource spécifique (à un domaine, à une tâche, ou à certains types d'unités lexicales) pour l'exploiter en désambiguïsation.

1.1.4 Discussion

Une classification prédéfinie des unités lexicales demande de fixer la granularité des sens et cette dernière peut varier d'un corpus/domaine à l'autre. Quelle que soit la tâche de désambiguïsation, il n'est pas envisageable d'annoter manuellement de nouveaux corpus pour permettre un apprentissage et s'adapter à un nouveau domaine ou un nouveau jeu d'étiquetage. De plus, il ne nous semble pas réaliste de constituer manuellement une base lexicale qui atteigne une couverture satisfaisante du vocabulaire présent dans un corpus. Il n'est clairement pas envisageable d'adapter manuellement ce type de base aux nouvelles connaissances qui apparaissent au fur et à mesure dans un corpus présenté sous la forme d'un flux. Notre démarche sera donc guidée par une approche non supervisée employée à la construction d'une base lexicale exploitable en désambiguïsation : nous focaliserons en outre la modélisation de cette BL sur la prise en compte

des recouvrements entre unités de sens et sur leur structuration à granularité variable. Il est important de noter toutefois que pour une valider expérimentalement les modèles élaborés dans cette thèse, nous resterons attaché aux corpus étiquetés manuellement qui constituent une référence pour évaluer des systèmes de désambiguïsation sémantique.

Nous précisons à présent notre démarche concernant la construction de la BL visée par rapport aux approches existant dans la littérature.

1.2 Constitution de bases lexicales

Les applications du traitement automatique des langues qui se confrontent au problème de l'ambiguïté lexicale font généralement usage de bases lexicales. La diversité des représentations sémantiques des BL existantes démontre la difficulté de trouver un consensus concernant la nature du sens et sa description optimale. Ces BL répertorient les UL dans des structures allant du simple lexique ou dictionnaire à une organisation beaucoup plus complexe de thésaurus ou d'ontologie. Les UL peuvent y être décrites par rapport aux sens ou définitions qu'elles admettent, aux relations sémantiques et syntaxiques qu'elles entretiennent avec d'autres UL et aux usages qu'elles acceptent en corpus (se référer à Habert *et al.* [1997] pour plus de détails).

Trois principales méthodologies sont habituellement employées pour construire ce type de ressources. La première nécessite un travail d'expertise pour décrire et structurer des unités de sens. La deuxième utilise des outils d'apprentissage supervisé pour inférer des descriptions d'unités de sens à partir de corpus étiquetés à la main. La troisième exploite des techniques de classification non supervisée de sorte que des UL sémantiquement proches soient regroupées automatiquement en fonction de leurs usages dans des corpus non étiquetés.

Dans cette section, nous examinons en premier lieu différents modes d'organisation de BL construites manuellement en explicitant leurs modélisations du sens et de la polysémie lexicale. Pour continuer la description des approches nécessitant une expertise, nous présentons diverses stratégies d'acquisition supervisée capables de construire une BL lexicale à partir de corpus étiquetés sémantiquement. Nous décrivons ensuite plusieurs techniques non supervisées habituellement exploitées pour constituer automatiquement des BL à partir de corpus non étiquetés. Pour terminer, nous étudions les limites d'évolutivité de ces différents modèles de BL.

1.2.1 Bases lexicales constituées manuellement

On assiste à un développement de ressources sémantiques et lexicales, qu'il s'agisse de dictionnaires généralistes (Collins, Oxford English Dictionary...), de

dictionnaires terminologiques ou de bases de données lexicales. Ces ressources, lorsqu'elles ont été adaptées vers des formats numériques manipulables par des programmes informatiques, sont largement exploitées en TAL. Nous donnons ici plusieurs exemples de telles ressources, utilisées en désambiguïsation lexicale.

1.2.1.1 Exemples de bases lexicales généralistes

Pour illustrer notre réflexion sur la modélisation du sens, nous présentons à présent les modèles sous-jacents à deux des plus importantes ressources de sémantique lexicale constituées manuellement : WordNet [Miller, 1995] et FrameNet [Baker *et al.*, 1998].

WordNet est une base de données lexicale développée au laboratoire des sciences cognitives de l'université de Princeton. Cette ressource fournit des descriptions sémantiques des mots, indépendamment de leur articulation syntaxique. Les noms, verbes, adjectifs et adverbes sont groupés dans des *synsets*[5], exprimant chacun un concept différent. WordNet indique principalement les relations de synonymie, d'hyperonymie, mais aussi d'antonymie et de méronymie.

FrameNet, créée à l'université de Berkeley, repose sur la sémantique des cadres (de l'anglais *frame semantics*). Son objectif est de fournir une description du lexique en se référant à des exemples choisis dans des corpus, à travers des critères de représentativité lexicographique, pour chacun des sens que comporte une entrée lexicale. La notion de « cadre conceptuel » sur laquelle FrameNet s'appuie tient compte de l'énonciation en proposant une description syntagmatique de ses entrées lexicales (prédicats, structure argumentale) fondée sur une analyse d'usages sur corpus et donne ainsi une large place à la syntaxe. À l'instar du projet WordNet, FrameNet fournit aussi des descriptions sur le plan paradigmatique (hyperonymes, définitions, descriptions encyclopédiques).

1.2.1.2 Ressources spécialisées dans le traitement des entités nommées

Comme nous l'avons indiqué précédemment, des applications telles que la recherche d'information, l'extraction d'information ou l'aide à la traduction nécessitent de repérer précisément les entités nommées et de les catégoriser. Bien que ces unités constituent à elles seules plus de 10% du contenu de textes journalistiques [Coates-Stephens, 1992], et qu'elles apparaissent dans près de 30% des requêtes envoyées à un moteur de recherche généraliste, elles sont souvent absentes des dictionnaires.

5. Dans WordNet [Miller, 1995], un *synset* est défini comme une ensemble de synonymes interchangeables dans le contexte spécifique d'une proposition, sans en modifier le sens. Par exemple, les verbes {*call, telephone, call up, phone, ring*} forment un synset parce qu'ils peuvent servir à référer au même concept.

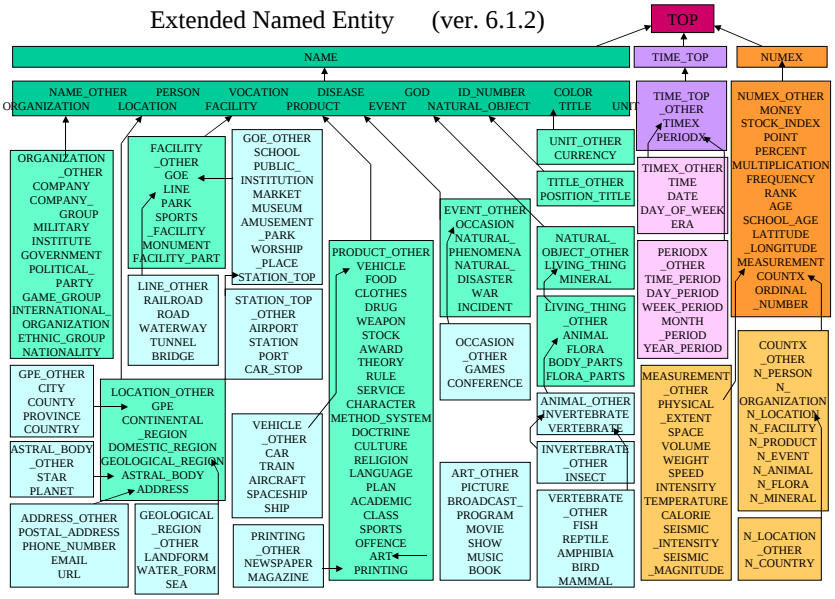


FIG. 1.4 – Un exemple de taxonomie des entités nommées proposé par Sekine [2008]

Des tentatives de classification fine d'EN ont déjà été menées. En France, une plate-forme technologique est consacrée au traitement automatique des noms propres. Il s'agit de l'ontologie Prolex [Maurel et Tran, 2006] qui correspond à un dictionnaire relationnel multilingue de noms propres complété par un système de règles [Friburger et Maurel, 2004].

On retrouve des travaux comparables pour l'anglais et le japonais dans [Sekine, 2008]. À titre d'illustration, nous fournissons un exemple de cette taxonomie en figure 1.4. Celle-ci développe le jeu d'étiquettes original des conférences MUC [Grishman et Sundheim, 1996] pour atteindre une hiérarchie de plus de 200 types d'EN.

1.2.1.3 Limites des bases lexicales expertes

Le travail considérable mis en œuvre pour des projets comme WordNet ou FrameNet a permis d'atteindre des ressources d'une envergure considérable. Les modèles de construction sous-jacents à ces BL présentent toutefois des incomplétudes autant au niveau de la structure que de la couverture lexicale. Les critiques suivantes prennent exemple sur WordNet et sont également valables pour d'autres bases lexicales constituées manuellement, notamment la taxonomie des entités nommées (*cf.* figure 1.4).

On peut noter tout d'abord que la hiérarchie des hyperonymes est répartie de manière assez déséquilibrée. Par exemple, les noms sont répertoriés en un système de catégories précis comprenant plusieurs niveaux d'héritage (atteignant parfois plus de 10 niveaux) alors que ces systèmes sont nettement moins développés et structurés pour les verbes ou les adjectifs. De plus, la transition entre deux niveaux de granularité n'est pas toujours suffisamment progressive, amenant parfois à un saut très brutal d'un concept très général à un concept très spécifique. Les co-hyponymes sont particulièrement affectés par cette hétérogénéité dans la granularité.

La réutilisabilité n'est pas triviale : les sens répertoriés dans des dictionnaires ne correspondent pas forcément à des usages rencontrés en corpus, rendant ainsi une tâche de désambiguïsation sémantique délicate [Ide et Véronis, 1998].

Il semble d'ailleurs particulièrement difficile de répertorier des UL selon une référence (jeu d'étiquettes) stable. Ainsi parmi elles, les EN se rapportent à des objets du monde en constante évolution : par exemple, les propriétés, activités et les relations qui caractérisent une personne peuvent évoluer tout au long de sa vie.

En conclusion, les dictionnaires et les ressources lexicales constitués manuellement n'atteignent généralement pas une couverture suffisante pour répertorier l'ensemble des unités lexicales d'un corpus et de leurs sens. Notre capacité à dériver les usages des mots existants et à produire de nouveaux mots rend la maintenance de ces ressources impossible. Il existe toutefois des alternatives permettant de limiter un travail d'expertise qui s'appuient sur l'exploitation de corpus par des techniques d'apprentissage artificiel pour la constitution de BL. Les stratégies d'acquisition supervisée que nous examinons maintenant font partie de ces alternatives possibles.

1.2.2 Acquisition supervisée de bases lexicales sur corpus

Les travaux fondés sur des techniques d'acquisition supervisées présentés ici répertorient des connaissances lexicales extraites de corpus soit en fonction d'unités de sens préétablies, soit en regard de relations sémantiques connues. Dans le premier cas, on dispose d'un corpus dans lequel on analyse des usages d'UL pour les étiqueter avec des unités de sens préétablies dans des BL existantes. La supervision consiste à étiqueter quelques exemples d'UL de sorte que cet étiquetage soit propagé à d'autres UL. Dans le second cas, l'objectif est d'extraire de corpus des couples d'UL qui établissent des liens dont la sémantique est connue (typiquement des liens d'hyperonymie). Nous décrivons ici successivement ces deux cas en montrant les limites des approches sous-jacentes pour traiter les questions de la modélisation la polysémie lexicale et de la granularité variable des sens.

1.2.2.1 Acquisition guidée par des unités de sens prédéfinies

Une première stratégie possible consiste à faire l'acquisition sur corpus de listes d'UL associées à des unités de sens prédéfinies dans des BL. Les techniques d'apprentissage supervisé sous-jacentes constituent des modèles capables de prédire le sens d'UL en fonction de leur contexte d'usage. Dans la perspective de réduire davantage l'expertise nécessaire pour constituer une BL, les techniques d'acquisition visent à limiter au maximum le nombre d'exemples à étiqueter. Pour atteindre cet objectif, une possibilité est de s'appuyer sur des techniques d'apprentissage semi-supervisé appliquées sur des corpus non étiquetés volumineux. Une méthode souvent envisagée repose sur une stratégie d'amorçage (*bootstrapping*) dont le fonctionnement est le suivant. Un petit nombre d'exemples étiquetés (*seeds*) est exploité lors d'un apprentissage supervisé de sorte qu'une première version d'un classifieur soit produite. Cette version sert à annoter des exemples supplémentaires qui sont exploités à leur tour pour générer une deuxième version du classifieur. Le processus est ensuite reproduit jusqu'à ce que l'intégralité des données ait été étiquetée.

Les stratégies de *bootstrapping* sont souvent employées pour l'acquisition de listes d'EN [Nadeau *et al.*, 2006]. De nombreuses variantes telles que le co-apprentissage [Blum et Mitchell, 1998] ont en outre été largement utilisées dans ce même cadre [Jones, 2005]. Les techniques d'acquisition peuvent aussi servir à construire et compléter des BL existantes : on parle alors de peuplement de BL (ou d'ontologie). Ainsi, dans le cadre d'un système de question/réponse, Mann [2002] construit automatiquement une base de noms propres à l'aide de patrons de cooccurrence en ayant pour finalité d'intégrer les données extraites à des *synsets* de WordNet. Dans la même lignée, Alfonseca et Manandhar [2002] étudient le problème d'étiquetage d'EN avec des concepts de WordNet.

Comme le souligne Pedersen [2006], ces techniques d'acquisition sont souvent qualifiées comme étant « non supervisées » alors qu'elles emploient plutôt des méthodes semi-supervisées ou faiblement supervisées (de l'anglais *weakly supervised*). Elles sont capables de répertorier des UL sur des corpus non étiquetés ; cependant, elles manipulent des unités de sens prédéfinies qui ne témoignent pas nécessairement d'usages d'UL. Elles ne sont donc pas en mesure de répondre à notre problématique d'une modélisation de la polysémie lexicale.

1.2.2.2 Acquisition guidée par des relations sémantiques prédéfinies

Il est envisageable de constituer des BL structurées en faisant usage de techniques d'acquisition supervisée de relations sémantiques entre des UL. Guidée par l'apprentissage de patrons d'extraction lexico-syntaxiques, l'acquisition consiste à identifier dans un corpus des motifs représentatifs d'une relation sémantique à partir d'un petit ensemble d'exemples de cette relation. Les patrons appris sont

ensuite réutilisés pour extraire de nouvelles unités en relation. La démarche générale d'acquisition est assimilable à processus de *bootstrapping*. Cette approche initiée par Hearst [1992] est par exemple capable d'acquérir des liens d'hyperonymie à l'aide de motifs tels que « *X est un Y* ».

Cette stratégie d'acquisition suppose que les UL extraites sont des signifiants représentant des unités de sens de la BL à construire et que les relations sémantiques qu'elles entretiennent peuvent être assimilées à des relations conceptuelles. Il est intéressant de noter que ces signifiants représentent des sens qui ne sont plus prédéfinis car ils sont extraits directement des textes. En revanche, l'acquisition est initiée par des relations dont la sémantique est prédéfinie. D'autres relations sémantiques participent sans doute à l'élaboration du sens. De plus, ces signifiants extraits automatiquement peuvent être ambigus. Cette approche ne propose donc pas de cadre pour modéliser véritablement la polysémie des unités extraites. D'autre part, une expertise est nécessaire pour établir les exemples initiaux permettant de guider l'apprentissage d'un modèle de classification visé. L'apprentissage est alors focalisé sur des exemples répondant à ce modèle et ne laisse pas émerger du corpus l'ensemble des informations pouvant participer à la construction du sens.

Face aux limitations identifiées dans la démarche d'acquisition supervisée de BL, nous focalisons à présent notre attention sur les approches distributionnelles dont l'intérêt est de constituer automatiquement des unités de sens à l'aide de méthodes non supervisées appliquées sur des corpus non étiquetés.

1.2.3 Construction automatique de bases lexicales distributionnelles

Ces dernières années, l'usage de techniques d'apprentissage artificiel a permis de combler en partie le manque de connaissances expertes par l'automatisation de la constitution de BL en s'appuyant sur la seule donnée du corpus (éventuellement étiqueté morpho-syntaxiquement). La démarche sous-jacente met en avant le fait que les UL prennent sens dans l'usage.

« For a large class of cases—though not for all—in which we employ the word "meaning", it can be defined thus : the meaning of a word is its use in the language. » [Wittgenstein, 1953].

Pour Harris [1968], le sens se définit de manière contextuelle, à travers les interactions qu'entretiennent les UL dans un énoncé. L'analyse distributionnelle automatique [Bouaud *et al.*, 1997] peut être définie comme une série d'étapes s'inspirant des hypothèses de Harris et reposant sur le regroupement d'unités linguistiques à partir de critères syntaxiques.

Dans ce cadre, Grefenstette [1994] définit deux ordres d'affinité entre des UL. Les affinités de premier ordre correspondent à des relations d'association obte-

nues par observation des cooccurrences entre UL dans un corpus. Les affinités du second ordre sont assimilables à des relations de substitution. Elles sont déterminées en identifiant des unités lexicales qui apparaissent dans des contextes similaires. La distinction entre le premier et le second ordre correspond en fait à la dichotomie saussurienne syntagmatique/paradigmatique.

Les affinités du second ordre permettent de constituer des regroupements d'UL qui ont des distributions de cooccurrents similaires. Ces regroupements peuvent être obtenus en employant des techniques d'apprentissage non supervisé sur des corpus textuels volumineux. Ces méthodes sont non supervisées dans le sens où l'acquisition de classes, ou d'affinités entre UL, procède de façon émergente à partir du corpus, sans être guidée par un inventaire de sens prédéfinis ou par un étiquetage sémantique du corpus. Les UL sont généralement séparées au sein de différents groupes (ou *clusters*) en fonction d'une contrainte de similarité. Les regroupements obtenus répertorient des UL qui ont des usages proches et dont le sens n'est pas connu *a priori*.

Nous examinons ci-dessous différentes couches de traitements sur lesquelles s'appuie la plupart des méthodes distributionnelles. La première étape consiste à extraire d'un corpus un ensemble de cooccurrences d'UL (affinités du premier ordre) représentées dans une matrice de cooccurrences. De nombreuses techniques d'apprentissage non supervisé peuvent alors être employées pour établir des listes d'usages ayant des sens proches. Ces méthodes calculent la similarité distributionnelle entre les UL identifiées (affinité du second ordre) pour constituer une matrice de similarité.

1.2.3.1 Extraction de contextes distributionnels

Les approches se référant aux travaux de Harris se différencient principalement au niveau de l'extraction de contextes distributionnels. La notion de contexte n'étant pas clairement définie en linguistique comme en TAL, de nombreuses variantes permettent d'extraire des éléments contextuels rattachés à des UL dans les énoncés d'un corpus. Dans notre cadre, les conditions qui caractérisent l'usage particulier d'une UL sont habituellement déterminées par ses cooccurrences, c'est-à-dire un ensemble d'UL qui apparaissent par exemple soit dans un espace de texte, appelé fenêtre (en anglais on fera référence aux *windowing techniques*), soit dans un contexte syntaxique donné. Dans le cas d'une extraction dans une fenêtre, les cooccurrences supposées caractériser l'essentiel du contexte « discriminant » sont situées dans un intervalle de mots autour de l'UL étudiée (par exemple n mots et m mots après). Dans le cas de l'extraction fondée sur un contexte syntaxique, on peut faire par exemple appel à des analyseurs en dépendances (voir par exemple Syntex [Bourigault, 2007] pour l'analyse du français). Grefenstette [1994] argumente en faveur d'outils syntaxiques pour l'extraction de contextes

distributionnels :

« In comparison to the more classical techniques, the use of syntactic analysis opens up a much wider range of more precise context than does simple document co-occurrence, or co-occurrence within a window of words [. . .]. Syntactic analysis allows us to seize more accurately dependencies between words, e.g., to recognize subjects of verbs, etc., and to develop this information as more precise contexts for word comparison. ».

Les approches fondées sur les fenêtres sont particulièrement utiles lorsque l'on souhaite traiter une langue pour laquelle on ne dispose pas d'analyseurs syntaxiques. En contrepartie, il n'est pas possible d'extraire les cooccurrences situées en dehors de la fenêtre d'étude et qui seraient déterminantes dans la caractérisation du sens.

Le cas des EN est un peu particulier : ces UL sont très souvent composées de plusieurs mots et leur repérage préalable repose lui même assez souvent sur un étiquetage morphosyntaxique. L'extraction des cooccurrents est souvent confiée à des patrons syntaxiques ou un analyseur en dépendances.

1.2.3.2 Représentation du contexte

La distribution d'une UL, c'est-à-dire la totalité de ses contextes d'occurrence, peut être caractérisée à différents degrés de précision, selon différentes contraintes.

Les approches distributionnelles s'appuient généralement sur des espaces de grande dimension définis par des cooccurrences d'UL. Étant données n UL identifiées dans le corpus, en supposant que l'ordre des UL n'est pas pris en compte, une matrice symétrique $M \in \mathcal{M}_n(\mathbb{R})$ peut être construite en représentant chaque UL par une vecteur-ligne et un vecteur-colonne dans M. Chaque élément $a_{i,j}$ de la matrice contient le comptage de cooccurrences entre l'UL de la colonne i et l'UL de la colonne j dans un contexte donné. Si l'on considère que l'ordre des mots dans le texte a une importance, la matrice $M \in \mathcal{M}_{m,n}(\mathbb{R})$ est alors asymétrique et rectangulaire et contient des comptes de bigrammes (*bleu ciel* et *ciel bleu* n'ont pas le même compte). La matrice (symétrique ou asymétrique) est assimilable à la base d'un espace vectoriel multidimensionnel dans lequel chaque UL (appelée objet, individu ou transaction selon le domaine) est associé à un vecteur-ligne défini par les mots du contexte (attributs, variables, ou items) dans lequel il apparaît. Une matrice symétrique peut être est associée à un graphe non orienté avec des arrêtes pondérées par le nombre de contextes partagés entre UL. Ce graphe est orienté dans le cas d'une matrice asymétrique.

1.2.3.3 Apprentissage non supervisé pour la structuration d'unités lexicales en classes sémantiquement homogènes

Nous examinons à présent différentes méthodes non supervisées couramment utilisées en analyse distributionnelle. Elles permettent de regrouper des unités lexicales apparaissant dans des contextes similaires dans des *clusters* assimilés à des unités de sens. Notons qu'aucune méthode d'apprentissage automatique n'est idéale : certaines sont justes plus adaptées que d'autres à certains cas.

Techniques de *clustering* Ces algorithmes génèrent et trouvent des regroupements (ou *clusters*) d'éléments qui se ressemblent et qui se distinguent d'éléments situés dans d'autres *clusters*. Bien qu'il existe de nombreuse variantes, la construction de *clusters* est habituellement dirigée par une forte similarité intra-classe, et une faible similarité inter-classes. En pratique, cette notion de similarité se traduit par une distance, un coefficient de dissimilarité ou une mesure de densité qui s'applique entre deux vecteurs-ligne (représentant le contexte de deux UL) de la matrice de cooccurrences. Les techniques de *clustering* emploient généralement une matrice des distances où chaque case d_{ij} contient la distance entre les objets i et j de la matrice de cooccurrences.

Il existe deux types de *clusters* : les *clusters* « durs » sont totalement disjoints les uns des autres (un élément d'un *cluster* n'appartient pas à un autre) alors que les *clusters* « mous » comportent des éléments pouvant appartenir à d'autres *clusters*. Les méthodes de *hard clustering* qui constituent des *clusters* « durs » parviennent à capter le sens principal associé à chaque mot. En contrepartie, elles sont moins adaptées pour modéliser la polysémie que les méthodes de *soft clustering*, ces dernières étant capables d'assigner une unité lexicale à plusieurs *clusters*.

Le résultat d'un *clustering* peut être structuré ou non. Dans le cas d'une classification hiérarchique [Jain et Dubes, 1988], l'ensemble des *clusters* est structuré par un arbre (souvent nommé *dendrogramme*) qui permet d'explorer les données sur plusieurs niveaux de granularité.

Les techniques de *clustering* doivent souvent être paramétrées pour déterminer (directement ou indirectement) le nombre de partitions dans lesquelles répartir l'ensemble des données d'apprentissage. Ce paramétrage est problématique pour la constitution de BL où le nombre d'unités de sens n'est pas évident à déterminer *a priori*.

Techniques de réduction de dimensionnalité L'objectif de ces techniques (voir par exemple l'analyse sémantique latente probabiliste ou PLSA [Hofmann, 1999]) est de créer un petit nombre d'éléments qui décrivent les objets et/ou les attributs d'une matrice (c'est-à-dire, les lignes et/ou les colonnes) presque aussi bien que

le font les variables « brutes » (les vecteurs colonnes et/ou lignes de la matrice), habituellement en grand nombre. Le processus réduisant la dimensionnalité d'une matrice agit comme un compresseur de l'espace de représentation des données et peut faire perdre de l'information. Ces techniques ont plusieurs avantages. Elles réduisent la quantité d'information que les algorithmes auront à traiter, diminuant ainsi, parfois fortement, les temps de calcul et l'encombrement mémoire. Chaque nouvel attribut (ou objet) compressé est assimilable à un *cluster* « mou » auquel chaque objet (ou attribut) appartient plus ou moins. En réduisant le nombre de variables à 2 ou 3, ce degré d'appartenance fournit des coordonnées pour une représentation visuelle plane ou tridimensionnelle. Par ailleurs, cette compression réalise une sorte de lissage des données (*smoothing*) qui permet de réduire les disparités des données d'apprentissage fournissant ainsi une généralisation vers un meilleur modèle de proximité distributionnelle (par l'intermédiaire d'une mesure de similarité).

Techniques fondées sur les graphes Les données textuelles précédemment représentées par une matrice peuvent être manipulées sous la forme d'un graphe dans lequel les nœuds sont associés à des UL et les arrêtes sont établies en fonction des contextes que ces UL partagent. En outre, les arrêtes peuvent être éventuellement pondérées par le nombre de contextes partagés entre UL. Plusieurs formes de regroupements d'unités lexicales peuvent être extraites de ce graphe. Bouaud *et al.* [1997] procèdent par exemple à l'extraction de cliques et de composantes connexes de ce graphe qui participent à la construction d'une ressource lexico-sémantique. Une méthodologie similaire a été proposée par Ehrmann et Jacquet [2006] pour construire une ressource répertoriant des EN d'un corpus.

Inférence de regroupements contextualisés Nous avons présenté différentes approches non supervisées qui fournissent des rapprochements entre des UL similaires. Ces méthodes sont unidimensionnelles car elles n'indiquent pas directement le contexte utilisé pour établir les *clusters* : une analyse des variables associées à ces *clusters* est toutefois possible. Nous avons identifié deux autres approches non supervisées qui ne sont pas habituellement exploitées ni pour la constitution de BL, ni en désambiguïsation lexicale. Les techniques de *co-clustering* [Hartigan, 1972] et d'extraction de règles d'association [Agrawal et Srikant, 1994] permettent d'identifier non seulement des groupes d'objets présents dans les données, mais également les sous-ensembles d'attributs spécifiques sur lesquels ces groupes sont définis.

Un *co-clustering* est défini comme le regroupement simultané des lignes et des colonnes d'une matrice. Le mot *co-clustering* comporte une certaine idée de simultanéité dans la phase de construction des deux partitions.

L'extraction de règles d'association est une technique non supervisée d'apprentissage automatique pour la découverte de relations fines entre des attributs. D'une certaine manière, cette technique effectue un regroupement préalable de l'ensemble des attributs (colonnes de la matrice) pour ensuite déduire le regroupement sur les objets. Il existe une correspondance directe entre les deux regroupements qui fournit une sorte de *co-clustering*. Par ailleurs, ces règles d'association ont l'avantage de fournir une interprétation riche des résultats et facilitent la manipulation et l'interprétation pour les humains et les programmes informatiques.

L'extraction de règles d'association a l'avantage de produire des résultats facilement interprétables. En revanche, un filtrage de ces résultats est nécessaire pour réduire le volume parfois considérable de règles produites. Par ailleurs, les regroupements inférés par *co-clustering* sont « expliqués » par les attributs qui ont permis de constituer ces regroupements. Les regroupements d'UL correspondent ici à des contextes d'usages qui définissent d'autres regroupements d'UL. L'inférence de règles ou de regroupements « contextualisés » pourrait participer à la constitution d'une BL exploitable en désambiguïsation sémantique et interprétable par des humains ou des programmes informatiques.

Il est important de noter qu'en principe, aucune méthode non supervisée n'est idéale. Retenons toutefois que la construction d'une BL facilement interprétable et réutilisable en désambiguïsation sémantique semble facilitée par l'inférence de regroupements ou de règles « contextualisés ». Nous devons aussi considérer les méthodes de *soft clustering* qui semblent mieux adaptées que les techniques de *hard clustering* pour tenir compte des recouvrements entre unités de sens, ce qui est un aspect important pour modéliser la polysémie lexicale. Nous ne pouvons pas encore nous fixer sur une approche particulière mais nous devons cependant remarquer que pour aboutir la construction d'une base lexicale évolutive, notre démarche doit aussi être dirigée par certaines contraintes imposées par un traitement séquentiel de flux. Nous allons à présent exposer ces contraintes.

1.2.4 Évolutivité des ressources lexicales

Dans des environnements dynamiques, les BL créées manuellement souffrent de leur manque d'évolutivité. Les approches distributionnelles existantes montrent aussi certaines limites pour intégrer de nouvelles connaissances issues de flux. Les mécanismes d'apprentissage sous-jacents aux approches distributionnelles ne sont en effet pas prévus pour intégrer de nouvelles connaissances à une structure déjà figée lors d'un apprentissage préalable. Un nouvel apprentissage complet est à envisager lorsque l'on veut prendre en compte de nouvelles connaissances : l'ajout de nouvelles dimensions associées à de nouveaux éléments nécessite la plupart du temps de refaire tous les calculs sur une matrice complétée, ce qui n'est pas envisageable pour certaines applications.

Les structures résultant d'une analyse distributionnelle sont donc le plus souvent des classes qui ne sont pas prévues pour évoluer. Des classes déjà figées demandent des restructurations pour s'adapter à de nouvelles observations. Comme l'indique la figure 1.5, l'introduction progressive d'une nouvelle connaissance est susceptible de perturber les subdivisions de sens existantes en les séparant en des sens plus spécifiques ou en les fusionnant dans des sens plus généraux. Il est notable que de nouvelles connaissances peuvent aussi parfois générer des sens totalement nouveaux.

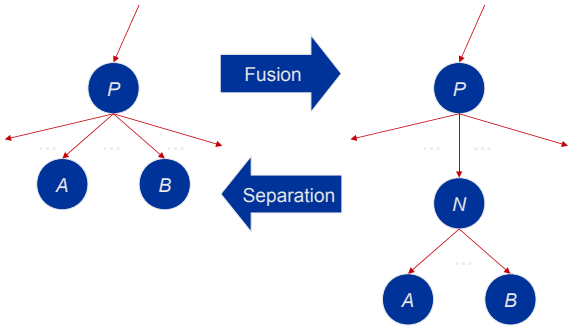


FIG. 1.5 – Fusion et séparation d'unités de sens dans une hiérarchie

Pour être plus évolutives, il serait intéressant que les structures qui modélisent la polysémie lexicale dans une BL soient conçues pour prendre en compte différents niveaux de granularité sémantique. Les mots peuvent être décrits selon des sens plus ou moins généraux à choisir en fonction du contexte. À un niveau de granularité grossier, les sens sont bien découpés et figés ; en revanche, à un niveau de description plus fin, ces sens généraux tendent à se différencier et à recouvrir des sens plus précis. Une base lexicale munie de différents niveaux de granularité pourrait nous permettre de fusionner ou séparer localement des sens précis sous l'influence de nouvelles connaissances sans pour autant perturber des sens plus généraux et établis.

Au regard de ces remarques, la prise en compte de données fournies séquentiellement serait facilitée par une base lexicale structurée sur plusieurs niveaux de granularité sémantique mais aussi par un processus capable d'apprendre des connaissances en continu. L'intégration des données en provenance de flux est un problème pris en compte par les méthodes incrémentales d'apprentissage artificiel. À notre connaissance, ces dernières n'ont toutefois pas été employées pour constituer des bases lexicales distributionnelles. Nous examinons à présent différentes techniques incrémentales susceptibles de s'accorder avec un modèle sémantique de base lexicale structurée sur plusieurs niveaux de granularité.

1.3 Approches incrémentales pour la construction de bases lexicales évolutives

Les outils de traitement automatique des langues sont souvent amenés à rencontrer de nouveaux usages (tournures, vocabulaire, sens nouveaux) dans les corpus. Les bases lexicales utilisées par ces outils ne sont alors généralement pas suffisamment exhaustives. Leur modification est un processus lourd nécessitant, en règle générale, la collaboration d'experts humains. Il serait pourtant souhaitable de pouvoir y intégrer automatiquement de nouvelles connaissances.

Nous considérons désormais que les données manipulées sont fournies sous une forme séquentielle, c'est-à-dire qu'elles apparaissent au fur et à mesure et qu'il est possible de les stocker. Il s'agit d'éviter de procéder à un apprentissage à partir de zéro pour chaque nouvel élément pouvant mettre en cause la structure d'une BL existante. En adoptant une stratégie incrémentale, on impose au système d'apprendre en continu, sans avoir reçu l'ensemble des données. Ce type de processus dynamique et historique est intéressant pour qu'une ressource existante puisse s'adapter et évoluer afin d'intégrer et répertorier de nouvelles connaissances.

Nous développons plusieurs axes dans cette section. Nous exposons d'abord nos motivations pour traiter des flux de données textuelles en fournissant des exemples d'applications. Nous examinons ensuite les travaux menés sur l'intégration de connaissances dans les ontologies afin de leur apporter un caractère évolutif. Pour terminer, nous examinons plusieurs techniques d'apprentissage incrémental non supervisé capables d'intégrer progressivement des données présentées en séquence.

1.3.1 Construction à partir de flux de données textuelles

Depuis quelques années, la thématique du traitement de flux de données est un sujet d'intérêt grandissant pour différentes communautés (bases de données, analyse de données...) et ceci aussi bien du côté industriel qu'académique. On observe actuellement l'émergence d'applications du TAL, notamment en recherche d'information, devant traiter des jeux de données textuelles volumineux présentés sous forme de flux dans lesquels de nouvelles unités lexicales apparaissent et les sens d'usages connus dérivent. Les bases lexicales existantes ne sont pas conçues pour adapter leur structure aux évolutions du vocabulaire.

Pour introduire cette problématique, nous commençons par éclaircir la notion de flux de données textuelles. Nous examinons ensuite une application de désambiguïsation lexicale dans des flux de données textuelles.

1.3.1.1 Notion de flux de données textuelles

Un flux de données (*data stream*) est une séquence ordonnée d'éléments $e_0, e_1, e_2 \ldots$ qui doivent être traités de manière séquentielle, élément par élément. De nombreux auteurs [Cormode, 2007; Muthukrishnan, 2005] considèrent chaque élément e_i d'un flux F comme un vecteur de dimension d ; $e_i = (e_{1i}, e_{2i} \ldots e_{di})$. Chaque lecture d'un flux est appelée une passe (ou *scan* linéaire) : seules quelques passes (une seule en général) sont autorisées pour accéder aux éléments d'un flux.

La communication linguistique est caractérisée par une transmission de signes en séquence et peut amener à examiner différents grains d'analyse textuelle [6]. Chaque élément d'une séquence peut par exemple correspondre à un document (dépêche d'actualité, courrier électronique), un paragraphe, un tour de parole (dans un dialogue ou dans le cadre d'un système de messagerie instantanée), un énoncé, une unité lexicale... En réalité, ces séquences d'unités textuelles sont imbriquées les unes dans les autres : un document contient une séquence de paragraphes possédant à leur tour une séquence d'énoncés... Un flux de données textuelles possède donc une multitude d'échelles d'observation.

Nous n'allons pas développer ici d'éventuelle définition générique de flux textuel « multi-échelles » qui risquerait d'être vague. Il nous semble nécessaire de restreindre les niveaux textuels à examiner en étant guidé par une tâche applicative précise. Nous décrivons un exemple d'application traitant un flux de dépêches d'actualité afin de détailler un cadre expérimental potentiel mis en œuvre pendant la constitution d'une base lexicale évolutive.

1.3.1.2 Applications et traitements sur des flux de données textuelles

De nombreux outils de recherche, accessibles au public, sont désormais amenés à traiter des documents textuels présentés sous forme de flux [7]. Pour une application de type *Google News* [8], plusieurs flux de données sont traités en même temps. Les éléments des flux sont des dépêches d'actualité récoltées régulièrement à partir des sites Web d'organes de presse ou de flux RSS. Les articles sont accessibles à partir de rubriques prédéfinies (économie, sport, santé, sciences...) ou par l'interrogation classique par mots-clés. Les articles qui traitent du même sujet sont regroupés grâce à une analyse de leur contenu textuel. Ces regroupements sont actualisés en temps réel par un enrichissement de nouveaux articles. Depuis peu,

6. Cette notion de grain d'analyse textuelle correspond au plan syntagmatique. Elle ne doit pas être confondue avec la notion de granularité sémantique qui concerne le plan paradigmatique.

7. Voir par exemple les outils développés par *Google* accessibles en ligne : *news search*, *audio indexing*, *trends*.

8. `http://www.google.com/news` .

l'outil *Google Trends*[9] est capable de montrer les fluctuations temporelles d'un mot-clé apparaissant dans les documents de *Google News*. La figure 1.6 est obtenue à partir de l'application en ligne *Google Trends* pour la requête « *Ivory Coast* » au cours de la période 2004-2008. Elle comporte trois zones. Le chronogramme situé en haut (*search volume index*) indique les évolutions du nombre de demandes envoyées par des utilisateurs pour la requête « *Ivory Coast* », normalisé par le nombre de recherches total. Les étiquettes situées sur les pics de la courbe sont à mettre en correspondance avec les titres de dépêches dans la zone située à droite. Les titres des documents traitent des événements relatifs à la guerre civile de Côte d'Ivoire et aux résultats de son équipe de football. Le chronogramme situé en bas (*new reference volume*) indique le nombre de documents contenant l'expression « *Ivory Coast* » parmi les documents de *Google News*.

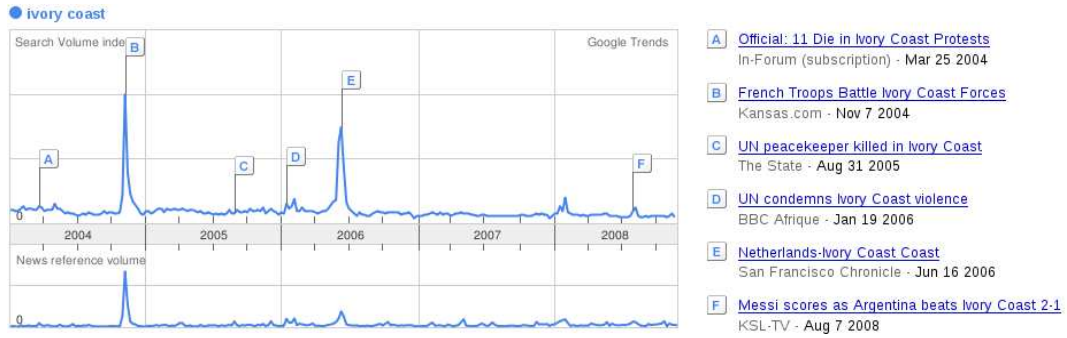


FIG. 1.6 – Google Trends : évolution du nombre d'occurrences des documents associés à la requête « *Ivory Coast* » au cours de la période 2004-2008

Comme nous l'avons dit, les EN représentent plus de 10% du contenu de textes journalistiques [Coates-Stephens, 1992]. Cette tendance reste probablement observable dans le contenu et les titres dépêches d'actualité qui apparaissent en flux sur Internet. Dans la perspective d'une indexation fine de ce type de flux, il est potentiellement intéressant d'examiner l'évolution de la classification des EN et de procéder à leur désambiguïsation au cours du temps. Par ailleurs, la désambiguïsation d'EN et la classification de documents journalistiques peuvent bénéficier de la prise en compte d'une composante temporelle. Une EN apparaissant au même moment dans des documents de flux différents a de bonnes chances de se référer au même objet et d'avoir la même classification [Shinyama et Sekine, 2004]. D'autre part, des documents apparaissant au même moment dans des flux différents et qui possèdent les mêmes EN ont de bonnes chances de traiter du même sujet [Vergne, 2003].

Les EN apparaissant dans des flux d'actualité font intervenir de nouvelles connaissances que nous souhaitons répertorier au sein d'une BL capable d'évo-

9. `http://www.google.com/trends` .

luer. Dans cette perspective, nous examinons à présent les travaux menés sur l'intégration de connaissances dans les ontologies.

1.3.2 Fusion de connaissances dans les ontologies évolutives

Pour évoluer, une base lexicale déjà établie (manuellement ou automatiquement) doit permettre une réorganisation de ses subdivisions de sens. De nouvelles relations entre les concepts de cette base lexicale sont susceptibles de perturber sa structure.

Le domaine de la *révision de croyances* (de anglais *belief revision*) a développé des formalismes et des procédures afin de faire évoluer des bases de connaissances. Il s'agit d'adapter une description (ou théorie) du domaine en fonction de quelques nouveaux exemples et contre-exemples. Ces processus d'adaptation font appel à des opérateurs (contraction, expansion, révision, consolidation, fusion) mettant en œuvre des raisonnements logiques.

Dans la continuité de ces travaux, la communauté du Web sémantique a explicité différents modes d'évolution d'ontologies. D'après Haase *et al.* [2005], une ontologie évolue pour s'adapter à des changements en les propageant aux objets qu'elle référence ainsi qu'aux applications qui lui sont liées. Ces changements sont classés selon trois niveaux. Le premier concerne le domaine de l'ontologie qui évolue lorsque les éléments du monde réel qu'elle modélise sont dotés d'une certaine dynamique. Le deuxième est lié aux changements de conceptualisation qui se manifestent par l'enrichissement des relations entre les concepts du domaine engendré par une nouvelle observation ou une restructuration des connaissances. Le troisième niveau se rapporte au changement de spécification de la description formelle de l'ontologie, c'est-à-dire à son langage de représentation. Stojanovic [2004] ajoute que l'utilisation d'une ontologie dans le cadre d'une application peut nécessiter des adaptations.

Issues d'une tradition symbolique classique en intelligence artificielle, les approches fondées sur la révision de croyance ont tendance à construire une axiomatique de départ comme fondement qui s'inscrit dans le cadre d'une sémantique formelle vériconditionnelle, dans laquelle le sens d'un mot est une « vérité ». Certains aspects du sens relèvent toutefois plus des conditions d'usage que de conditions de vérité. Dans cette optique, Cimiano et Völker [2005] ont élaboré une méthode pour construire et faire évoluer automatiquement des ontologies terminologiques à partir de textes. Les relations de l'ontologie sont des prédicats probabilisés extraits de corpus à l'aide de patrons prédéfinis tels que « x *IS A* y » ou « x *SUCH AS* y » [Cruse, 1986; Hearst, 1992] pour lesquels les unités lexicales x et y extraites sont des signifiants représentant des concepts de l'ontologie. Le peuplement de cette ontologie est obtenu en associant ces unités lexicales extraites à une sélection d'autres unités lexicales qui leur sont similaires (au sens distribu-

tionnel). La mise à jour de l'ontologie se focalise uniquement sur les relations et les pondérations affectées par une modification. Ces mécanismes d'acquisition et de mise à jour facilitent la compréhension des évolutions qui se manifestent dans un corpus. Notons toutefois que chaque concept de l'ontologie fait référence à un unique « terme » sélectionné pour être peu ambigu.

Dans le cadre des ontologies terminologiques évolutives, il n'est pas question de modéliser la polysémie, de regrouper des unités lexicales similaires au sein de *clusters*, ou encore d'une exploitation en désambiguïsation sémantique ; des efforts concernant les questions d'évolution du sens sont encore à mener. Ce cadre ne correspond donc pas à notre problème de construction d'une BL évolutive. Dans cette perspective, nous examinons maintenant les travaux existants en *clustering* incrémental qui pourraient offrir aux approches distributionnelles un aspect évolutif.

1.3.3 Partitionnement incrémental de données fournies en séquence

L'aspect temporel est fondamental en linguistique comme en TAL puisque l'information linguistique est transmise sous forme de séquences. Les travaux actuels en analyse distributionnelle automatique ne considèrent pas encore que les usages d'une UL puissent évoluer au cours du temps alors que cet aspect temporel nous semble lié aux questions d'émergence d'unités de sens. Dans cette perspective, nous proposons d'examiner des techniques capables de tenir compte d'évolutions pour construire une représentation.

Le cadre de l'apprentissage incrémental permet d'effectuer des traitements de données fournies en séquence selon un processus dynamique et historique. Un système incrémental doit être capable d'apprendre en continu, sans avoir reçu l'ensemble des données. Plusieurs types d'apprentissage incrémental existent. Nous nous concentrons ici principalement sur les travaux en *clustering* pour répondre à un objectif d'ajout de la composante temporelle à des systèmes d'analyse distributionnelle automatique. Nous distinguons le *clustering* incrémental « hors ligne » du *clustering* incrémental « en ligne ». Dans un cadre hors ligne, il y a une possibilité de stocker les données ; le modèle non supervisé obtenu peut être le même que dans un mode non incrémental. La représentation apprise peut être enrichie à mesure que les exemples d'apprentissage sont traités. Pour un apprentissage en ligne, toutes les données ne sont pas stockées. Des mécanismes d'oubli doivent être considérés dans la mise à jour de la représentation des exemples. Dans la suite de ce manuscrit, ces derniers mécanismes ne sont pas détaillés et notre étude se focalisera sur l'apprentissage incrémental hors ligne.

Nous examinons ci-dessous plusieurs méthodes incrémentales d'apprentissage non supervisé potentiellement intéressantes pour constituer des bases lexicales

distributionnelles évolutives.

1.3.3.1 Analyse de concepts formels

L'analyse de concepts formels (ACF) [Ganter et Wille, 1999] est un cadre mathématique pour la classification non supervisée symbolique. L'ACF permet d'inférer des concepts formels pour mettre en relation des regroupements d'objets et des regroupements d'attributs. Les concepts formels constituent donc des regroupements contextualisés. À cet égard, l'ACF est parfois considérée comme une méthode de *co-clustering*. Les concepts sont organisés au sein d'une structure hiérarchique de treillis de Galois (ou treillis de concepts, en anglais, *concept lattice*). Le treillis de Galois [Birkhof, 1967; Barbut et Monjardet, 1970] d'une relation entre deux ensembles est à la base de plusieurs méthodes de classification conceptuelle.

Nous examinons maintenant le cadre général de l'ACF en donnant sa terminologie, ses définitions et ses notations standards.

Définition 1.1 (contexte formel) *Un triplet* $\mathbb{K} = (O, A, R)$ *est appelé* ***contexte formel*** *si* O *et* A *sont des ensembles disjoints et* $R \subseteq O \times A$ *est une relation binaire entre* O *et* A*. Les éléments de* O *sont appelés les* ***objets*** *et ceux de* A ***attributs****.*

À titre d'exemple, nous avons constitué un contexte formel « jouet » (*cf.* table 1.1) qui associe des entités nommées (objets représentés sur les lignes) à des « mots clés » (attributs représentés sur les colonnes).

	album de	concert de	acteur	film avec	fan de	interview de	victoire de	élection de	politique de	discours de
Arnold Schwarzenegger			×	×	×	×	×	×	×	×
Angela Merkel							×	×	×	×
Yannick Noah	×	×			×	×	×			
Michael Jackson	×	×			×	×				

TABLE 1.1 – Exemple de contexte formel binaire

Définition 1.2 (opérateur de dérivation, connexion de Galois) *Soit* $\mathbb{K} = (O, A, R)$*, un contexte formel. Pour tout* $E \subseteq O$ *et* $I \subseteq A$*, on définit deux ensembles* $E' \subseteq A$ *et* $I' \subseteq O$ *:*

$$E' = \{a \in A | \forall o \in E : (o, a) \in R\}$$
$$I' = \{o \in O | \forall a \in I : (o, a) \in R\}$$

*E' est l'ensemble de tous les attributs de A qui sont en relation avec tous les objets de E, et I' est l'ensemble des objets de O en relation avec tous les attributs de I. L'**opérateur de dérivation**, noté « ' » s'applique aussi bien aux sous-ensembles de O qu'aux sous-ensembles de A.*

Par exemple, si $I = \{$discours de, élection de$\}$, alors $I' = \{$Angela Merkel, Arnold Schwarzenegger$\}$. Pour $E = \{$Michael Jackson$\}$, nous avons $E' = \{$album de, concert de, interview de, fan de$\}$.

L'opérateur « ′ » peut se composer avec lui même, pour partir d'un sous ensemble d'objet E, en lui faisant correspondre les attributs de l'ensemble E', et associer à E' les objets de l'ensemble E'' (la composition est ici marquée par la notation « ″ »).

Les opérateurs « ′ » et « ″ » vérifient les propriétés suivantes pour $E, E_1, E_2 \subseteq O$ et $I, I_1, I_2 \subseteq A$:

- $E_1 \subseteq E_2 \Rightarrow E_2' \subseteq E_1', I_1 \subseteq I_2 \Rightarrow I_2' \subseteq I_1'$
- $E \subseteq E''$ et $E' = E'''$, $I \subseteq I''$ et $I' = I'''$ (la composition de « ′ » et « ″ » est notée « ‴ »)
- $E \subseteq I' \Leftrightarrow I \subseteq E'$

Définition 1.3 (concept formel, intension, extension) *Un **concept formel** d'un contexte formel $\mathbb{K}$ est une paire $c = (E, I)$ qui satisfait $E \subseteq O, I \subseteq A$, $E' = I$ et $I' = E$. E et I sont respectivement appelés l'**intension** (notée int(c)) et l'**extension** (notée ext(c)) du concept c.*

Par exemple, la paire ({Angela Merkel, Arnold Schwarzenegger},{victoire de, élection de, discours de, politique de}) est un concept formel.

Définition 1.4 (ensemble des concepts formels, relation de subsomption) *Soit $\mathfrak{T}(O, A, R)$, l'**ensemble des concepts formels** d'un contexte formel $\mathbb{K} = (O, A, R)$. L'ensemble $\mathfrak{T}(\mathbb{K})$ est muni d'une relation d'ordre partiel $\leq$ appelée **relation de subsomption**. Elle est définie par la relation d'inclusion $\subseteq$ entre les extensions des concepts $C_1 = (E_1, I_1)$ et $C_2 = (E_2, I_2)$ de l'ensemble $\mathfrak{T}(\mathbb{K})$. De manière équivalente, cette relation de subsomption est définie par la relation d'inclusion inverse $\supseteq$ entre les intensions des concepts C_2 et C_1. Plus formellement, on peut écrire :*

$$C_2 \leq C_1 \Leftrightarrow ext(C_2) \subseteq ext(C_1) \Leftrightarrow int(C_2) \supseteq int(C_1) \tag{1.1}$$

Par exemple, nous avons $C_2 \leq C_0$ pour les concepts
C_2 =({Arnold Schwarzenegger, Yannick Noah},{victoire de, interview de, fan de}) et
C_0 =({Michael Jackson, Yannick Noah, Arnold Schwarzenegger},{interview de, fan de}).

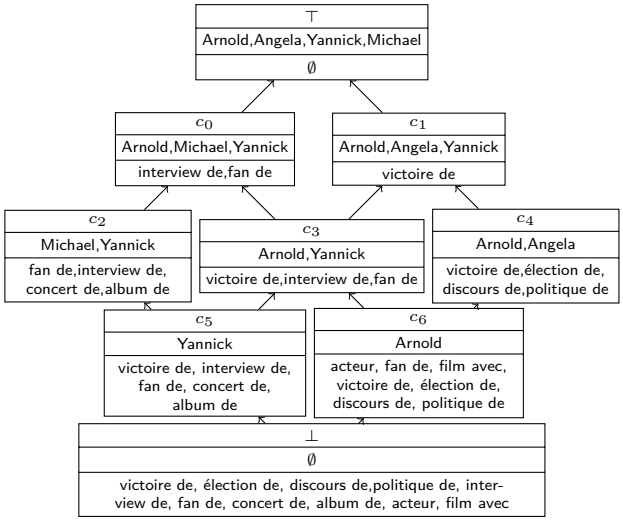


FIG. 1.7 – Treillis de Galois associé à un contexte formel (*cf.* figure 1.1)

Définition 1.5 (treillis de Galois) *La relation de subsomption* $\leq$ *permet de constituer un treillis complet* $(\mathfrak{T}(O, A, R), \subseteq)$*, appelé* ***treillis de Galois*** *ou treillis de concepts qui est noté* $\underline{\mathfrak{T}}(O, A, R)$*. Le treillis de Galois admet un* ***élément minimal*** *noté* $\perp$ *et un* ***élément maximal*** *noté* $\top$*.*

Le diagramme de Haase (*cf.* figure 1.7) représente le treillis de Galois associé au contexte formel (*cf.* table 1.1). La borne supérieure $\top$ du treillis regroupe à l'ensemble des objets O dans son extension et la borne inférieure $\perp$ regroupe à l'ensemble des attributs A dans son intension. Pour des questions de lisibilité, les objets du contexte formel (*cf.* table 1.1) associés à des prénoms/noms sont représentés uniquement par des prénoms sur la figure 1.7 (par exemple, « Yannick Noah » est remplacé par « Yannick »).

Les treillis fournissent une organisation fortement structurée qui résume les relations de généralité entre les attributs vis à vis des objets et, symétriquement, celle des objets vis à vis des attributs. La flexibilité de la structure de treillis permet une construction selon une procédure incrémentale (voir par exemple l'algorithme *add_intent* proposé par van der Merwe *et al.* [2004]). Le cadre de l'ACF se montre ainsi particulièrement intéressant dans un domaine en évolution car un treillis peut être construit dynamiquement avec de nouveaux objets et attributs. En considérant le treillis comme une base de connaissance déjà structurée, son adaptation à de nouveaux corpus permettrait de combler le manque d'évolutivité de nombreuses ressources lexicales. Il faut noter toutefois que les algorithmes existants de construction incrémentale ne sont pas réellement capables d'intégrer des connaissances issues de flux à un treillis déjà construit. Des extensions à ces

algorithmes sont donc nécessaires pour atteindre cet objectif de construction à partir de flux.

1.3.3.2 *Clustering* conceptuel incrémental

Le *clustering* conceptuel est un paradigme d'apprentissage automatique développé en classification non supervisée dans les années 1980 [Fisher, 1987]. Il se distingue des méthodes de *clustering* statistiques en produisant une description interprétable pour chaque classe générée. La plupart des méthodes symboliques de *clustering* sont capables de créer des structures hiérarchisées de catégories.

CobWeb [Fisher, 1987] est l'un des algorithmes de *clustering* conceptuel les plus connus. Sa procédure incrémentale lui permet d'intégrer progressivement les instances d'un jeu de données dans un arbre de classification. Chaque nœud de l'arbre est un concept probabilisé qui représente une classe d'objets. Au démarrage, l'arbre de classification possède un seul nœud racine. Les instances sont ensuite ajoutées progressivement à l'arbre qui est mis à jour à chaque étape. L'intégration d'un nouvel élément est un processus de classification qui réalise un parcours de l'arbre en descendant le long d'un chemin approprié, en actualisant les comptages (pour les probabilités) le long du chemin et en faisant des opérations de fusion, de séparation, de création de concepts ou de classification d'un objet en fonction des classes existantes. L'opération choisie est celle qui maximise une fonction mesurant l'utilité d'une catégorie [10].

1.3.3.3 *Clustering* incrémental exploitant la densité des données

Les algorithmes incrémentaux de *clustering* les plus utilisés manipulent des mesures de densité pour regrouper des instances voisines (*density based clustering*). Les instances à classer sont représentées dans un espace métrique dans lequel on s'intéresse aux zones denses en exemples. Pour les méthodes telles que DBSCAN [11] [Ester *et al.*, 1996], la densité est définie en comptant le nombre d'exemples dans le voisinage d'un exemple. Cette notion de voisinage permet d'intégrer de nouvelles instances à des *clusters* de manière incrémentale, et c'est ce fonctionnement que l'algorithme *IncrementalDBSCAN* [Ester *et al.*, 1998] implémente. La méthode exploite la densité des données pour ne perturber le *clustering* que dans un petit voisinage d'une instance lors de son insertion ou de sa suppression. Le *clustering* produit à chaque étape est identique à celui réalisé en une seule étape (mode non incrémental).

10. Cette mesure (*category utility*) calculée à l'aide de l'entropie se rapproche de l'information mutuelle.

11. *Density Based Spatial Clustering of Application with Noise.*

1.3.3.4 Discussion

En fournissant des algorithmes pour construire des regroupements étape par étape, les méthodes de *clustering* incrémental présentées nous ont permis d'amorcer une réflexion sur la constitution automatique de BL évolutives. Aucune des approches incrémentales présentées dans cette section ne possède tous les avantages d'un algorithme de *clustering* idéal. Nous allons ici nous intéresser à l'expressivité de la représentation des exemples et des *clusters* (langages des exemples et des hypothèses).

Les méthodes reposant sur la densité entrent dans le cadre des approches distributionnelles classiques utilisant des métriques pour identifier des éléments proches sémantiquement. Ne fournissant pas de description hiérarchisée des *clusters*, elles ne permettent pas de modéliser la granularité sémantique. Par ailleurs, le recouvrement entre *clusters* n'est pas supporté.

CobWeb fournit une organisation hiérarchique des *clusters* décrivant les unités lexicales à travers plusieurs niveaux de granularité conceptuels. La hiérarchie constituée possède une structure d'arbre qui empêche l'héritage multiple des propriétés d'un niveau conceptuel à un autre (un *cluster* ne peut posséder qu'un seul parent). Les ensembles d'exemples issus de différentes branches de *clusters* sont disjoints : il n'y a pas de recouvrements possibles entre des *clusters* de même granularité.

IncrementalDBScan et *CobWeb* n'étant pas capables de prendre en compte le recouvrement entre les *clusters*, la modélisation de la polysémie lexicale est limitée. Par ailleurs, les deux méthodes ne fournissent pas les variables expliquant la formation des *clusters* : on ne connaît donc pas le contexte d'utilisation des éléments regroupés, ce qui limite d'une part l'interprétation et, d'autre part, la réutilisation de *clusters* pour désambiguïser des unités lexicales en contexte.

L'ACF a la particularité de réaliser une co-classification (ou *co-clustering*) des données en réalisant simultanément la construction de deux partitions structurées par un treillis de Galois. Les concepts formels sont des *bi-clusters* qui ont l'avantage d'être contextualisés contrairement aux *clusters* des autres méthodes : chaque concept présente une correspondance entre une partition d'objets et une partition d'attributs qui facilite l'interprétation et la réutilisation. L'héritage multiple dans les treillis autorise des recouvrements entre les descriptions des concepts. Un exemple peut ainsi se retrouver dans des branches de sens distinctes d'un treillis. En proposant une modélisation plus fine de la polysémie lexicale et de la granularité sémantique, le cadre mathématique de l'ACF apparaît comme le plus approprié pour faire progresser notre problématique de constitution automatique de BL évolutive.

L'ACF présente toutefois quelques limitations que nous devrons dépasser. D'une part, les concepts formels ne sont pas construits sur des critères de si-

milarité : ils ne fournissent donc pas nécessairement des regroupements d'unités lexicales sémantiquement homogènes d'un point de vue distributionnel. D'autre part, la taille d'un treillis peut être exponentielle en fonction du nombre d'objets et d'attributs. Cette explosion combinatoire est toutefois maîtrisable en ne développant que certaines parties du treillis lors de sa construction.

Un bon compromis serait une méthode qui puisse combiner la facilité d'interprétation des concepts formels avec la capacité de modélisation des méthodes statistiques. Dans ce mémoire, nous allons donc proposer de faire progresser les formalismes de l'ACF vers une représentation numérique interprétable dans un cadre statistique.

1.4 Conclusion

Ce chapitre nous a permis d'identifier les objectifs principaux liés à notre problématique. À cet égard, nous avons étudié différentes techniques pour construire automatiquement des BL structurées à partir de corpus non étiquetés sémantiquement.

La BL que nous cherchons à établir se focalise sur une modélisation sémantique d'UL à plusieurs degrés de granularité. Dans le cadre d'une tâche de désambiguïsation lexicale, il apparaît parfois nécessaire d'étiqueter des UL avec des unités de sens plus ou moins précises. Par ailleurs, les processus permettant de faire évoluer une BL doivent être capables de fusionner des unités de sens en des unités plus générales ou de les séparer vers des sens plus spécifiques.

Parmi les différentes techniques d'acquisition automatique employées pour constituer des BL, notre intérêt s'est focalisé vers les techniques d'apprentissage non supervisé opérant sur des corpus non étiquetés et qui évitent ainsi de faire des hypothèses sur la sémantique des UL à modéliser. Leur intérêt est de faire émerger des *clusters* sémantiquement homogènes qui regroupent des usages d'UL similaires et qui peuvent être assimilés à des unités de sens. Afin de considérer la polysémie lexicale, nous orientons nos choix vers des techniques tenant compte de recouvrements entre unités de sens. Par ailleurs, il apparaît judicieux de privilégier des techniques de *co-clustering* fournissant des *clusters* contextualisés qui facilitent l'interprétation et la réutilisabilité pour une tâche de désambiguïsation lexicale.

Le modèle de BL visé doit pouvoir intégrer de nouvelles connaissances sans bouleverser sa cohérence globale : la structure et les connaissances de la base doivent évoluer pour s'adapter à de nouvelles UL observées en corpus. De manière à constituer automatiquement des BL distributionnelles évolutives, une stratégie envisageable est de s'orienter vers des méthodes d'apprentissage incrémental non supervisé. Ces méthodes n'ont, jusqu'à présent, pas été examinées au regard de la

question de l'évolution du sens. Parmi les différentes techniques d'apprentissage que nous avons étudiées, nous avons fait émerger l'intérêt particulier de l'analyse de concept formel permettant de constituer des BL s'appuyant sur la structure de treillis de Galois. Dans le chapitre suivant, nous construisons une telle BL à partir d'un corpus à l'aide d'un algorithme incrémental original capable de traiter des données fournies en séquence.

Chapitre 2

Constitution d'une base lexicale hiérarchisée par un treillis de Galois

Dans le chapitre précédent, nous avons examiné différentes stratégies potentiellement utilisables pour constituer automatiquement des bases lexicales. Nous avons conclu qu'un processus d'apprentissage incrémental non supervisé apporterait un caractère évolutif à cette constitution. Parmi les méthodes incrémentales présentées, nous avons mis en évidence l'intérêt de l'analyse de concepts formels [Barbut et Monjardet, 1970; Ganter et Wille, 1999] (ACF) qui établit un cadre générique de *clustering* conceptuel (considéré comme de l'apprentissage symbolique non supervisé). Cette technique permet d'inférer des concepts formels définis comme des regroupements d'objets partageant un ensemble d'attributs. Ces concepts formels ont la particularité d'être organisés au sein d'une structure hiérarchisée de treillis de Galois.

Notre objectif de construction automatique de bases lexicales évolutives peut s'inscrire dans le cadre mathématique de l'ACF. À l'instar des méthodes de *clustering* statistique habituellement utilisées en analyse distributionnelle, l'ACF peut regrouper et différencier des unités lexicales selon une classification établie automatiquement. Les concepts formels d'un treillis de Galois sont en effet inférés à partir de données (dans notre cas, des usages d'unités lexicales observés en corpus) au lieu d'être prédéfinis par un expert ou spécifiés dans une base de connaissance préétablie. D'autre part, la structure hiérarchique de treillis permet de décrire et répertorier automatiquement des concepts sur plusieurs niveaux de granularité. Tout en gardant l'intégrité de sa structure globale, un treillis de Galois peut intégrer de nouvelles connaissances et faire évoluer ses concepts. Théoriquement, une base lexicale structurée par un tel treillis pourrait alors s'adapter à des éléments provenant de flux de données textuelles. Cependant, l'ajout d'une nouvelle relation entre des unités lexicales déjà intégrées peut engendrer la fusion de concepts formels : ce cas n'est pas considéré par les algorithmes incrémentaux de construc-

tion de treillis [Carpineto et Romano, 1993; Godin *et al.*, 1995; van der Merwe *et al.*, 2004], nous amenant à étendre ces algorithmes aux traitement de flux.

Dans ce chapitre, nous proposons de constituer de manière incrémentale une base lexicale hiérarchisée par un treillis de Galois. La section 1 dresse un panorama des principaux travaux traitant de l'utilisation de treillis de Galois en TAL. La section 2 détaille la méthodologie que nous avons employée pour extraire d'un corpus un ensemble de relations (contexte formel) entre des EN et des unités lexicales qui leur sont rattachées syntaxiquement. Comme nous l'avons précédemment noté, le fait de se limiter, à titre illustratif, aux EN n'est en rien une restriction, les traitements restant transposables à d'autres types d'unités lexicales. Nous présentons enfin (section 3) un algorithme incrémental original pour la construction de treillis de Galois à partir de données fournies en séquences. L'application de cet algorithme sur nos données textuelles permet d'inférer des concepts formels que nous considérons comme des unités de sens organisées dans une base lexicale structurée par un treillis de Galois.

2.1 Treillis de Galois et traitement automatique des langues

Les treillis de Galois sont utilisés à la fois en analyse de données et pour la représentation de connaissances : ils peuvent ainsi potentiellement être appliqués à une variété de problèmes en TAL. Ils apportent naturellement une représentation hiérarchique comparable aux bases lexicales structurées par relations d'hyperonymie. Par ailleurs, constitué à partir de relations entre des unités lexicales extraites de corpus, un treillis devient un outil pour la fouille de données textuelles, et les concepts formels qui le forment sont potentiellement exploitables pour de nouvelles tâches.

Cette section établit un panorama d'approches qui emploient les treillis de Galois dans une perspective de TAL. Nous montrons d'abord comment des ressources lexicales existantes peuvent être restructurées par des treillis. Nous présentons ensuite des exemples d'utilisation des treillis de Galois pour la fouille de textes permettant d'aboutir à des bases lexicales structurées. Dans un cadre d'apprentissage supervisé, nous examinons comment les treillis de Galois peuvent être employés pour des applications de TAL. Pour terminer, nous montrons l'influence des travaux étudiés dans cette section sur la méthodologie que nous allons développer.

2.1.1 Treillis de relations issues de ressources lexicales existantes

Les treillis fournissent un cadre pour la représentation structurée des connaissances issues de dictionnaires et de bases lexicales existantes. Dans cette perspective, leur intérêt pour des applications TAL a été mis en évidence par Priss [1996] et Priss et Old [2004]. Ces travaux mettent en œuvre des éléments pour modéliser des BL complexes telles que WordNet [Miller, 1995]. Les relations de synonymie représentées par les *synsets* permettent de modéliser la polysémie lexicale dans une structure de treillis. L'« analyse relationnelle de concepts » [Priss, 1998], une extension de l'ACF, se focalise sur la modélisation des relations de sémantique lexicale telles que la méronymie ou l'hyperonymie.

Ces modèles de BL structurées par des treillis pourraient être exploités pour des tâches de désambiguïsation sémantique. Notons toutefois que les unités de sens et les relations modélisées sont préétablies par expertise et ne correspondent pas à des usages d'unités lexicales observés dans des corpus. L'utilisation de techniques d'acquisition automatique du lexique en corpus couplées avec une ACF permettrait d'inférer et structurer des unités de sens à partir d'usages d'unités lexicales. Plusieurs travaux qui ont émergé en ce sens sont maintenant présentés.

2.1.2 Analyse de concepts formels pour la fouille de données textuelles

La structure de treillis entretient des liens proches avec l'apprentissage de règles d'association, qui est une méthode très utilisée en fouille de données. Dans ce cadre, Toussaint *et al.* [2000] proposent une méthode de fouille de données textuelles fondée sur les treillis de Galois et sur l'extraction de règles d'association en vue d'aider des experts dans leur tâche de veille scientifique. La stratégie proposée consiste à réaliser une ACF d'un ensemble de termes concurrents extrait d'un corpus spécialisé (les termes doivent aussi correspondre à des entrées d'un thésaurus). Un sous-ensemble des concepts formels inférés est sélectionné sur des critères probabilistes (mesures de support et de confiance, *cf.* chapitre 3). La validation de la méthode repose sur une évaluation qualitative de la pertinence des concepts obtenus par des experts du domaine. Cette méthode de validation manuelle est malheureusement trop coûteuse si l'on souhaite évaluer systématiquement la pertinence des concepts utilisés pour la désambiguïsation d'un corpus entier.

Les treillis peuvent répertorier automatiquement des informations lexicales potentiellement utiles pour des applications du TAL. Ils ont déjà été utilisés pour faire l'acquisition automatique de taxonomies à partir de corpus. À cet égard, l'ACF et son extension, l'analyse de concepts relationnels, ont été utilisées par

Bendaoud *et al.* [2008] pour intégrer des ressources hétérogènes (documents textuels, taxonomies [1] et bases de données biomédicales) au sein d'une unique ontologie formelle. Une méthodologie reposant sur l'ACF a égalemment été proposée par Cimiano *et al.* [2005] pour constituer une hiérarchie de concepts à partir de corpus.

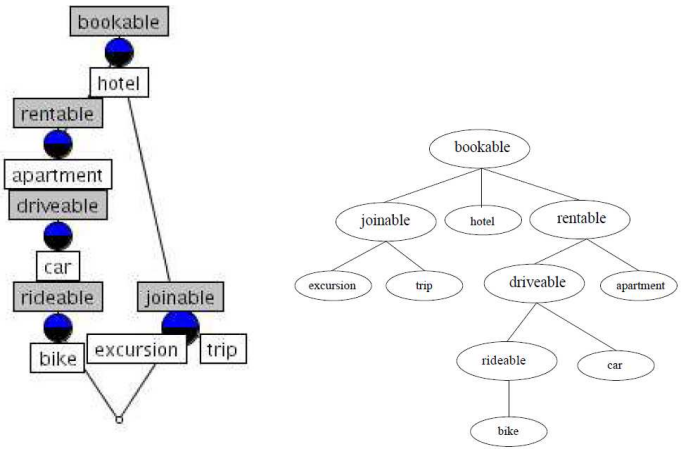


FIG. 2.1 – Un exemple de treillis de Galois (à gauche) constitué à partir d'un corpus et sa taxonomie dérivée (à droite)

Les auteurs s'appuient sur l'hypothèse distributionnelle [Harris, 1968], et procèdent à l'extraction automatique de relations entre des termes à l'aide d'un analyseur en dépendances. Pour réduire le bruit dans les données et améliorer les traitements ultérieurs, différentes techniques de lissage sont utilisées.

Le treillis de Galois des relations extraites (*cf.* figure 2.1, à gauche) est simplifié puis reconverti en un ordre partiel constituant une hiérarchie de concepts (*cf.* figure 2.1, à droite). L'approche est évaluée en comparant la hiérarchie obtenue avec des taxonomies établies manuellement pour deux domaines (tourisme et finance). Nous envisageons par la suite de valider la pertinence de ce type de base lexicale par rapport à une tâche de désambiguïsation sémantique.

Ces méthodes répondant en partie à notre but de constitution automatique de BL à partir de corpus, notre approche s'oriente précisément dans ce cadre de fouille de données textuelles. Notre méthodologie de construction de BL s'appuie sur la démarche proposée par Cimiano *et al.* [2005]. Nous allons nous en différencier sur deux points. D'une part, Cimiano *et al.* [2005] procèdent à plusieurs transformations d'un treillis qui laissent difficilement envisager sa construction incrémentale, propriété que nous souhaitons exploiter par la suite. D'autre part,

1. Une taxonomie est une représentation hiérarchique qui organise un classement de concepts.

Cimiano *et al.* [2005] ne cherchent pas à exploiter les connaissances du treillis en désambiguïsation sémantique : cette tâche de désambiguïsation est pourtant à effectuer dans notre cadre.

Plutôt que d'évaluer la pertinence des concepts obtenus face à une expertise (expertise humaine, ou bases de connaissances établies manuellement), nous proposons dans le chapitre 4 une évaluation de notre approche sur une tâche de classification supervisée d'entités nommées.

Nous examinons maintenant comment les treillis de Galois peuvent être employés pour des applications de TAL, dans le cadre de l'apprentissage de concepts en apprentissage supervisé.

2.1.3 L'espace des versions en classification supervisée

L'ACF définit un cadre pour la classification non supervisée qui peut être comparé à l'apprentissage de concepts en apprentissage supervisé [2] qui consiste à induire des hypothèses de classification à partir d'exemples étiquetés. Le problème de cette induction peut être formulé comme une recherche d'hypothèses de classification dans un espace. Cet espace contient l'ensemble des hypothèses consistantes avec les données d'apprentissage, c'est-à-dire qui décrivent au mieux les exemples d'apprentissage. D'après Mitchell [1982], cet « espace des versions » est structuré par une relation d'ordre partiel organisant l'espace des hypothèses par un treillis de Galois [3]. La structuration de ce treillis est exploitée par l'algorithme d'élimination des candidats [Mitchell, 1997] pour sélectionner des hypothèses ni trop spécifiques (risque d'apprendre par cœur), ni trop générales (risque de couvrir trop de contre-exemples).

La construction d'une BL est analogue à l'apprentissage de concepts dans l'espace des versions. De ce point de vue, notre problème peut se formuler comme l'inférence d'hypothèses de regroupements (concepts formels assimilables à des unités de sens) à partir d'exemples non étiquetés (usages d'UL en corpus). L'objectif est alors de rechercher des concepts formels, dans un treillis constitué de manière non supervisée par ACF, qui décrivent au mieux les usages d'UL observés sur corpus. Cette recherche de concepts est centrale pour notre problème d'exploitation d'une BL en désambiguïsation sémantique (*cf.* chapitre 4) : le compromis spécifique/général revient alors d'une certaine manière à choisir un niveau de granularité approprié dans une BL structurée pour désambiguïser une UL en contexte.

L'espace des versions décrit des concepts formulés dans la logique des propositions. Elle manipule les connaissances apprises dans un langage simple pour lequel

2. Pour une présentation détaillée de l'apprentissage de concepts, se référer à Mitchell [1997, chapitre 2].

3. Les liens entre l'apprentissage de concepts et l'ACF sont examinés dans [Kuznetsov, 2004].

il est facile de mettre en œuvre des opérations de généralisation/spécialisation. Cette formulation peut être étendue à la logique des prédicats. L'exploration de l'espace des versions peut ainsi être menée grâce à la programmation logique dans le cadre de la programmation logique inductive (PLI) [Muggleton et De Raedt, 1994], où la notion de treillis de subsomption joue un rôle important. La PLI fournit un langage d'hypothèses expressif qui facilite les échanges entre les communautés de linguistes et la communauté *TAL apprentissage*. Elle a notamment été utilisée acquérir automatiquement des relations lexicales à partir de textes [Sébillot, 2002; Claveau, 2003]. L'approche développée s'appuie sur une stratégie semi-supervisée qui combine apprentissage statistique non supervisé et apprentissage symbolique supervisé par PLI [Claveau et Sébillot, 2004]. Les approches numériques exploitent l'aspect fréquentiel des données, et utilisent des techniques statistiques, tandis que les approches symboliques exploitent l'aspect structurel des données [Sébillot, 2006]. Les mécanismes issus de l'apprentissage statistique complètent les techniques symboliques telles que l'ACF ou la PLI. Ces dernières sont capables de tenir compte de phénomènes rares que les méthodes statistiques ne parviennent pas à capter facilement. L'usage de techniques symboliques pourrait être particulièrement adapté à la modélisation de la polysémie pouvant nécessiter l'identification d'usages peu fréquents. En revanche, elles parviennent moins facilement à constituer un modèle synthétique des données, contrairement aux approches statistiques. Ces points de vue numérique et symbolique sont complémentaires autant en apprentissage artificiel que pour la représentation de lexiques sémantiques. Nous aurons l'occasion de développer cette réflexion en introduisant une interprétation numérique des treillis de Galois dans le chapitre 3.

Nous présentons maintenant une méthodologie pour extraire à partir de corpus des données d'apprentissage utilisables pour construire une BL.

2.2 Constitution d'un contexte formel à partir de corpus

Cette section se focalise sur l'extraction automatique de relations entre des unités lexicales à partir de corpus. Le corpus utilisé pour notre étude est issu d'une campagne d'évaluation de systèmes de reconnaissance d'EN. L'extraction de relations requiert une analyse en dépendances des énoncés qui permet d'identifier des liens entre des entités nommées et des unités lexicales. Conformément aux travaux de Cimiano *et al.* [2005], l'ensemble des relations extraites constitue un contexte formel qui pourra être utilisé ultérieurement pour construire une base lexicale structurée par un treillis de Galois.

Cette section comporte trois parties. La première présente le corpus de la campagne d'évaluation CoNLL-2003 à partir duquel nos données d'apprentissage

sont extraites. La deuxième définit notre méthodologie pour extraire des dépendances syntaxiques entre des EN et d'autres UL dans les énoncés du corpus. Pour terminer, la troisième partie présente les limitations de notre méthodologie d'extraction ainsi que quelques perspectives pour l'améliorer.

2.2.1 Corpus de la campagne CoNLL-2003

Le corpus utilisé pour notre travail est constitué d'un ensemble de dépêches issues de la presse anglophone (collection Reuters) de la campagne d'évaluation CoNLL-2003 [Tjong Kim Sang et De Meulder, 2003]. L'objectif de cette campagne était de comparer les performances de systèmes de reconnaissance d'EN ayant appris leur modèle de classification à partir de données d'apprentissage communes. Cette tâche de reconnaissance d'EN est focalisée sur la classification de quatre types d'EN : personnes, lieux, organisations et « autres » (*miscellaneous*).

Le corpus est divisé en trois parties : une partie réservée à l'apprentissage (*train*), une partie de développement (*testa*) destinée à stabiliser l'apprentissage et un corpus de test (*testb*) pour l'évaluation du modèle appris. Chaque corpus est découpé en documents qui correspondent en fait à des dépêches d'agence de presse. Ces dépêches sont à leur tour scindées en phrases. Le tableau 2.1 regroupe des caractéristiques associées à chaque corpus en précisant le nombre de dépêches, de phrases et de mots qu'il contient.

	Dépêches	Phrases	Mots
Corpus d'apprentissage (*train*)	946	14987	203621
Corpus de développement (*testa*)	216	3466	51362
Corpus de test (*testb*)	231	3684	46435

TABLE 2.1 – Caractéristiques des corpus anglophones de la campagne CoNLL-2003

Les énoncés du corpus sont segmentés en mots (*tokens*) et possèdent plusieurs niveaux de description. Le premier correspond à l'étiquetage morphosyntaxique des mots : le jeu d'étiquettes utilisé est issu du corpus du *Penn TreeBank*. Le deuxième niveau établit une segmentation des énoncés en *chunks*[4]. Ces *chunks* sont typés conformément aux spécifications de la campagne d'évaluation CoNLL-2000 [Tjong Kim Sang et Buchholz, 2000]. Il convient de noter que les deux premiers niveaux de description sont issus de traitements automatiques qui peuvent produire des résultats erronés et engendrer du bruit dégradant des

4. La notion de *chunk* introduite par Abney [1991] peut être traduite comme « syntagme non récursif ».

post-traitements (en l'occurrence, notre extraction de relations entre des EN et leurs dépendances syntaxiques). Le troisième niveau de description correspond à une délimitation et à une annotation manuelle des EN selon le jeu d'étiquettes {`PER, LOC, ORG, MISC`}. À titre d'illustration, l'énoncé extrait du corpus proposé en figure 2.2 donne un aperçu des niveaux de description exploitables.

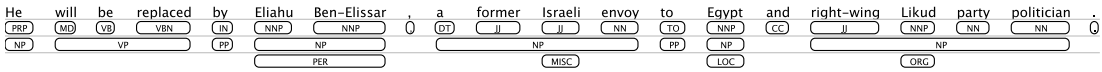


FIG. 2.2 – Échantillon du corpus Reuters utilisé dans le cadre de la campagne CoNLL-2003

Le tableau 2.2 indique pour chaque partie du corpus le nombre d'EN en fonction de leur catégorie sémantique. Le jeu d'étiquettes utilisé peut sembler limité pour décrire les entités nommées du corpus. Afin de préciser cette catégorisation des EN, la tâche *template element filling* des conférences MUC [Grishman et Sundheim, 1996] visait à remplir une fiche (prédéfinie) décrivant des EN déjà étiquetées. Dans cette perspective, nous proposons dans le chapitre 3 des méthodes d'« annotation conceptuelle » visant également à compléter un étiquetage supervisé des EN en utilisant les concepts d'un treillis de Galois.

	Lieux	Autres	Organisations	Personnes
Corpus d'apprentissage (train)	7140	3438	6321	6600
Corpus de développement (testa)	1837	922	1341	1842
Corpus de test (testb)	1668	702	1661	1617

TABLE 2.2 – Nombre d'étiquettes sémantiques associées aux entités nommées du corpus

Le jeu d'étiquettes proposé avec le corpus constitue toutefois une référence stable sur laquelle nous pouvons mesurer l'intérêt d'exploiter un treillis sur ce corpus. La constitution de ce treillis nécessite l'extraction d'attributs qui caractérisent les EN dans le corpus. Nous allons maintenant exposer notre approche pour extraire ces attributs à l'aide d'un analyseur en dépendances.

2.2.2 Analyse en dépendances pour la constitution d'un contexte formel

Nous présentons une méthodologie générale pour extraire à partir de corpus des relations entre des EN et les éléments contextuels avec lesquels elles sont en

interaction dans les énoncés. Nous cherchons ici à identifier des UL rattachées syntaxiquement à des EN de notre corpus d'étude pour répertorier les EN dans une BL structurée par un treillis. Cette démarche s'inscrit dans le cadre d'une approche harrissienne qui considère que le partage de contextes disposés selon une même configuration syntaxique constitue un indice de proximité sémantique. Nous allons ainsi utiliser un outil d'analyse en dépendances sur le corpus pour extraire des relations qui seront utilisées pour constituer un contexte formel.

2.2.2.1 Extraction des dépendances syntaxiques des entités nommées

Notre méthodologie d'extraction s'appuie sur les résultats d'un analyseur en dépendances. Il structure un énoncé comme un ensemble de dépendances représentées par un graphe d'analyse. Dans un énoncé, chaque dépendance est une relation binaire entre deux mots appelés la tête et le modifieur.

Pour obtenir une analyse en dépendances du corpus, nous avons utilisé l'outil *MaltParser*[5] [Nivre *et al.*, 2007]. Cet analyseur est capable de traiter des corpus munis d'un étiquetage morphosyntaxique compatible avec l'étiquetage du *Penn TreeBank*. *MaltParser* identifie des relations de dépendances typées par des fonctions grammaticales. La figure 2.3 est un exemple d'analyse pour l'échantillon du corpus proposé en figure 2.2.

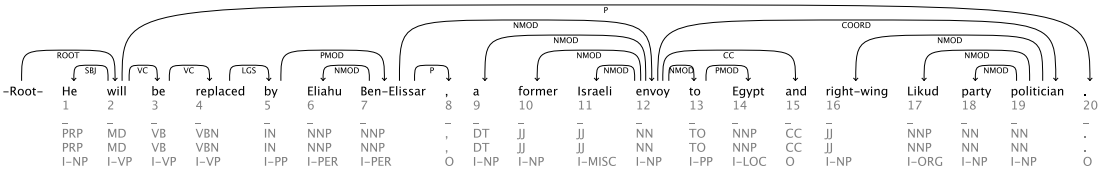


FIG. 2.3 – Analyse en dépendances d'un échantillon du corpus CoNLL-2003

Dans le graphe proposé, les dépendances sont représentées par des flèches orientées de la tête vers un modifieur. Elles peuvent être étiquetées par différents types de fonctions grammaticales : par exemple, `NMOD` (*noun modifier*) indique une gouvernance d'une expression nominale (dont les EN font partie) ; `COORD` : une relation de coordination... C'est à partir de ces listes de dépendances que nous déterminons les relations syntaxiques pertinentes établissant un lien entre une entité nommée et les éléments de son contexte.

Nous privilégions l'extraction des noms, verbes et prépositions qui entretiennent des liens syntaxiques avec des EN : les relations entre deux EN ne sont

5. L'analyseur en dépendances *MaltParser* est réputé comme étant un des meilleurs systèmes dans le cadre des campagnes d'évalution CoNLL-2006 et CoNLL-2007. Il peut être téléchargé librement à partir de `http://w3.msi.vxu.se/users/jha/maltparser`.

pas prises en compte pour le moment. Les dépendances conservées entre des EN et des expressions nominales se traduisent par des constructions du type :

- relation `NMOD` entre un modifieur nominal et une EN qui permet de faire le lien entre « Taleban » et « guards » ou « prime minister » et « Shimon Peres » ;
- relation entre une EN et un modifieur nominal `NMOD` combinée avec une relation d'attachement prépositionnel (`PMOD`) permettant de relier « envoy to » avec « Egypt ».

Les dépendances extraites entre des EN et des expressions verbales possèdent les configurations syntaxiques suivantes :

- relation entre un modifieur verbal (`VMOD`) et une entité nommée qui connecte par exemple « Pakistan » et « declared » ;
- relation entre une EN et un modifieur verbal combiné avec une relation d'attachement prépositionnel (`PMOD + VMOD`) reliant « replaced by » avec « Eliahu Ben-Elissar » ;
- relation sujet-verbe (`SUB`) où le sujet est une EN pour faire le lien entre « Arafat » et « had been scheduled » ;
- relation verbe-objet (`OBJ`), l'objet étant une EN.

Le choix manuel de ces relations syntaxiques est la seule étape supervisée de notre méthode. Ce choix dépend de l'analyseur syntaxique utilisé et de la langue du texte. Il est important de préciser que nous ne conservons pas d'informations sur le type (`NMOD`, `SUB`...) des relations extraites. Nous cherchons simplement à extraire du corpus des relations syntaxiques entre des entités nommées et des unités lexicales qui ne sont pas des entités nommées. Chaque unité lexicale extraite est appelée un **dépendant** et celui-ci peut aussi bien jouer le rôle de tête que celui de modifieur, selon la nature de sa relation avec une entité nommée.

2.2.2.2 Contexte formel des relations extraites

Notre utilisation d'un analyseur en dépendances fournit un ensemble de couples (entité nommée, dépendant) dans lesquels chaque élément est potentiellement polysémique. Cet ensemble d'informations est naturellement représenté par deux « vues » interconnectées :

- une vue sur les entités nommées qui est associée à un ensemble d'objets $O = \{o_1, o_2, \cdots, o_m\}$;
- une vue sur leurs contextes syntaxiques représentée par un ensemble d'attributs $A = \{a_1, a_2, \cdots, a_n\}$.

Ces deux vues sont connectées par une relation $R \subseteq O \times A$, où $R(o, a)$ signifie que l'objet o possède l'attribut a (*i.e.*, l'EN o possède le contexte syntaxique a). Nous appellerons dorénavant relation syntaxique chaque élément de R. Le graphe biparti proposé en figure 2.4 représente un contexte formel $\mathbb{K}$ qui illustre

un sous-ensemble de relations syntaxiques identifiées dans le corpus entre des entités nommées et leurs dépendants.

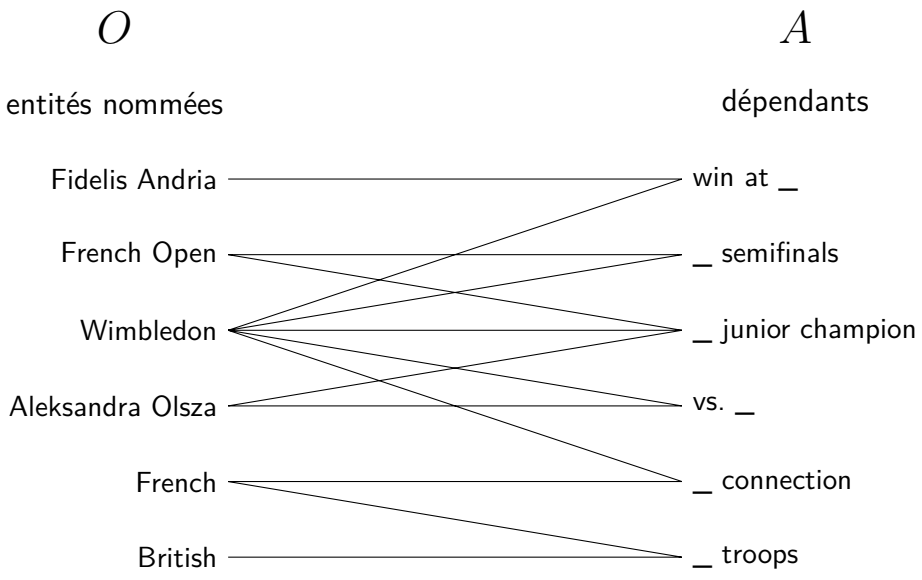


FIG. 2.4 – Contexte formel des relations entre les EN et leurs dépendants

L'extraction réalisée sur les corpus *train*, *testa* et *testb* conduit à des contextes formels dont les nombres d'objets, d'attributs et de relations sont donnés dans le tableau 2.3.

	objets	attributs	relations
train	6565	10859	21620
testa	2294	3538	6288
testb	2105	3058	5557

TABLE 2.3 – Nombres d'objets, d'attributs et de relations dans le contexte formel

2.2.3 Discussion

La technique d'acquisition décrite dans cette section se limite au corpus CoNLL-2003. Pour des travaux futurs, il serait envisageable d'exploiter d'autres ressources pour lesquelles il est facile d'obtenir des connaissances sur des entités nommées. Une première possibilité est de faire l'acquisition d'un corpus d'apprentissage à partir du Web. Un moteur de recherche interrogé avec des EN comme requêtes permet de collecter facilement des usages EN à partir des « résumés » (ou *snippets*) et des pages.

Une autre solution est d'intégrer au contexte formel des connaissances sur des entités nommées préexistantes pouvant être facilement extraites de l'encyclopédie multilingue Wikipédia : dans cette perspective, nous[6] avons travaillé sur l'extraction de connaissances liées aux entités nommées dans Wikipédia. Ce travail se focalise sur l'exploitation des catégories des articles de Wikipédia de manière à produire à terme une classification des entités nommées avec un jeu d'étiquettes beaucoup plus fin et structuré. Ces catégories étant structurées par un treillis, une perspective intéressante serait d'examiner leur recouvrement avec les concepts formels inférés à partir des occurrences d'EN associées à des articles dans Wikipédia.

Notre travail se limite pour le moment à une tâche de classification des entités nommées avec seulement quatre étiquettes. Il serait judicieux d'intégrer des unités lexicales autres que les entités nommées comme objets du contexte formel. La connectivité du contexte formel serait ainsi nettement plus élevée et pourrait renforcer sensiblement les liens entre les données acquises pendant l'apprentissage et les données observées sur le corpus de test. De plus, cette extension pourrait élargir notre approche à la construction d'une BL modélisant la sémantique de toutes les unités lexicales dans une perspective de leur désambiguïsation (voir par exemple les tâches de type *All-words Word Sense Disambiguation* dans le cadre des campagnes Senseval [Kilgarriff et Rosenzweig, 2000]).

Malgré ces restrictions au corpus CoNLL-2003, notre approche reste néanmoins acceptable pour faire progresser notre objectif de construction de bases lexicales évolutives. La section suivante présente des algorithmes incrémentaux pour constituer un treillis de Galois à partir de nos données d'apprentissage. Nous y décrivons en premier lieu l'algorithme existant *add_indent* [van der Merwe *et al.*, 2004] que nous étendons avec un nouvel algorithme dont l'originalité est d'être capable de construire un treillis à partir de données fournies séquentiellement.

2.3 Construction incrémentale à partir de données fournies en séquence

La génération d'un ensemble de concepts formels $\mathfrak{T}$ d'un contexte formel $\mathbb{K}$ et la construction du treillis $\underline{\mathfrak{T}}$ sont des problèmes bien connus. Différentes approches algorithmiques permettent de construire l'ensemble $\mathfrak{T}$ associé à $\mathbb{K}$. Des algorithmes efficaces ont été mis en œuvre pour faire face à la croissance exponentielle théorique des treillis sur des contextes formels denses. Parmi ces approches, les algorithmes incrémentaux existants sont capables d'ajouter progressivement

6. Travail réalisé au printemps-été 2007 avec Adeline Derazey lors de son stage de Master 2 Professionnel (Ingénierie multilingue à l'INaLCO).

de nouveaux objets à un treillis en y incluant des concepts. Bien que la structure du treillis soit effectivement mise à jour selon une procédure incrémentale, ces algorithmes ne sont pas réellement capables de traiter des données fournies sous forme de séquence : ils nécessitent de connaître l'ensemble des attributs en relation avec chaque objet à insérer alors que cette connaissance n'est pas disponible dans le cas de données fournies séquentiellement. À titre d'illustration, considérons une séquence S (*cf.* equation 2.1) dans laquelle chaque élément est une relation entre un objet et un attribut :

$$S = [(a,1),(b,2),(a,2),(b,1),(c,1)] \tag{2.1}$$

Chaque fois qu'une relation de S est ajoutée au treillis, l'ensemble des concepts formels devrait être transformé de la manière suivante :

$$\begin{aligned}
\mathfrak{T}_1 &= \{(\{a\},\{1\})\} \\
\mathfrak{T}_2 &= \{(\{a\},\{1\}),(\{b\},\{2\})\} \\
\mathfrak{T}_3 &= \{(\{a\},\{1\}),(\{a,b\},\{2\})\} \\
\mathfrak{T}_4 &= \{(\{a,b\},\{1,2\})\} \\
\mathfrak{T}_5 &= \{(\{a,b\},\{1,2\}),(\{a,b,c\},\{1\})\}
\end{aligned}$$

Le traitement de la séquence S montre que la taille du treillis n'est pas strictement croissante : il apparaît ici que l'intégration d'une nouvelle relation entre un objet déjà existant dans le treillis et un attribut quelconque peut engendrer la fusion de concepts. Un tel cas de fusion est illustré par le passage de l'étape ($\mathfrak{T}_3$) à l'étape ($\mathfrak{T}_4$) lors de l'intégration de la relation $(b,2)$. Ce cas de figure doit être nécessairement traité dans notre cadre : une nouvelle relation entre des éléments connus (en l'occurrence, une EN et son dépendant) peut en effet apparaître dans un flux sans que cela soit prévisible. Les algorithmes incrémentaux que nous connaissons ne peuvent toutefois pas traiter ce cas car ils ne sont capables que de créer des concepts (pas de fusion ou de suppression). Nous proposons donc ici un nouvel algorithme capable de traiter des cas de fusion lors de l'ajout de relations dans un treillis.

Dans cette section, nous présentons un premier exemple de treillis de Galois constitué à partir d'exemples de notre corpus d'apprentissage pour décrire le comportement de l'algorithme de construction incrémentale *add_intent* [van der Merwe *et al.*, 2004] et pour montrer que ce dernier nécessite des adaptations pour intégrer des données fournies séquentiellement. Nous contribuons alors à une alternative originale qui consiste à supprimer des objets ou des attributs existants dans le treillis pour faciliter leur réintégration avec de nouveaux éléments identifiés dans le flux. À l'issue de cette section, nous discutons de l'efficacité de notre solution.

2.3.1 Construction incrémentale à partir de contextes formels prédéfinis

Nous rappelons ici les notions élémentaires de l'analyse de concepts formels avant de décrire les mécanismes de l'algorithme *add_intent*. Pour illustrer nos propos, nous employons ici un exemple simple reprenant des relations extraites de notre corpus d'apprentissage. Considérons le contexte formel $\mathbb{K} = (O, A, R)$ représenté par trois premières lignes du tableau 2.4.

	_ airport	_ lead	_ midfielder	_ newsroom	_ state bank	_ scored
PSV		×	×			
PAKISTAN		×			×	
SAO PAULO	×	×		×		×

TABLE 2.4 – Exemple de contexte formel binaire $\mathbb{K}$

L'opérateur de dérivation, noté « $'$ » s'applique aussi bien aux sous-ensembles de O qu'aux sous ensembles de A. L'opérateur « $'$ » peut se composer avec lui même pour partir d'un sous-ensemble d'objet E, produire un ensemble d'attributs E' et à partir de E' produire le sous-ensemble d'objets E'' (la notation « $''$ » est utilisée pour marquer la composition).

Par exemple, l'objet $o =$ SAO PAULO est associé aux attributs de l'ensemble $\{o\}' =$ {_ airport, _ lead, _ newsroom, _ scored} et l'attribut $a =$ _ lead est en correspondance avec les objets de l'ensemble $\{a\}' =$ {PAKISTAN, SAO PAULO, PSV[7]}. Pour simplifier l'écriture, les ensembles d'attributs $\{o\}'$ et d'objets $\{a\}'$ sont respectivement notés o' et a', où $o \in O$ et $a \in A$.

Les concepts formels du contexte $\mathbb{K}$ sont les couples (E, I) tels que $E' = I$ et $I' = E$, avec $E \subseteq O$ et $I \subseteq A$. À titre d'illustration, les concepts associés à $\mathbb{K}$ sont énumérés dans la liste de la figure 2.5.

0. ({}, {_, _ lead, _ midfielder, _ scored, _ newsroom, _ airport, _ state bank})
1. ({PAKISTAN}, {_ lead, _ state bank})
2. ({PAKISTAN, SAO PAULO, PSV}, {_ lead})
3. ({SAO PAULO}, {_ scored, _ newsroom, _ lead, _ airport})
4. ({PSV}, {_ lead, _ midfielder})

FIG. 2.5 – Concepts formels associés au contexte formel $\mathbb{K}$(*cf.* tableau 2.4)

7. PSV est un club omnisports néerlandais.

Le cinquième concept formel $c_4 = (E, I)$ de cette liste possède l'extension $E = \{\text{PSV}\}$ et l'intension $I = \{_\ \text{lead}, _\ \text{midfielder}\}$. L'ensemble des concepts formels de la figure 2.5 est muni d'une relation d'ordre partiel (la relation d'inclusion ensembliste $\subseteq$) engendrant une structure de treillis $\underline{\mathfrak{T}}(\mathbb{K})$ représentée en figure 2.6.

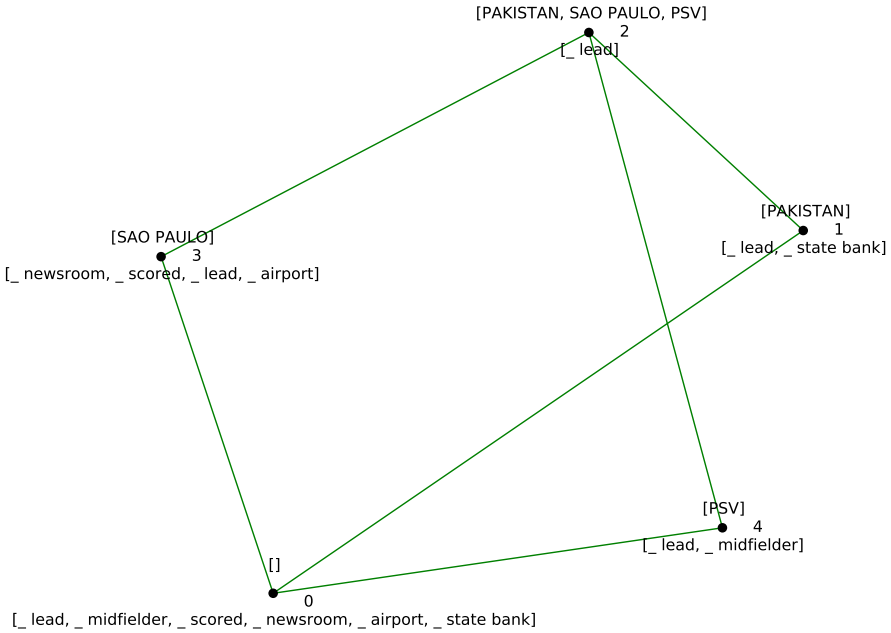


FIG. 2.6 – Treillis de Galois du contexte formel $\mathbb{K}$

Considérons maintenant que l'on cherche à intégrer au treillis de la figure 2.6 l'objet $o = \text{BELGRADE}$ muni de l'ensemble d'attributs $o' = \{_\ \text{airport}, \text{arrived in}\ _, _\ \text{newsroom}\}$. Comme l'indique le treillis de la figure 2.7, cette intégration donne lieu à la création des concepts $c5$, $c6$ et $c7$ ainsi que l'ajout de liens pour connecter ces nouveaux concepts à ceux qui existent déjà.

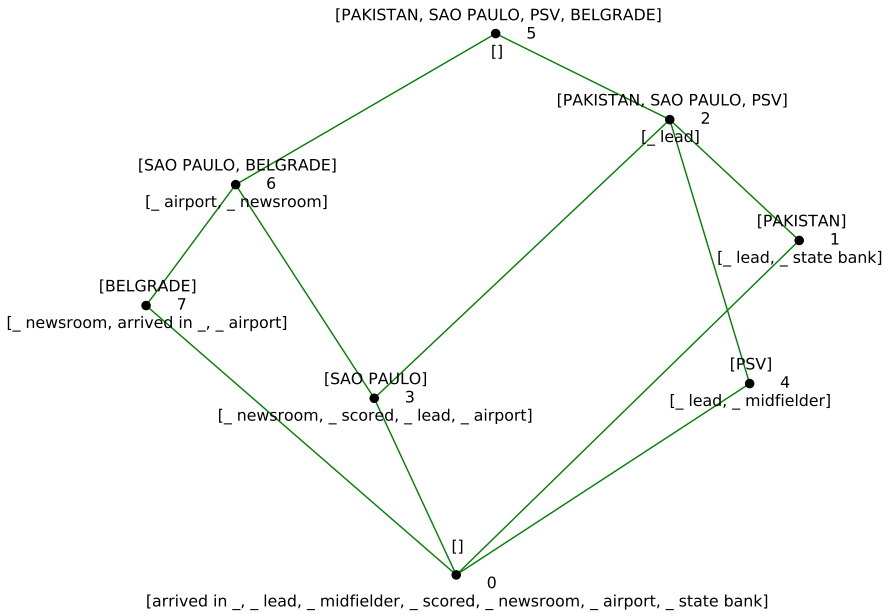


FIG. 2.7 – Modification du treillis de la figure 2.6 suite à l'intégration de l'objet $o =$ BELGRADE avec ses attributs $o' =$ {_ airport, arrived in _, _ newsroom}

Différents types d'algorithmes sont envisageables pour calculer l'ensemble de concepts formels $\mathfrak{T}$ d'un contexte formel $\mathbb{K}$. Nous avons choisi de décrire un algorithme particulier nous permettant de comprendre les mécanismes de mise à jour problématiques dans notre cadre de construction de treillis à partir de séquences. Cet algorithme, appelé *add_intent*, met en œuvre une procédure de construction incrémentale relativement peu complexe (d'un point de vue calculatoire) au regard des autres méthodes décrites dans la littérature. Nous insistons sur le fait qu'il n'est pas capable de traiter des données fournies en séquence.

Nous caractérisons ici en premier lieu les concepts formels jouant un rôle important lors de la construction incrémentale d'un treillis. Nous détaillons ensuite le fonctionnement de l'algorithme d'intégration *add_intent* qui participe à cette construction. Nous illustrons son utilisation sur le treillis de la figure 2.6 pour insérer l'ensemble d'attributs $o' =$ {_ airport, arrived in _, _ newsroom} associés à l'objet $o =$ BELGRADE et aboutir au treillis de la figure 2.7. Pour terminer, nous montrons comment les objets d'un contexte formel sont successivement ajoutés au treillis en montrant qu'une démarche guidée uniquement par *add_intent* ne permet pas de traiter des données fournies séquentiellement.

2.3.1.1 Caractérisation des concepts dans un treillis en construction

Le calcul de l'ensemble de concepts formels $\mathfrak{T}$ d'un contexte formel $\mathbb{K}$ peut être abordé de deux façons différentes. La première consiste à construire la structure du graphe associé au treillis de Galois $\underline{\mathfrak{T}}$, c'est-à-dire l'ensemble des concepts formels ainsi que les relations de subsomption établies entre eux. La seconde s'intéresse à l'introduction de contraintes (notamment par le biais de mesures de support et confiance introduites en fouille de données) afin de réduire l'espace de recherche et optimiser le calcul de sous-ensembles de concepts formels, sans chercher à établir la structure du treillis.

Dans notre cadre, nous nous intéressons aux algorithmes incrémentaux capables de construire dynamiquement l'ensemble des concepts formels et la structure du treillis en y intégrant progressivement de nouveaux objets en relation avec des attributs. Ces algorithmes (voir par exemple [Carpineto et Romano, 1993; Godin *et al.*, 1995]) ont la particularité de caractériser les nœuds du treillis uniquement par leur intension car on peut toujours retrouver leur extension à l'aide du contexte formel. Ils sont capables de modifier un treillis $\underline{\mathfrak{T}}_i$ produit par les i premiers objets d'un contexte formel en lui insérant un nouvel objet o pour générer un nouveau treillis $\underline{\mathfrak{T}}_{i+1}$. Cette construction peut être décrite en termes de quatre types de concepts : les concepts *modifiés*, les concepts *générateurs*, les concepts *nouveaux* et les concepts *anciens*.

Un concept $(C, D) \in \underline{\mathfrak{T}}_{i+1}$ est *nouveau* si son intension D n'existe dans aucun concept de $\underline{\mathfrak{T}}_i$ et que son intension $D = B \cap o'$ est égale à l'intersection de l'intension d'un concept existant (A, B), appelé concept générateur, et de l'ensemble d'attributs o' en relation avec le nouvel objet o à insérer. Chaque nouveau concept (C, D) possède au moins un générateur, mais il peut aussi en avoir plusieurs. Le concept (unique) le plus général de ces générateurs est appelé le *générateur canonique* de (C, D). Les générateurs restants de (C, D) sont appelés ses *générateurs non canoniques*. Les concepts *modifiés* sont ceux dont l'intension est incluse dans l'ensemble des attributs o' et pour lesquels il faut ajouter o à leur extension. Les concepts *anciens* sont ceux qui ne sont ni des générateurs, ni des concepts modifiés.

Un des problèmes principaux de construction incrémentale est d'identifier tous les concepts modifiés (pour ajouter o à leur extension) et tous les générateurs à associer à de nouveaux concepts. La détermination du concept générateur canonique est donc importante : ce dernier indique précisément l'emplacement où de nouveaux concepts peuvent se greffer et aussi dans quels concepts ajouter de nouveaux objets. Cette détermination, réalisée par l'algorithme 1 [van der Merwe *et al.*, 2004], met en œuvre un parcours en profondeur du treillis $\underline{\mathfrak{T}}$ pour lui ajouter l'ensemble d'attribut *intent* en partant d'un concept spécifique *generator*. Le concept *generator* est un point de départ à partir duquel on cherche le générateur

canonique de l'ensemble *intent*. Les attributs de *intent* sont initialement choisis pour être en correspondance avec un objet à ajouter. Ainsi, pour intégrer un nouvel objet o, on doit faire appel à $\underline{\mathfrak{T}}.get_intent_generator(o', \bot)$, où *generator* est initialisé comme le concept le plus spécifique $\bot$ et *intent* est assigné à l'ensemble o' des attributs en correspondance avec o dans le contexte $\mathbb{K}$.

Algorithme 1: $\underline{\mathfrak{T}}.get_intent_generator(intent, generator)$

```
1 parent_is_maximal = True
2 tant que parent_is_maximal faire
3     parent_is_maximal = False
4     pour chaque parent ∈ 𝔗.get_parents(generator) faire
5         si intent ⊆ int(parent) alors
6             generator = parent
7             parent_is_maximal = True
8             break
9 retourner generator
```

À titre d'illustration, nous reprenons l'exemple de l'intégration du nouvel objet o = BELGRADE. Le générateur canonique de l'ensemble d'attributs o' = {_ airport, arrived in _, _ newsroom} est déterminé par un appel à $\underline{\mathfrak{T}}.get_intent_generator(o', \bot)$. L'ensemble o' n'est pas inclus dans l'intension des concepts $c1$, $c3$ et $c4$, parents de $\bot = c0$. L'algorithme retourne donc $c0$ comme concept générateur de o'. À ce générateur canonique $c0$ viendra se greffer ultérieurement le nouveau concept $c7$ (*cf.* figure 2.7) suite à l'intégration de o' dans le treillis. La détermination de ce concept générateur constitue en fait une première étape dans l'algorithme d'intégration *add_intent* qui participe à la construction incrémentale du treillis et que nous décrivons maintenant.

2.3.1.2 Intégration des attributs associés à un objet

L'algorithme *add_intent* procède à un parcours récursif du treillis en profondeur en partant du nœud *generator* et retourne un concept c dont l'intension correspond à l'ensemble des attributs o' en relation avec l'objet o à insérer. La définition de *add_intent* (*cf.* Algorithme 2) peut se décomposer en trois étapes.

Pour la première (A : lignes 1 à 3), on cherche à déterminer si le concept associé à $intent = o'$ existe déjà dans le treillis par un appel à *get_intent_generator* (*cf.* Algorithme 1). Si l'intension *intent* est égale à l'intension de *generator*, alors le treillis n'a pas besoin d'être modifié et ce concept générateur est fourni comme valeur de retour. Dans le cas contraire, de nouveaux concepts doivent être introduits dans le treillis en partant du générateur canonique *generator* de *intent*.

Pour la deuxième (B : lignes 4 à 8), les parents de *generator* sont examinés

Algorithme 2: $\underline{\mathfrak{T}}.add_intent(intent, generator)$

```
1  generator = 𝔗.get intent_generator(intent, generator)
2  si int(generator) = intent alors
3      retourner generator
4  newParents = ∅
5  pour chaque candidate ∈ 𝔗.get_parents(generator) faire
6      si int(candidate) ⊄ intent alors
7          candidate = 𝔗.add_intent(int(candidate) ∩ intent, candidate)
8      newParents = min(newParents ∪ {candidate})
9  newConcept = Concept(ext(generator), intent)
10 𝔗.add_concept(newConcept)
11 pour parent ∈ newParents faire
12     𝔗.remove_link(parent, generator)
13     𝔗.set_link(parent, newConcept)
14 𝔗.set_link(newConcept, generator)
15 retourner newConcept
```

pour déterminer l'ensemble $newParents$ de concepts à connecter à $generator$ ultérieurement. Les parents candidats de $newParents$ doivent respecter deux contraintes. La première est que leur intension soit incluse dans $intent$ (c'est-à-dire o', lors de l'appel initial de add_intent) ; dans le cas contraire, il est nécessaire d'introduire au treillis au moins un concept c entre (o'', o') (où $o' = intent$ et o'' est l'ensemble des objets associés à o') et le parent courant $candidate$. L'appel récursif à add_intent permet de réaliser cette intégration et retourne un concept qui se substitue au parent courant $candidate$. Le nouveau concept $candidate$ est alors ajouté à l'ensemble $newParents$. La seconde contrainte est que l'ensemble $newParents$ doit être minimal, c'est-à-dire qu'il n'existe pas de relation de parenté entre les concepts de $newParents$ et que les concepts plus généraux que d'autres sont supprimés. L'appel de $min(newConcept \cup \{candidate\})$ retourne la borne supérieure de $newConcept$ augmentée du candidat courant et garantit que la seconde contrainte soit satisfaite.

La troisième et dernière étape (C : lignes 9 à 15) consiste à mettre à jour la structure du treillis en modifiant les relations de généralisation/spécialisation. Cette opération est réalisée en connectant le nouveau concept $newConcept$ aux éléments de $newParents$, en déconnectant $generator$ des concepts de $newParents$ et en connectant $newConcept$ avec $generator$. L'élément $newConcept$ qui vient d'être intégré au treillis est alors retourné en sortie de l'algorithme.

2.3.1.3 Exemple

Voyons concrètement le fonctionnement de cet algorithme sur le treillis (*cf.* figure 2.6) comportant les objets de {PAKISTAN, PSV, SAO PAULO} du contexte formel représenté dans la table 2.4. Nous montrons ici comment insérer au treillis l'ensemble d'attributs $o' =$ {_ airport, arrived in _, _ newsroom} associé à l'objet $o =$ BELGRADE . L'insertion de o' engendre la création et la modification de concepts que nous illustrons à travers les figures 2.8, 2.9 et 2.10. Chaque première colonne de ces figures désigne une étape de l'algorithme identifiée à l'aide de la notation suivante :

- A, B et C correspondent aux instructions situées respectivement aux lignes 1-7, 4-8 et 9-15 de l'Algorithme 2 ;
- L'étape B peut être décomposée en plusieurs sous-étapes B1, B2... qui correspondent à chaque fois à une itération de boucle examinant un parent candidat (*cf.* ligne 6-8 de l'algorithme) ;
- le niveau de récursivité des appels à *add_intent* est aussi pris en compte : l'étape Bx.C correspond par exemple à un deuxième niveau de récursivité et By.Bz.C, à un troisième niveau.

Seules les actions susceptibles d'engendrer une modification du treillis sont examinées ici. Lors de l'appel initial de $add_intent(\underline{\mathfrak{T}}, o', \bot)$, le concept $\bot = c_0$ est le générateur maximal pour l'intension o'. À l'étape B1, le premier concept parent de *generator* est $c1 =$ ({PAKISTAN}, {_ lead, _ state bank}). L'intension de ce parent candidat n'est pas incluse dans o', ce qui engendre l'exécution de $add_intent(\text{int}(c1) \cap o', c1)$. Dans ce nouvel appel (étape B1.A), le générateur de $\text{int}(c1) \cap o' = \emptyset$ est $c2 =$ ({PAKISTAN, SAO PAULO, PSV}, {_ lead}). Il ne possède pas de parent car il correspond à la borne supérieure $\top$ du treillis. Un concept $c5 =$ ({PAKISTAN, SAO PAULO, PSV}, $\emptyset$) est alors créé pour être ensuite connecté à $c2$ (étape B1.C). Notons qu'étant plus général que $c2$, $c5$ devient la borne supérieure $\top$.

Suite à l'appel initial de *add_intent*, l'étape B2 (*cf.* première ligne du tableau en figure 2.9) examine le parent candidat $c3$. Le fait que $\text{int}(c3) \not\subset o'$ engendre l'exécution de $add_intent(\{$_ newsroom, _ airport$\}, c3)$ (étape B2.B1). Cette opération donne lieu à l'appel $add_intent(\emptyset, c2)$ car $c2$ est parent candidat de $c3$ et $\text{int}(c2) \cap$ {_ newsroom, _ airport}$= \emptyset$. Le générateur de l'intension $\emptyset$ devient $c5$ à l'étape B2.B1.A : $c5$ est alors affecté à *candidate* pour l'étape B2.B1. Un nouveau concept $c6$ est créé à l'étape B2.C pour être ensuite connecté à son enfant, le générateur $c3$, et à son parent $c5$ alors que le lien entre $c5$ et $c3$ disparaît.

En observant le tableau et le treillis de la figure 2.10, nous pouvons décrire de manière similaire la création de $c7$, sa connexion avec $c6$ et $c0$ et la suppression du lien $(c6, c0)$. Nous devons remarquer ici que les intensions des concepts $c5$, $c6$ et $c7$ resteront inchangées même après l'insertion de nouveaux objets. En revanche, les extensions de ces concepts doivent être complétées avec l'objet initial BELGRADE suite au premier appel de $\underline{\mathfrak{T}}.add_intent(\{\text{BELGRADE}\}', \bot)$. L'ajout de cet

étape	*intent*	*generator*	*candidate*	int(*candidate*) $\cap$*intent*	*newParents*
B1	o'	$c0$	$c1$	$\emptyset$	$\emptyset$
B1.A	$\emptyset$	$c2$	-	-	-
B1.C	Création du concept $c5$ avec int($c5$) $= \emptyset$, connexion de $c5$ avec $c2$				

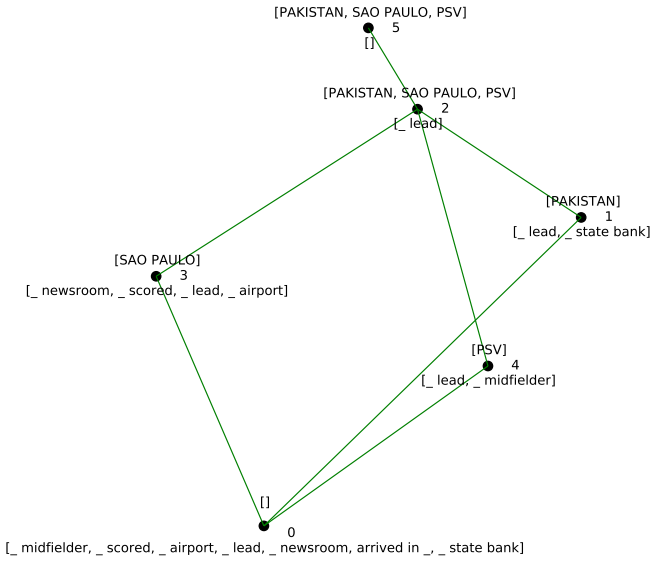


FIG. 2.8 – Ajout de l'objet $o =$ BELGRADE dans le treillis de la figure 2.6 : évolution du treillis et trace des variables manipulées par l'algorithme lors de la création du concept $c5$

étape	*intent*	*generator*	*candidate*	int(*candidate*) ∩*intent*	*newParents*
B2	o'	$c0$	$c3$	{_ newsroom, _ airport}	$\{c5\}$
B2.B1	{_ newsroom, _ airport}	$c3$	$c2$	$\emptyset$	$\emptyset$
B2.B1.A	$\emptyset$	$c2$	-	-	-
B2.B1	{_ newsroom, _ airport}	$c3$	$c5$	$\emptyset$	$\{c5\}$
B2.C	Création du concept $c6$ avec int($c6$) ={_ airport, _ newsroom}, création des liens ($c5$, $c6$) et ($c6$,$c3$), suppression de ($c5, c3$)				

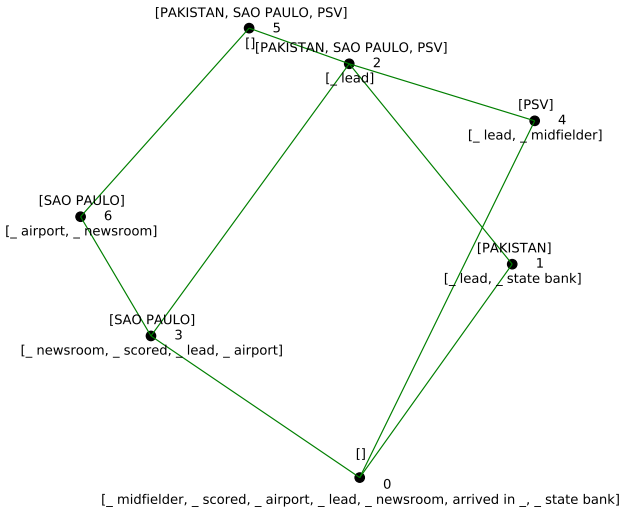


FIG. 2.9 – Ajout de l'objet o = BELGRADE : évolution du treillis et trace des variables manipulées par l'algorithme lors de la création du concept $c6$

étape	*intent*	*generator*	*candidate*	int(*candidate*) $\cap$*intent*	*newParents*
B3	$\emptyset$	$c0$	$c4$	$\emptyset$	$\{c6\}$
B3.C	Création de $c7$ avec int($c7$) ={_ newsroom, arrived in _, _ airport}, connexions des liens (c6,c7) et (c7,c0), suppression de (c6,c0)				

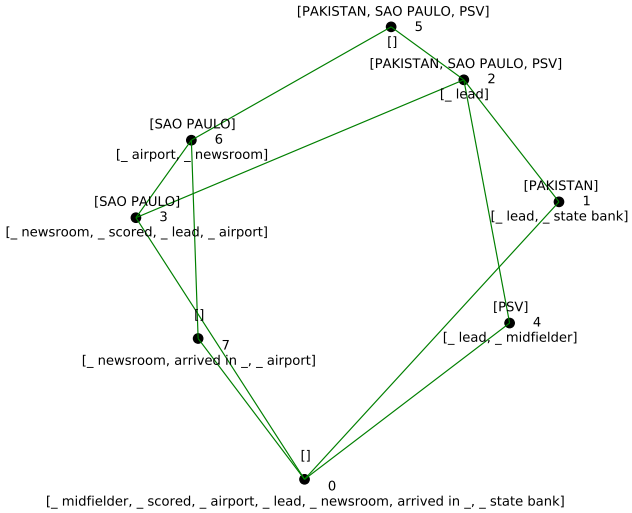


FIG. 2.10 – Ajout de l'objet o = BELGRADE : évolution du treillis et trace des variables manipulées par l'algorithme lors de la création du concept $c7$

objet est réalisé dans le cadre de la construction incrémentale d'un treillis que nous décrivons maintenant.

2.3.1.4 Construction incrémentale d'un treillis de Galois

La procédure générale de construction (*cf.* Algorithme 3) initialise $\perp$ (le concept le plus spécifique de $\underline{\mathfrak{T}}$) avec comme intension, les attributs A du contexte formel $\mathbb{K}$. Chaque appel de *add_intent* peut compléter les intensions des concepts existants et créer de nouveaux concepts afin d'y intégrer les attributs o' associés à o. La fonction renvoie le concept c contenant l'intension o' et l'extension o''. On ajoute enfin o à l'extension de tous les concepts $parents \geq c$. La procédure utilisée est un simple parcours de graphe (en profondeur ou en largeur) à la recherche des concepts $c_i \geq c$.

Algorithme 3: construction_incrémentale($\mathbb{K}$)

```
1 ⊥ = (∅, A)
2 𝔗 = {⊥}
3 pour chaque o ∈ O faire
4 |   c = 𝔗.add_intent(o', ⊥)
5 |   pour chaque parent ≥ c faire
6 |   |   ext(parent) = ext(parent) ∪ {o}
7 retourner 𝔗
```

Nous avons déjà souligné que cette construction incrémentale ne permet pas de prendre en compte des données fournies en séquence. La raison est que l'algorithme *add_intent* et les autres méthodes de construction incrémentales s'appuient généralement sur le théorème fondamental de l'analyse de concepts formels [Wille, 1982] selon lequel la borne supérieure (O_i, A_i) de deux concepts (O_j, A_j) et (O_k, A_k) est telle que $A_i = A_j \cap A_k$ Une conséquence de ce théorème est que les concepts déjà présents dans le treillis ne sont jamais supprimés lors de sa construction incrémentale. Seule l'extension d'un concept peut être modifiée lors de l'ajout d'un objet. De nouveaux objets peuvent être insérés progressivement dans le treillis mais leurs attributs en relation doivent être nécessairement connus pour une intégration correcte. L'utilisation de l'algorithme 3 ne garantit pas que l'ajout de nouveaux attributs associés à o puisse se faire correctement par la suite, surtout si ces attributs sont déjà présents dans le treillis. Pour pallier ce problème, nous présentons maintenant une extension à *add_intent* permettant le traitement de données fournies en séquence.

2.3.2 Adaptations aux contextes formels présentés en séquence

Comme nous l'avons déjà indiqué dans cette section, la version actuelle de l'algorithme *add_intent* est incapable de supprimer ou de fusionner des concepts. Une extension est nécessaire pour que de nouvelles relations puissent être correctement ajoutées au treillis. La contribution présentée maintenant est une méthode originale permettant d'ajouter une relation quelconque au treillis. L'intégration de relations est ici mise en œuvre à l'aide d'un nouvel algorithme capable de supprimer un objet dans le treillis.

Pour illustrer nos propos, nous reprenons les données étudiées dans cette section, en les représentant cette fois par la séquence S de relations suivante :

$S =$ (PSV, _ lead),(SAO PAULO, _ airport), (BELGRADE, _ airport),(PAKISTAN, _ lead), (SAO PAULO,_ scored), (PAKISTAN, _ state bank), (SAO PAULO,_ newsroom), (BELGRADE, _ newsroom), (PSV, _ midfielder), (BELGRADE,arrived in _),(SAO PAULO,_ lead).

Le treillis représenté en figure 2.11 est construit avec toutes les relations de S sauf la dernière, $(o, a) =$ (SAO PAULO,_ lead), que l'on cherche à intégrer.

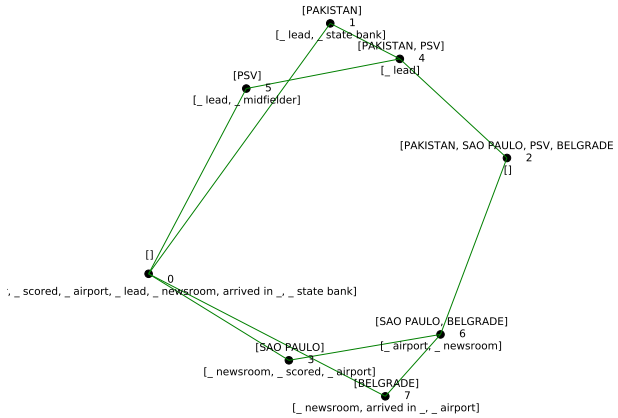


FIG. 2.11 – État du treillis avant l'intégration de la relation $o =$ (SAO PAULO,_ lead)

Nous nous intéressons ici à l'intégration d'une relation quelconque (o, a) dans un treillis. Nous proposons de réaliser cette intégration en supprimant un objet o qui existe déjà dans le treillis puis en le réintégrant avec tous ses attributs identifiés dans le flux. Nous présentons ici successivement deux algorithmes originaux illustrés par notre exemple. Le premier contribue à modifier la structure du treillis engendrée par la suppression d'un objet. Le second nous permet d'ajouter une

relation quelconque en supprimant éventuellement des objets qui apparaissent à plusieurs reprises dans le flux.

2.3.2.1 Suppression d'un objet

Dans un flux, l'ajout d'une relation quelconque peut engendrer la création, la suppression ou la fusion de concepts. Dans cette perspective, nous [8] avons mis au point la procédure $\underline{\mathfrak{T}}.update(o, c, children)$ (*cf.* Algorithme 4) qui réalise un parcours du graphe $\underline{\mathfrak{T}}$ en profondeur (de type *depth-first-search*) à la recherche de concepts à supprimer contenant l'objet o. Les paramètres d'initialisation sont un objet o à supprimer, le concept $c = (o'', o')$ associé à o et un ensemble vide $children$.

Algorithme 4: $\underline{\mathfrak{T}}.update(o, c, children)$

```
1  X = ext(c) − {o}
2  si 𝔗.get_children(c) == {d}, avec d = (X, X′) alors
3      parents = 𝔗.get_parents(c)
4      𝔗.remove(c)
5      pour chaque p ∈ parents faire
6          𝔗.update(o, p, max(children ∪ {d}))
7  sinon
8      flag = True
9      pour chaque child ∈ children faire
10         pour chaque child_i ∈ 𝔗.get_children(c) faire
11             si child ≤ child_i alors
12                 flag = False
13                 break
14         si flag alors
15             𝔗.set_link(c, child)
```

Nous distinguons ici deux cas. Le premier (lignes 2 à 6) consiste à supprimer un concept c du treillis lorsqu'il possède un unique descendant d tel que $\text{ext}(d) = \text{ext}(c) - \{o\}$. En dehors de l'objet o à supprimer, le concept c n'ajoute pas d'information par rapport à son descendant d. Il convient toutefois de mémoriser les liens d'héritage de c, à savoir la relation entre son enfant unique $d = (X, X')$ et les parents p de c. L'ensemble $children \cup \{d\}$ contient des concepts qu'il faudra éventuellement reconnecter avec les parents p de c lors d'un nouvel appel récursif de $\underline{\mathfrak{T}}.update$ (ligne 6). L'appel de $max(children \cup \{d\})$ permet de garantir que l'ensemble $children \cup \{d\}$ ne contient aucun concept plus spécifique que d'autres,

8. Merci à Sergei Obiedkov pour son aide précieuse à la mise au point de l'algorithme.

évitant de connecter accidentellement un parent p avec deux concepts déjà reliés lors de l'appel de $\underline{\mathfrak{T}}.update$.

Le second cas (lignes 7 à 15) traite le problème de connexion du concept c avec les descendants *children* de concepts préalablement supprimés. Le concept c est cette fois conservé dans le treillis car il contient au moins deux objets (dont o) et possède au moins un descendant d ne pouvant pas remplacer c en cas de suppression. Il convient de rétablir certains liens hiérarchiques entre les enfants *children* d'un concept précédemment supprimé et c, un de ses parents. Ces connexions ont lieu uniquement entre c et les concepts $child \in children$ si aucun concept $child_i$ n'existe parmi les enfants de c tels que $child \leq child_i$. En d'autres termes, aucun enfant $child_i$ de c ne doit s'interposer entre c et $child$.

Suite à l'appel initial de $\underline{\mathfrak{T}}.update(o, (o'', o'), \emptyset)$, la structure du treillis est telle que si o n'avait jamais été intégré dans le treillis.

À titre d'illustration, nous donnons maintenant une trace de cet algorithme sur l'exemple développé dans cette section.

2.3.2.2 Exemple de suppression

Nous disposons du treillis de la figure 2.11 dans lequel nous cherchons à supprimer l'objet $o =$ SAO PAULO associé aux attributs $o' =$ {_ airport, _ newsroom, _ scored} et au concept $(o'', o') = c3 =$ ({SAO PAULO}, {_ newsroom, _ scored, _ airport}). Comme l'indique la figure 2.12, cette suppression entraîne la suppression de $c3$ puis de $c6$. La figure montre également l'évolution du treillis ainsi que l'état des variables manipulées par l'algorithme lors de la suppression de $o =$ SAO PAULO.

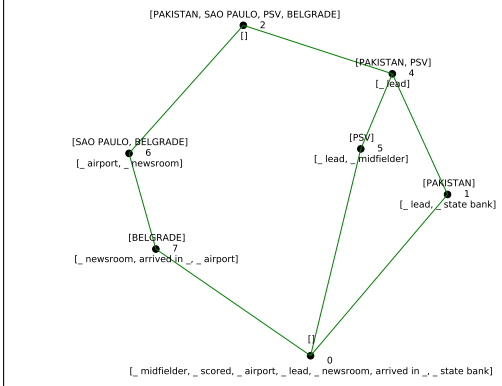
La première ligne du tableau en haut à gauche de la figure 2.12 montre que la suppression de o est initiée par un premier appel de $\underline{\mathfrak{T}}.update(o, c3, \emptyset)$. Le concept $c3$ est supprimé car il possède un enfant unique $d = c0$ avec $\text{ext}(c0) = \text{ext}(c3) - o$ (*cf.* premier cas de l'algorithme 4). Les liens de parentés de $c3$ avec son parent $c6$ et son enfant unique $c0$ disparaissent du treillis. Ces liens sont mémorisés lors de l'appel récursif de $\underline{\mathfrak{T}}.update(o, c6, \{c0\})$, $c0$ étant le seul élément de *children* et correspondant ainsi à la borne supérieure de $\underline{\mathfrak{T}}$.

En suivant une démarche similaire, l'appel $\underline{\mathfrak{T}}.update(o, c6, \{c0\})$ (*cf.* deuxième ligne du tableau en haut à droite de la figure 2.12) engendre la suppression de $c6$ et de ses liens avec son seul parent $c2$ et son enfant unique $c7$. Cette suppression donne aussi lieu l'exécution de $\underline{\mathfrak{T}}.update(o, c2, \{c7\})$, où l'enfant unique $c7$ est le seul élément de $max(\{c0, c7\})$.

À l'appel à $\underline{\mathfrak{T}}.update(o, c2, \{c7\})$ (*cf.* première ligne du tableau en bas de la figure 2.12), le concept $c2$ possède un enfant unique $c4$; en revanche on a $\text{ext}(c4) \neq \text{ext}(c2) - o$. L'algorithme 4 entre cette fois dans le deuxième cas de figure (lignes 7 à 15) : le concept $c2$ est alors connecté à l'unique élément de $children = \{c7\}$ car il n'existe aucun concept c dans le treillis tel que $c7 < c < c2$.

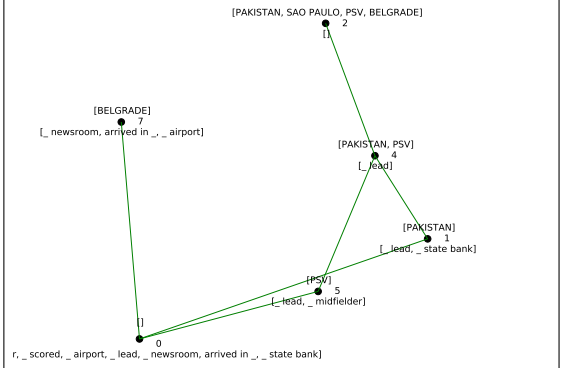
concept	ext(*concept*) − *o*	*children*
$c3$	$\emptyset$	$\emptyset$

Suppression de $c3$



concept	ext(*concept*) − *o*	*children*
$c6$	{BELGRADE}	$\{c0\}$

Suppression de $c6$



concept	ext(*concept*) − *o*	*children*
$c2$	{PAKISTAN, PSV, BELGRADE}	$\{c7\}$

Création du lien $(c2, c7)$

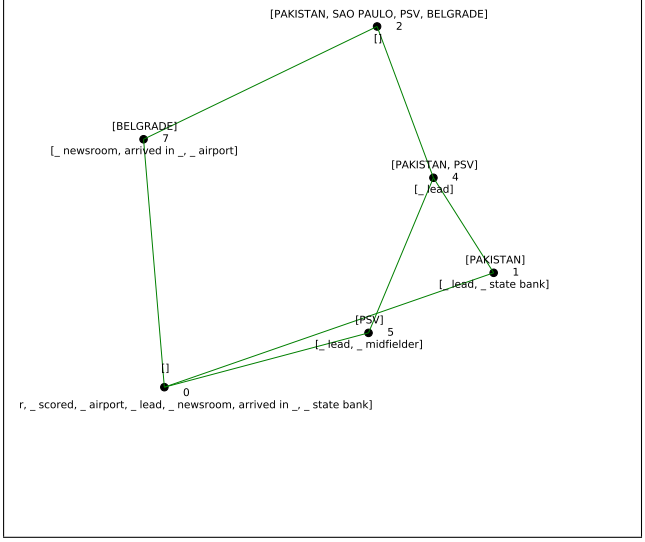


FIG. 2.12 – Suppression de l'objet o = SAU PAULO : évolution du treillis et des variables manipulées par l'algorithme 4 lors de la suppression du concept $c3$ (en haut, à gauche), de la suppression de $c6$ (en haut, à droite) et de la création du lien $(c2, c7)$ (en bas, à gauche)

Il est important de noter que l'objet o subsiste dans l'extension de la borne supérieure $\top$ ($c2$ sur la figure 2.12) et les attributs spécifiques à o sont eux aussi présents dans l'intension de la borne inférieure $\bot$ ($c2$ sur la figure 2.12). La suppression de o et de ses attributs spécifiques dans $\top$ et $\bot$ peut être réalisée simplement. Cette suppression n'est toutefois pas nécessaire dans notre cas car o et ses attributs spécifiques seront réintégrés ultérieurement dans le cadre de l'algorithme d'ajout de relations que nous présentons maintenant.

2.3.2.3 Ajout d'une relation quelconque

Nous décrivons ici comment procéder à la transformation du treillis lorsqu'une nouvelle relation est ajoutée à un contexte formel. Des données en provenance d'un flux peuvent comporter des relations dont les éléments (objets ou attributs) sont nouveaux ou existent déjà dans le treillis. L'intégration d'une relation (o, a) est effectuée par un appel à $\underline{\mathfrak{T}}.add_relation(o, a)$ (*cf.* Algorithme 5) qui traite différents cas.

Algorithme 5: $\underline{\mathfrak{T}}.add_relation(o, a)$

Données : $\mathbb{K}_i = (O_i, A_i, R_i), \underline{\mathfrak{T}}(\mathbb{K}_i)$

```
1  si (o, a) ∉ R_i alors
2      K_{i+1} = (O_i ∪ {o}, A_i ∪ {a}, R_i ∪ {(o, a)})
3      si a ∈ A_i alors
4          si o ∈ O_i alors
5              T.update(o, (o'', o'), {})
6          c = T.add_intent(o', ⊥)
7          T.add_extent_above_concept(c, o)
8      sinon
9          c = T.add_extent(a', ⊤)
10         T.add_intent_below_concept(c, a)
11         si o ∉ O_i alors
12             T.add_extent_above_concept(c, o)
```

La solution pour ajouter une relation (o, a) entre un nouvel objet et un attribut quelconque (*cf.* lignes 6 et 7 de l'Algorithme 5) ne diffère pas de celle donnée pour l'Algorithme 3 : il convient de faire un appel à la fonction $\underline{\mathfrak{T}}.add_intent(o', \bot)$ puis d'ajouter l'objet o à tous les concepts plus généraux que le concept (o'', o') renvoyé par *add_intent*.

C'est presque la même chose pour une relation entre un nouvel attribut et un objet quelconque (*cf.* lignes 9 et 10). Nous avons toutefois besoin d'un algorithme

que nous appelons *add_extent* qui est une version duale de *add_intent*. Il y a une symétrie algorithmique exacte entre l'ajout d'un objet avec un ensemble d'attributs dans le treillis, et l'ajout d'un attribut avec un ensemble d'objets. De ce fait, la modification du treillis est effectuée par un appel à $\underline{\mathfrak{T}}.add_extent(a', \top)$ pour insérer le concept (a', a''), puis, par une mise à jour de ses concepts plus spécifiques, en ajoutant l'attribut a à leur intention. Si o n'existe pas non plus dans le treillis (lignes 11-12), il convient de l'ajouter aux extensions des concepts $c \geq (a', a'')$.

Un cas plus complexe se présente pour une nouvelle relation (o, a) entre un objet et un attribut déjà connus dans le treillis et son contexte formel $\mathbb{K}$. La stratégie adoptée (lignes 4 et 5) pour insérer une relation (o, a) dans un treillis consiste à d'abord supprimer o du treillis puis à l'ajouter à nouveau avec son nouvel attribut a (grâce à *add_intent*).

Pour illustrer le comportement de l'algorithme 5, nous reprenons le treillis de la figure 2.11 dans lequel nous souhaitons intégrer la relation $(o, a) =$ (SAO PAULO, _ lead). Les éléments o et a sont déjà présents dans le treillis : l'insertion de (o, a) donne lieu au traitement le plus complexe décrit aux lignes 4 et 5 de l'algorithme. Pour ce cas, l'objet $o =$ SAO PAULO doit être préalablement supprimé grâce à un appel de $\underline{\mathfrak{T}}.update(o, (o'', o'), \{\})$ détaillé en figure 2.12. Pour réintégrer o avec l'ensemble de ses attributs $o' =$ {_ lead, _ scored, _ airport, _ newsroom}, l'algorithme *add_relation* procède ensuite à l'appel de $\underline{\mathfrak{T}}.add_intent(o', \bot)$ (ligne 6) puis à l'ajout de o aux extensions des concepts c dont l'intention est incluse dans o' (appel à $\underline{\mathfrak{T}}.add_extent_above_concept(c, o)$, ligne 7).

Notre algorithme de mise à jour fonctionne donc sur le petit exemple simple décrit dans cette section. Le treillis modifié par la réintégration de o représenté en figure 2.13 possède exactement les mêmes concepts et la même structure que le treillis de la figure 2.7, construit avec l'algorithme *add_intent* sur les mêmes données, à la différence que *add_relation* traite ces mêmes données séquentiellement. Nous avons également vérifié sur nos données que notre algorithme, fonctionnant à partir de flux, construit le même treillis que l'algorithme *add_intent* sur les mêmes données fournies en une seule fois. Cependant, il est important de préciser que nous n'apportons pas ici de preuve formelle de la validité de notre algorithme. En terme de temps de calcul, il est bien entendu avantageux d'employer notre algorithme sur un flux plutôt que de construire systématiquement un nouveau treillis avec *add_intent* sur l'intégralité de nos données.

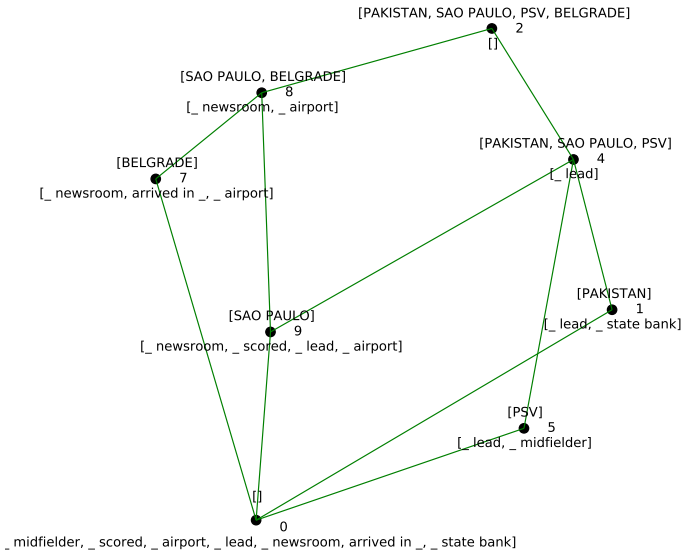


FIG. 2.13 – État du treillis après la réintégration de l'objet $o =$ SAU PAULO avec ses attributs $o' =$ {_ lead, _ scored, _ airport, _ newsroom} dans le treillis de la figure 2.12

L'algorithme *add_relation* peut maintenant être employé pour construire un treillis à partir des données issues de notre corpus. À cet égard, nous examinons maintenant l'évolution du nombre de concepts en fonction de la quantité de données fournies.

2.3.2.4 Croissance du treillis

Contrairement à d'autres techniques de classification non supervisée, l'ACF ne compresse pas les données d'apprentissage comme un algorithme de *clustering* statistique. Par exemple, un algorithme de *clustering* de type *k-means* répartit un ensemble d'objets O dans des *clusters* dont le nombre k est généralement nettement inférieur à $|O|$. Théoriquement, le nombre de concepts $|\mathfrak{T}|$ d'un treillis de Galois complet croit exponentiellement en fonction du nombre de relations entre les objets et les attributs.

En pratique, le contexte formel associé aux données du corpus *train* correspond à une matrice creuse (constituée d'une majorité de cases vides dans le tableau binaire), évitant ainsi une explosion combinatoire lors de la construction du treillis. Le graphique donné en figure 2.14 montre l'évolution du nombre de

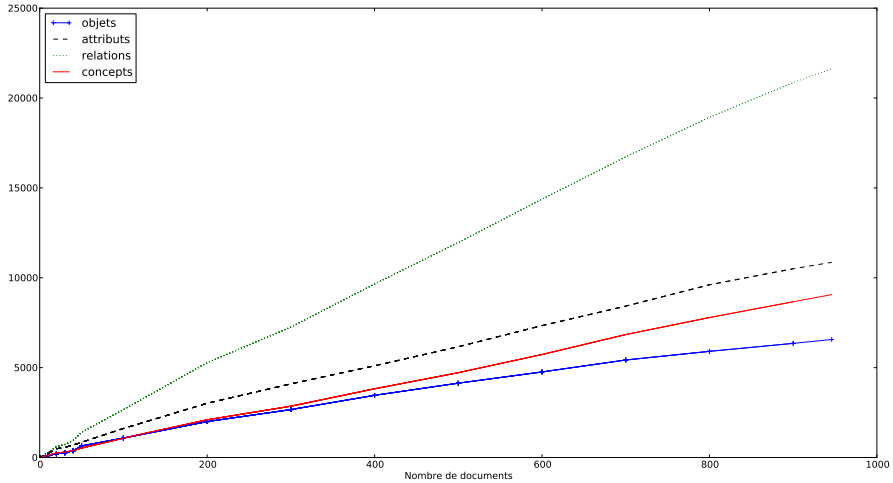


FIG. 2.14 – Évolution du nombre d'objets, d'attributs, de relations et de concepts formels en fonction du nombre de documents observés dans le corpus *train*

concepts en fonction du nombre d'objets $|O|$ et du nombre d'attributs $|A|$ insérés dans le treillis. Il indique qu'avec les données extraites du corpus *train*, la taille du treillis est majorée par le nombre d'attributs et minorée par le nombre d'objets alors que le nombre de relations est nettement plus élevé. Les treillis constitués à partir des corpus *train*, *testa* et *testb* possèdent respectivement 9063, 2413 et 2138 concepts formels. Ces corpus seront utilisés ultérieurement pour des expérimentations exploitant des techniques d'apprentissage artificiel (*train* et *testa* pour la phase d'apprentissage et *testb* pour la phase de test).

2.3.3 Bilan

Jusqu'à présent, la construction incrémentale des treillis faisait appel à des procédures incapables d'intégrer correctement des données fournies en séquence : la mise à jour d'une structure de treillis existante était problématique lorsque de nouvelles relations entre des éléments connus apparaissaient dans le flux. Pour résoudre ce problème, nous avons mis au point un nouvel algorithme (*cf.* Algorithme 5) qui ajoute une relation quelconque à un treillis existant en procédant à la suppression préalable d'un objet existant dans le treillis (*cf.* Algorithme 4). Cette solution n'est probablement pas la plus efficace d'un point de vue calculatoire. À cet égard, nous pourrions partir de l'algorithme de construction *Divide & Conquer* [Valtchev et Missaoui, 2001; Valtchev *et al.*, 2002] qui permet de construire un treillis de Galois en réalisant un assemblage de treillis établis à l'aide de contextes formels différents. En principe, cet algorithme devrait être capable d'intégrer une nouvelle relation dans un treillis déjà établit en mettant

à jour l'ensemble de ses concepts formels. Notre technique de suppression d'éléments laisse toutefois une plus grande marge de manœuvre si un utilisateur (ou un programme) cherche à retoucher la base lexicale en supprimant (manuellement ou automatiquement) des relations inconsistantes *a posteriori*. Comme perspective, nous pourrons établir les liens entre *Divide & Conquer* et notre algorithme pour aboutir à une procédure de construction optimisée tout en offrant flexibilité pour la manipulation de treillis.

2.4 Conclusion

Ce deuxième chapitre a présenté un premier modèle de base lexicale hiérarchisée par un treillis de Galois construit à partir de corpus. Après une brève étude sur l'utilisation des treillis de Galois en TAL (section 1) nous avons présenté les données qui seront utilisées dans nos expérimentations (section 2) : elles sont issues d'une extraction préalable d'unités lexicales rattachées syntaxiquement à des entités nommées dans des corpus. Une analyse de concepts formels sur ces données permet d'inférer un ensemble de concepts regroupant des EN en relation avec leurs contextes syntaxiques. Afin de réaliser un traitement des données fournies en séquence, nous avons mis au point un algorithme original pour la construction incrémentale d'un treillis de concepts (section 3). Nous proposons maintenant de faire le bilan des contributions de ce chapitre au regard de notre problématique générale.

Concernant la représentation de la BL constituée, les travaux existants sur les treillis de Galois s'intègrent aux différentes problématiques exposées au cours du premier chapitre. L'ACF est une technique d'apprentissage non supervisé capable d'inférer automatiquement des concepts formels et de les organiser dans une représentation structurée sur plusieurs niveaux de granularité. De plus, les concepts formels constituent des regroupements « contextualisés », c'est-à-dire qu'ils établissent des regroupements d'objets et d'attributs s'expliquant mutuellement. Hormis l'intérêt d'interprétabilité, nous ne savons pas encore comment exploiter cette contextualisation pour une tâche de désambiguïsation lexicale. Il est important de remarquer que, contrairement aux *clusters* déterminés avec des techniques de *clustering*, les concepts inférés par ACF ne sont pas optimisés sur des critères de similarité. Nous ne pouvons donc pas affirmer que tous les concepts formels de la structure sont sémantiquement homogènes et, par conséquent, assimilables à des unités de sens. À première vue, il semble donc qu'une notion de similarité fasse défaut à notre modèle de BL.

Concernant les aspects incrémentaux, nous avons élaboré un nouvel algorithme capable de faire évoluer la structure de treillis en fonction de nouvelles relations issues d'un flux. À l'instar des autres algorithmes incrémentaux exis-

tants, notre méthode est capable d'intégrer de nouveaux concepts à une structure existante en fonction de nouveaux objets. Notre apport consiste à pouvoir intégrer des relations entre des objets et des attributs existants pouvant mener à la suppression ou la fusion de concepts.

La procédure que nous avons proposée réalise des opérations successives d'ajout et de suppression d'éléments dans le treillis, qui tendent à complexifier les traitements. Une mise à jour plus efficace de la structure devrait éviter la suppression systématique d'objets à réinsérer : dans cette perspective, nous suggérons de déterminer au préalable les concepts destinés à être modifiés dans le treillis sous l'influence d'une nouvelle relation (ce point sera précisé dans le chapitre 4 dans le cadre de notre problématique de désambiguïsation lexicale). Une autre optimisation envisageable serait d'intégrer au treillis chaque objet avec tous ses attributs relatifs dans un document (en l'occurrence une dépêche) plutôt que d'ajouter chaque relation successivement : cette optimisation présuppose que les éléments du flux correspondent à des documents et non des relations.

Dans le chapitre suivant, nous analysons quelques propriétés du treillis construit nous permettant de le considérer comme une base lexicale. À cet égard, nous examinons les concepts et les relations de ce treillis à l'aide d'outils d'analyse de données afin d'identifier les propriétés linguistiques pertinentes pour établir un modèle de sémantique du treillis en vue de son exploitation en désambiguïsation sémantique.

Chapitre 3

Analyse des données linguistiques d'un treillis de Galois lexical

Nous venons de proposer une méthodologie générale pour constituer — de manière incrémentale — une base lexicale répertoriant des relations entre des unités lexicales extraites d'un corpus. Cette base lexicale regroupe et différencie ces relations au sein de concepts formels organisés dans un « treillis de Galois lexical ». Nous nous appliquons ici à étudier en quoi la base lexicale construite fournit une représentation lexicale proche de la « réalité » des mots en usage. Nous cherchons en fait à établir un modèle sémantique reposant sur le treillis construit pour une exploitation en désambiguïsation lexicale. Notre réflexion est s'appuie sur deux hypothèses de travail. La première est que certains concepts sont capables d'exprimer les usages du corpus et peuvent à ce titre être assimilés à des unités de sens. La seconde hypothèse concerne la structuration globale du sens et suppose que le treillis de Galois modélise la polysémie lexicale en organisant une différenciation d'unités de sens.

Dans ce chapitre, nous examinons nos deux hypothèses de travail en nous penchant sur une analyse des données issues de notre corpus. Pour étudier notre première hypothèse, nous nous consacrons à l'essai de critères de sélection des concepts formels représentatifs des mots en usage dans le corpus. Pour la seconde, nous cherchons à modéliser la polysémie lexicale qui sous-tend la structure de treillis inférée en analysant les différents niveaux de granularité qui répartissent des concepts sur une échelle allant de valeurs très générales à très spécifiques. Par ailleurs, chaque unité lexicale polysémique étant associée à plusieurs concepts formels, nous nous appliquons à caractériser les différences qui se manifestent entre ces derniers, et qui seraient en mesure de discriminer le sens en contexte, autrement dit, de permettre une désambiguïsation.

Ce chapitre comporte trois sections. Nous étudions d'abord (section 1) la structure de treillis d'un ensemble de concepts associés à deux unités lexicales

polysémiques en relation pour proposer une première méthodologie de sélection de concepts représentatifs des usages du corpus. Nous employons ensuite (section 2) des mesures nous permettant d'identifier des concepts possédant des intensions ou extensions sémantiquement homogènes assimilables à des unités de sens. Par ailleurs, les recouvrements observés entre les concepts formels nous incitent à mettre en œuvre une modélisation graduelle de la polysémie aboutissant à une interprétation nouvelle des treillis (section 3) qui repose sur une représentation sémantique continue.

3.1 Structuration d'unités lexicales polysémiques

Nous examinons ici la structure d'une sélection de concepts formels contenant une relation entre une entité nommée et une unité lexicale polysémiques. Nous mettons ici en évidence des règles d'implication entre les attributs ou les objets des concepts grâce aux relations de subsomption du treillis, en observant les changements entre différents niveaux de granularité du treillis. Ces règles d'implication, plus communément appelées règles d'association, regroupent des usages apparaissant dans les mêmes contextes et qui pourraient exprimer une sémantique proche.

Le plan de cette section est le suivant. Nous étudions d'abord un exemple montrant la structure d'un sous-treillis associé à une entité nommée et une unité lexicale en relation, celles-ci étant polysémiques. Nous mettons ensuite en évidence des règles d'implication entre les différents niveaux conceptuels de ce sous-treillis en indiquant qu'elles peuvent potentiellement expliquer l'usage d'une unité lexicale en le généralisant à d'autres usages rencontrés dans des contextes proches. Pour terminer, nous analysons quelques exemples de règles, sélectionnées sur des critères de fiabilité existants, en discutant de la pertinence de ces règles pour discriminer le sens en contexte.

3.1.1 Treillis d'une relation entre des unités polysémiques

Notre objectif est de comprendre comment la structure hiérarchisée d'un treillis de Galois, établie par des relations d'inclusion entre les intensions et extensions des concepts formels, permet d'organiser la sémantique d'UL sur différents niveaux de granularité. Nous avons choisi d'illustrer cette étude en examinant le sous-ensemble des concepts formels munis de la relation $(o, a) = (\textit{Australian, results})$, les résultats obtenus étant bien sûr généralisables à d'autres cas. Les concepts sont énumérés dans la liste de la figure 3.1.

1. ({RUSSIAN, AUSTRALIAN, ISRAELI}, {police, results, government})
2. ({AUSTRALIAN, ARGENTINE}, {coach, results})
3. ({SWISS, RUSSIAN, AUSTRALIAN, ISRAELI}, {results, government})
4. ({AUSTRALIAN, ENGLISH}, {city, league matches, league standings, results})
5. ({PORTUGUESE, ISRAELI, U.S. OPEN, HONG KONG OPEN, PHILIPPINE, FRENCH, SWISS, MAJOR LEAGUE BASEBALL, ARGENTINE, MOROCCAN, ENGLISH, RUSSIAN, AUSTRALIAN, HONG OPEN, MAJOR LEAGUE, BELGIAN}, {results})
6. ({RUSSIAN, AUSTRALIAN, ISRAELI, BELGIAN}, {police, results})
7. ({AUSTRALIAN, ENGLISH, BELGIAN}, {city, results})
8. ({AUSTRALIAN}, {purchaser, jail breaker, coach, diplomats, cabinet minister, ministry official, results, nuns, fellow, team manager, embassy, rising, captain, city, said, police, opened, for, rugby league premiership standings, feed barley, parrot scam lands, australian, wheat, government, hired, parrots, minister, league matches, share market, of, league standings})
9. ({AUSTRALIAN, ISRAELI}, {police, results, minister, government})
10. ({AUSTRALIAN, BELGIAN}, {city, police, results})
11. ({AUSTRALIAN, U.S. OPEN}, {of, results, for})

FIG. 3.1 – Concepts formels associés au treillis de la figure 3.2

Dans cette liste, l'unité lexicale *results* (*cf.* concept c_5 sur les figures 3.1 et 3.2) est polysémique : elle peut désigner par exemple les résultats de la politique d'un gouvernement, d'une compétition sportive ou d'une compagnie cotée en bourse. Comme l'indique le concept c_8 (*cf.* figures 3.1 et 3.2), l'EN *Australian* fait référence à la nationalité australienne associée à toutes sortes de facettes : un gouvernement (*governement* : ORG), une ville (*city* : LOC), une ambassade (*embassy* : ORG), un entraîneur (*coach* : PER)...

La figure 3.2 permet de mieux comprendre comment l'ensemble des concepts formels s'organise dans un treillis. Sur cette figure, les extensions sont placées au-dessus d'un concept (en majuscules) et les intensions sont placées au-dessous (en minuscules). Pour simplifier la figure et éviter les redondances, les attributs et les objets n'apparaissent qu'une seule fois en tirant profit de la double relation d'héritage entre les intensions et extensions des concepts. La description de chaque concept c est réduite en ne conservant que ses **attributs propres** $\text{int}_{\circlearrowleft}(c)$ et ses **objets propres** $\text{ext}_{\circlearrowleft}(c)$. L'ensemble des attributs propres $\text{int}_{\circlearrowleft}(c)$ correspond aux éléments d'un concept c qui n'appartiennent pas aux intensions des concepts plus généraux (au sens de $\leq$). L'ensemble des attributs hérités $\text{int}^{\downarrow}(c)$ correspond aux éléments d'un concept c qui appartiennent aux intensions des concepts plus généraux.

$$\text{int}^{\downarrow}(c) = \bigcup_{c<parent} \text{int}(parent) \tag{3.1}$$

$$\text{int}_{\circlearrowleft}(c) = \text{int}(c) - \text{int}^{\downarrow}(c) \tag{3.2}$$

La relation $\geq$ sur l'ensemble des concepts permet de définir de manière analogue l'ensemble des *objets propres* $\text{ext}_{\circlearrowleft}(c)$ et des *objets hérités* $\text{ext}_{\uparrow}(c)$.

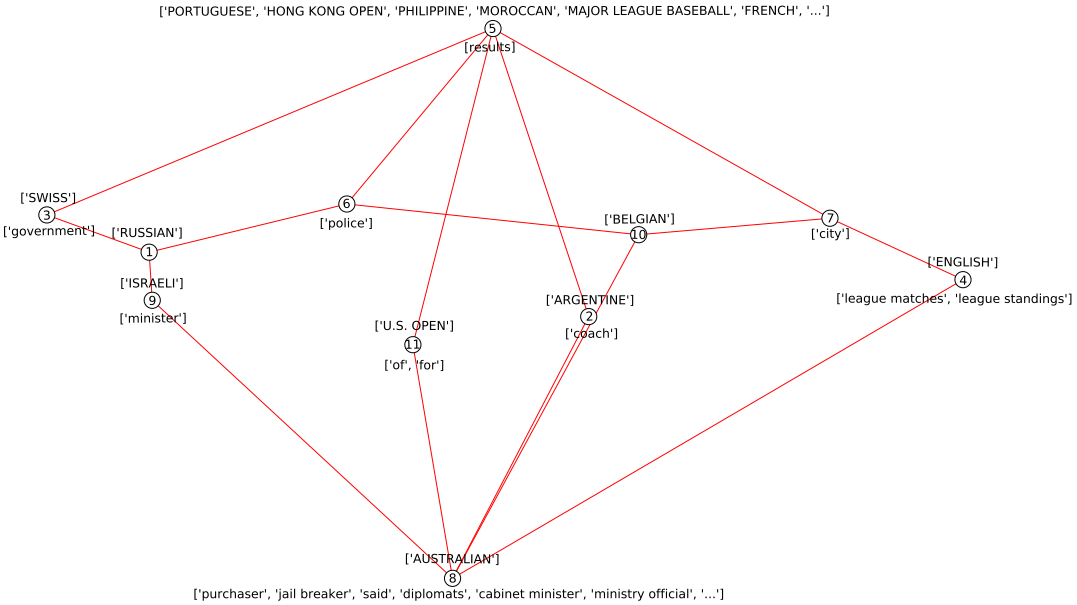


FIG. 3.2 – Sous-treillis des concepts contenant la relation (*Australian, results*)

$$\text{ext}_{\uparrow}(c) = \bigcup_{enfant<c} \text{ext}(enfant) \tag{3.3}$$

$$\text{ext}_{\circlearrowleft}(c) = \text{ext}(c) - \text{ext}_{\uparrow}(c) \tag{3.4}$$

Les attributs et les objets hérités $\text{int}^{\downarrow}(c)$ et $\text{ext}_{\uparrow}(c)$ sont supprimés de la figure 3.2. Certains concepts peuvent paraître vides mais possèdent en fait des objets hérités de leurs parents et/ou des attributs hérités de leurs enfants. Cette représentation réduite met en évidence les objets ou les attributs responsables d'une différenciation entre des concepts entretenant une relation de subsomption. De plus, ces différenciations expriment des règles d'implication entre des ensembles d'attributs ou d'objets que nous allons maintenant considérer.

3.1.2 Règles d'implication entre niveaux conceptuels

D'après Toussaint *et al.* [2000], chaque concept formel c permet de générer des règles d'implication indiquant que l'emploi d'un attribut propre $a_i \in \text{int}_{\circlearrowleft}(c)$ est associé à l'emploi des attributs hérités $\text{int}^{\downarrow}(c)$ de c : la règle pour chaque attribut propre a_i est notée $a_i \rightarrow \text{int}^{\downarrow}(c)$. Ces règles d'implication ajoutent une sémantique supplémentaire aux concepts formels. Par exemple, le concept c_5, très générique pour le nom commun *results*, se dérive en des concepts plus spécifiques qui précisent son contexte d'utilisation. Ainsi, le concept c_4 du treillis (*cf.* figure 3.2)

possède deux attributs propres $\text{int}_{\circlearrowleft}(c_4) = \{\textit{league standings, league matches}\}$ et deux attributs hérités $\text{int}^{\downarrow}(c_4) = \{\textit{city, results}\}$ à partir desquels les règles suivantes sont générées :

$$\begin{aligned} r_1 &: \ \textit{league standings} \ \rightarrow \textit{city}, \textit{results} \\ r_2 &: \ \textit{league matches} \ \rightarrow \textit{city}, \textit{results} \\ r_3 &: \ \textit{league matches} \ \leftrightarrow \textit{league standings} \end{aligned}$$

La règle r_3 exprime l'équivalence mutuelle des attributs propres *league matches* (matchs du championnat) et *league standings* (classements du championnat) qui s'interprète comme suit : les attributs *league matches* (matchs du championnat) et *league standings* (classements du championnat) sont employés dans les mêmes contextes. Les règles r_1 et r_2 s'interprètent de la façon suivante : l'emploi des attributs *league standings* ou de *league matches* est associé à celui de *city* et *results*. De manière analogue, d'autres règles impliquant *police*, *governement* et *minister* décrivent d'autres facettes de *results*.

En complément aux implications entre les attributs, des règles peuvent être découvertes entre des EN (objets du contexte formel). À l'instar de *Australian*, la plupart des extensions situées entre $\top$ et $\bot$ correspondent à des entités nommées ($English, Belgian, Israeli, Swiss$) qui auraient été étiquetées comme `MISC` dans le corpus. On retrouve aussi un nom de pays pour c_2 (*Argentine* : `LOC`) et un nom de championnat de tennis pour c_{11} (*U.S. Open* : `MISC`). Les concepts de ces deux dernières EN sont situés dans des branches indépendantes du treillis n'ayant pas de relation avec les autres concepts : leur description est moins riche en intension ou en extension. D'autre part, ces branches possèdent moins de niveaux conceptuels et leurs concepts font intervenir moins de règles d'implication que ceux situés dans d'autres branches.

Les propriétés structurelles des treillis de Galois permettent ainsi d'organiser des unités de sens sur plusieurs niveaux de granularité. La partie supérieure du treillis englobe des concepts généraux qui regroupent plusieurs objets partageant des attributs polysémiques. À l'opposé, la partie inférieure présente des concepts très spécifiques qui contiennent des objets polysémiques. Les concepts appartenant à la zone intermédiaire respectent un compromis entre spécifique et général. On retrouve d'ailleurs ce compromis entre spécifique et général dans le cadre d'inférence d'hypothèses de classification dans l'espace des versions [Mitchell, 1997]. Dans cette zone, il est remarquable que les tailles des intensions et des extensions tendent à s'équilibrer. Dans le cas d'une relation (o, a) entre un objet et attribut (typiquement l'exemple de l'usage *Australian results*), les regroupements d'objets et d'attributs des concepts intermédiaires sont capables d'expliquer l'usage d'un objet o en le généralisant à d'autres objets rencontrés dans des contextes proches (définis par des attributs). De manière similaire, ces

concepts intermédiaires peuvent expliquer l'usage d'un attribut a en le généralisant à d'autre attributs rencontrés dans des contextes proches (définis par des objets). Ce compromis générique/spécifique pourrait donc potentiellement réaliser une désambiguïsation.

Nous ne pouvons toutefois pas prétendre que toutes les implications et généralisations inférées de nos corpus peuvent être considérées comme des indicateurs fiables pour sélectionner des concepts formels capables de représenter des unités de sens et de désambiguïser des unités lexicales polysémiques. À cet égard, nous proposons maintenant de faire une sélection de concepts en évaluant la validité des règles d'implication plus communément appelées des *règles d'association*.

3.1.3 Extraction de règles d'association

Le nombre très important de concepts générés par la construction d'un treillis constitue une difficulté importante pour l'interprétation humaine. Pour sélectionner les concepts les plus informatifs du treillis, on peut avoir recours à l'extraction de règles d'association [Agrawal et Srikant, 1994]. Cette méthode de fouille de données est une technique non supervisée d'apprentissage artificiel pour la découverte de relations fines entre des attributs. Les règles d'association correspondent à des règles de type « SI condition, ALORS conséquence » : elles se formalisent sous forme de règles d'implication entre des ensembles d'attributs telles que celles que nous avons décrit précédemment. L'extraction de ce type de règles a montré son utilité dans plusieurs domaines d'applications tels que la gestion de la relation client en grande distribution (analyse du panier de la ménagère pour déterminer les produits souvent achetés simultanément afin d'agencer les rayons et d'organiser les promotions en conséquence), la biologie moléculaire (analyse des associations entre gènes)... Nous allons voir ici que les concepts formels instanciés dans un treillis permettent d'identifier des règles d'association [Zaki et Ogihara, 1998; Toussaint *et al.*, 2000]. Plus précisément, il est possible de générer un ensemble de règles d'association d'objets et d'attributs en examinant les différences ensemblistes entre les intensions/extensions d'un concept général et celles d'un concept plus spécifique.

Par souci de clarté, nous conservons ici la terminologie employée en analyse de concepts formels pour définir les principales notions utilisées dans le domaine de l'extraction de règles d'association. L'extraction de règles est habituellement pratiquée sur une base de données binaire qui correspond à un contexte formel $\mathbb{K} = (O, A, R)$ où chaque objet $o \in O$ (ou transaction) est en relation avec des attributs (ou *items*). Une **règle d'association** est une expression r du type $X \rightarrow Y$, où l'antécédent $X \subseteq A$ et le conséquent $Y \subseteq A$ sont des sous ensembles d'attributs (ou *itemset*) qui n'ont pas d'attributs communs. Pour faire face à l'explosion du nombre d'*itemsets* générés, les algorithmes d'extraction recherchent

de façon exhaustive les règles d'association dont certaines mesures dépassent des seuils fixés au préalable par un utilisateur.

Soient $|X'|$ et $|Y'|$ les nombres d'objets en relation avec les *attributs* de X et de Y. Le nombre d'objets à la fois en relation avec X et Y est $|X' \cap Y'|$. Le *support* d'une règle $X \rightarrow Y$ est la proportion d'objets en relation avec X et Y :

$$support(X \rightarrow Y) = P(X, Y) = \frac{|X' \cap Y'|}{|O|} \tag{3.5}$$

La *confiance* est la proportion d'objets vérifiant Y, parmi ceux qui vérifient X, c'est-à-dire la probabilité conditionnelle de Y sachant X :

$$conf(X \rightarrow Y) = P(Y|X) = \frac{P(X, Y)}{P(X)} \tag{3.6}$$

La notion de support d'une règle peut être étendue aux concepts formels d'un contexte formel $\mathbb{K} = (O, A, R)$. Pour l'intension d'un concept formel $c = (E, I)$, le support est défini de la manière suivante :

$$support_a(c) = \frac{|I'|}{|O|} = \frac{|E|}{|O|} \tag{3.7}$$

Le nombre d'objets $|X' \cap Y'|$ de la formule 3.5 correspond en fait au nombre d'objets $|E| = |I'|$ associés à l'intension I du concept c pour les associations d'attributs.

La notion de règle montrée jusqu'ici établit des implications entre des attributs. La structure de treillis étant duale, il est aussi envisageable de manipuler des règles associant des objets. De manière similaire, nous pouvons définir le support d'une extension comme le nombre d'attributs $|I| = |E'|$ par rapport au nombre d'attributs total $|A|$:

$$support_o(c) = \frac{|E'|}{|A|} = \frac{|I|}{|A|} \tag{3.8}$$

Notons qu'il est parfois plus aisé d'utiliser la fréquence d'une règle d'attributs (respectivement règle d'objets) qui correspond simplement au cardinal des extensions (respectivement des intensions) du concept invoqué.

Selon Toussaint *et al.* [2000], un concept $c_1 = (E_1, I_1)$ qui subsume $c_2 = (E_2, I_2)$ permet de générer une règle d'association où $\text{int}(c_1)$ désigne l'intension de c_1 et $\text{int}_{\circlearrowleft}(c_2)$ correspond aux attributs propres de c_2. La confiance d'une règle d'attributs est calculée de la manière suivante :

$$conf_a(I_1 \rightarrow \text{int}_{\circlearrowleft}(c_2)) = \frac{|I_2'|}{|I_1'|} = \frac{|E_2|}{|E_1|} \tag{3.9}$$

Nous pouvons calculer la confiance d'une règle d'objets de manière similaire :

$$conf_o(E_2 \rightarrow \text{ext}_{\circlearrowleft}(c_1)) = \frac{|I_1|}{|I_2|} \tag{3.10}$$

Pour éclaircir cette notion de confiance dans les treillis, nous avons repris la figure 3.3 initialement proposée par Zaki et Ogihara [1998]. Les concepts de ce demi-treillis sont représentés par des couples de la forme (extension × intension) et par leurs éléments propres (en gras). Notons que les objets et les attributs utilisés dans cet exemple font référence à un exemple « jouet » qui n'est pas issu de notre corpus et qui ne se rapporte pas à notre réflexion sur la modélisation de la polysémie lexicale. Les arcs du treillis, orientés vers le bas, sont annotés par la confiance des règles d'association entre attributs. À titre d'illustration, nous pouvons calculer la confiance de la règle d'association d'attributs (*cf.* équation 3.9) générée entre les concepts $c_1 = (12345, CW)$ et $c_2 = (245, CDW)$, où $c_1 > c_2$:

$$\begin{aligned} conf_a(\text{int}(c_1) \rightarrow \text{int}_{\circlearrowleft}(c_2)) &= conf_a(CW \rightarrow D) \\ &= \frac{|\{C,W,D\}'|}{|\{C,W\}'|} \\ &= \frac{|\{2,4,5\}|}{|\{1,2,3,4,5\}|} \\ &= \frac{3}{5} \end{aligned}$$

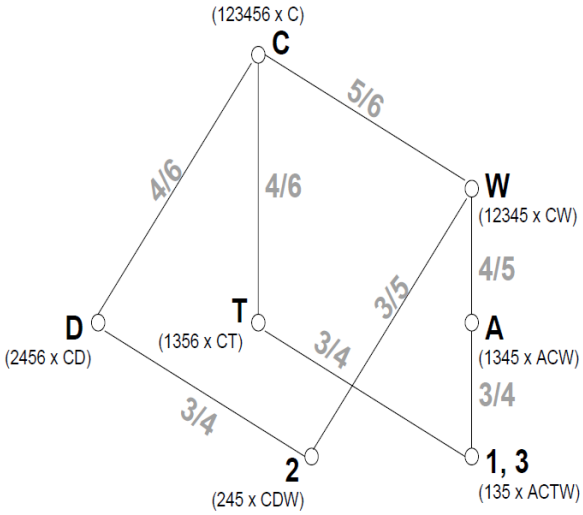


FIG. 3.3 – Confiances des règles d'association pondérant les arcs d'un demi treillis

L'utilisation combinée des mesures de confiance et de fréquence (équivalente au support) permet de sélectionner des règles d'association d'attributs. À titre d'illustration, certaines d'entre elles ont été extraites de notre corpus :

Confiance	Condition $\text{int}(c_1)$	Conséquence $\text{int}_{\circlearrowleft}(c_2)$	Objets impliqués $\text{ext}(c_2)$
0.67	television, ambassador	message	U.S., Israel, Algerian
0.67	convention	party	Democratic, GOP, Republican
0.67	prime minister, troops, foreign minister	soldiers	Russian, Turkish

TABLE 3.1 – Exemples de règles d'attributs extraites à partir de concepts avec $min_freq = 2$ et $min_conf = \frac{2}{3}$

Comme l'indique la deuxième ligne du tableau 3.1, la confiance de la règle d'attributs *convention* → *party* (« SI *convention* ALORS *party* » ou « l'unité lexicale *party* s'emploie dans le même contexte que *convention* ») est de 0.67. Cette règle est vérifiée par trois objets : *Republican*, *GOP*[1] et *Democratic* qui désignent ici des partis politiques. Ces règles permettent de décrire des unités lexicales avec des ensembles d'attributs potentiellement informatifs dans un processus de désambiguïsation lexicale. De manière similaire, des associations d'objets respectant une confiance et une fréquence minimale sont présentées ci-dessous (*cf.* table 3.2).

Confiance	Condition $\text{ext}(c_1)$	Conséquence $\text{ext}_{\circlearrowleft}(c_2)$	Attributs impliqués $\text{int}(c_2)$
0.77	Weizman	Arafat	scheduled, quoted, invited, consult, talks, called, said,
0.67	Lindsay Davenport	Olympic	gold medalist,champion
0.67	Chechnya, Middle East	Liberia, Israel	peace, war

TABLE 3.2 – Exemples de règles d'objets extraites à partir de concepts

La dernière ligne du tableau 3.2 indique que la confiance de la règle d'objets $Chechnya, MiddleEast \rightarrow Liberia, Israel$ est de 0.67. Cette règle est vérifiée par les attributs *peace* et *war*.

La génération de règles d'association à partir de concepts fournit une interprétation simple de la structuration des unités lexicales dans un treillis. Nous avons observé que les concepts avec des règles d'attributs ayant un support élevé sont plutôt situés dans la zone supérieure du treillis. Ils ont tendance à posséder beaucoup d'objets, peu d'attributs et sont trop généraux pour préciser suffisamment le contexte d'usage d'une unité lexicale. De manière similaire, les règles d'objets ayant un support élevé ne semblent pas convenir pour décrire précisément l'usage d'entités nommées. Par ailleurs, il peut être difficile d'interpréter un ensemble de règles atteignant un très grand volume. Ces observations nous incitent à associer

1. *Grand Old Party* (GOP) est un nom donné au parti républicain américain.

les unités lexicales d'un corpus avec les concepts d'un treillis en essayant d'identifier ceux qui seraient les plus appropriés pour expliquer les usages de ces unités lexicales. Dans cette perspective, nous considérons à présent le treillis comme une base lexicale comparable à une ressource lexicale électronique (telle qu'un thésaurus) en supposant que les concepts formels peuvent être assimilés à des unités de sens capables de modéliser la polysémie lexicale.

3.2 Polysémie et recouvrements des sens

Dans une perspective de désambiguïsation lexicale, nous considérons les treillis de Galois comme des ressources sémantiques comparables à des dictionnaires électroniques ou des thésaurus : par analogie avec des définitions ou des *synsets*, les concepts formels sont vus comme des unités de sens décrivant la sémantique d'unités lexicales.

Dans cette section, nous commençons par annoter les occurrences d'unités lexicales du corpus par les concepts qui leur sont associés dans le treillis. Cette étude nous amène à rechercher des concepts constitués d'éléments (objets ou attributs) peu polysémiques et similaires d'un point de vue distributionnel. Des propriétés de continuité apparaissent dans le treillis lorsque l'on suppose que les recouvrements observés entre les concepts polysémiques et monosémiques réalisent des transitions progressives. Ces observations ouvrent la perspective d'une modélisation graduelle du sens, complémentaire et compatible avec la vision structurelle habituellement envisagée dans le cadre de l'analyse de concepts formels.

3.2.1 Degré de polysémie des concepts dans un treillis

Nous proposons d'annoter ici les UL extraites du corpus (par une analyse en dépendances, *cf.* chapitre 2) par un ensemble de concepts formels qui possèdent ces UL dans leur intension ou leur extension. Nous examinons ici le nombre d'occurrences des concepts formels qui annotent les unités lexicales d'un corpus. La méthodologie que nous développons ici s'appuye sur les travaux menés par De Loupy [2002] sur l'évaluation des taux de synonymie et de polysémie dans un texte. Le graphique de la figure 3.4 représente la répartition du nombre d'occurrences des unités lexicales (objets et attributs) en fonction du nombre de concepts dans le treillis : les échelles sont logarithmiques en abscisses et en ordonnées.

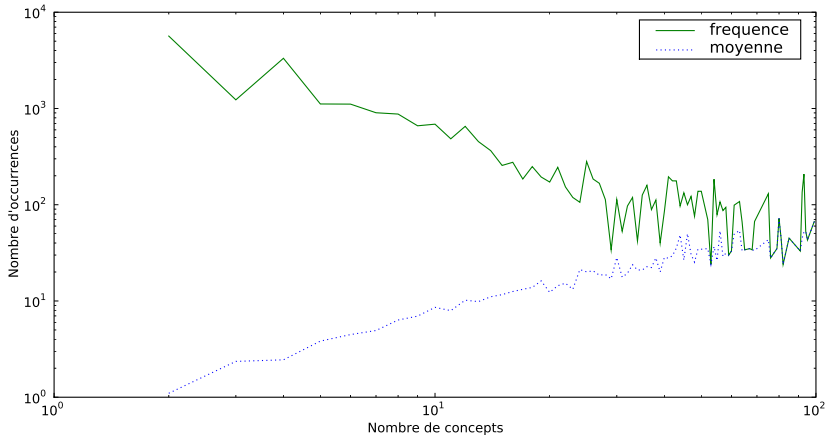


FIG. 3.4 – Lien entre le nombre de concepts et d'occurrences

La première courbe (ligne continue) indique le nombre d'occurrences des unités lexicales en fonction du nombre de concepts qui leur correspondent dans le treillis. On voit qu'elle décroît très rapidement. La seconde courbe (ligne discontinue) indique le nombre moyen d'occurrences des unités lexicales en fonction de leur nombre de concepts associés dans le treillis. La courbe croît avec le nombre de concepts. Ces tendances respectent le principe de Zipf [1945] selon lequel plus un terme est fréquent, plus il a tendance à être ambigu. Il est normal de constater qu'il n'existe pas d'unités lexicales ayant seulement un seul concept : chaque unité lexicale est présente dans $\top$ ou $\bot$ et dans un autre concept dans lequel elle est associée à d'autres éléments.

Le taux d'ambiguïté peut être estimé en rapportant la somme de tous les concepts possibles au nombre d'occurrences des attributs ou des objets dans le corpus. Pour le corpus utilisé et son treillis associé, le nombre moyen de concepts par unité lexicale est de 5,064. Cette valeur concorde avec les mesures effectuées par De Loupy [2002] sur WordNet. Ce taux de polysémie est lié au calcul du degré de monosémie d'une unité lexicale qui correspond à l'inverse du nombre de concepts du treillis qui contiennent cette unité lexicale. Ce degré de monosémie est décliné pour les objets et les attributs :

$$Monosemie(o) = \frac{1}{|\{c \in \mathfrak{T} | o \in \text{ext}(c)\}|} \tag{3.11}$$

$$Monosemie(a) = \frac{1}{|\{c \in \mathfrak{T} | a \in \text{int}(c)\}|} \tag{3.12}$$

Ces mesures prennent en compte le nombre de concepts du treillis qui contiennent un attribut ou un objet. Elles tendent vers 0 si l'unité lexicale considérée est polysémique et vers 1 si elle est monosémique. Nous regardons maintenant comment

ces mesures interviennent pour sélectionner des concepts peu polysémiques et sémantiquement homogènes, susceptibles de désambiguïser des usages d'unités lexicales.

3.2.2 Critère distributionnel pour l'extraction de concepts sémantiquement homogènes

Les règles d'attributs ayant un support élevé sont produites par des concepts généraux, contenant beaucoup d'objets et peu d'attributs alors que les concepts très spécifiques ayant un support élevé favorisent des règles d'objets avec beaucoup d'attributs en relation avec peu d'objets. La sélection menée par les mesures de confiance et de support ne s'accorde pas avec l'hypothèse distributionnelle selon laquelle les UL qui partagent un grand nombre de contextes tendent à avoir des sens proches. Cette hypothèse étant valable autant pour les objets que pour les attributs, une règle d'association ne devrait pas seulement être représentée par des implications d'objets ou d'attributs : pour sélectionner un concept formel sémantiquement homogène, il faut tenir compte des objets autant que des attributs. Par ailleurs, la sélection des concepts ne devrait par être dirigée sur des critères de fréquence ou de confiance appliqués uniquement sur les attributs ou les objets du contexte formel. Une sélection plus efficace pourrait être menée à la fois sur les intensions et les extensions d'un concept. Dans cette perspective, Maddouri [2005] propose une mesure de gain pour un concept $c = (E, I)$ qui se calcule de la manière suivante :

$$Gain((E, I)) = |E| \times |I| - (|E| + |I|) \tag{3.13}$$

Cette fonction de gain est maximisée seulement lorsqu'à la fois l'intension et l'extension ont une grande taille, c'est-à-dire que le concept contient beaucoup d'objets et d'attributs. Sa valeur est faible (voir négative) lorsque l'extension ou l'intension du concept possède moins de deux éléments. En maximisant leur nombre d'objets et leur nombre d'attributs, les concepts sélectionnés ne sont ni trop généraux, ni trop spécifiques.

Cependant, lorsque cette valeur de gain est maximisée, on retrouve des concepts contenant uniquement les objets ou les attributs les plus fréquents et les plus polysémiques. Pour éviter de retenir systématiquement ces concepts en priorité, nous employons une mesure de concentration [Durand et Crémilleux, 2002] qui régule le recouvrement entre concepts en limitant le cumul d'éléments entre les concepts. Cette mesure permet de sélectionner des concepts dont les éléments apparaissent rarement dans d'autre concepts. Pour Durand et Crémilleux [2002], la concentration d'un concept $c = (E, I)$ est définie comme la moyenne des taux

de monosémie des objets d'un concept $c = (E, I)$:

$$Concentration_o((E, I)) = \frac{1}{|E|} \sum_{o \in E} Monosemie(o) \tag{3.14}$$

Nous adaptons cette mesure pour le calcul de la moyenne des taux de monosémie d'un concept en considérant à la fois les objets et les attributs :

$$Concentration((E, I)) = \frac{1}{|E|} \sum_{o \in E} Monosemie(o) + \frac{1}{|I|} \sum_{a \in I} Monosemie(a) \tag{3.15}$$

La concentration d'un concept est élevée lorsque ses éléments tendent alors à être monosémiques. Dans le cas d'une concentration faible, les éléments du concept sont polysémiques. Une concentration faible tend vers 0 alors qu'une concentration élevée tend vers 1. Cette mesure permet de sélectionner en priorité les concepts qui possèdent des objets et attributs monosémiques.

À titre d'illustration, nous présentons une sélection des concepts concentrés qui ont un gain positif (*cf.* tableau 3.3). Pour faciliter la lisibilité, les intensions et extensions des concepts sont présentés sur plusieurs lignes. Le symbole $\rightarrow$ indique les règles d'implication existantes entre les éléments propres des éléments hérités dans ces concepts formels.

Concentration	ext(c)	int(c)
0.131	German, American	star, girl, men, request, diplomat
0.135	Iraq, U.N.	sanctions imposed, for, in, asked $\rightarrow$ trade sanctions, oil sales pact, balks
0.147	Iraqi, Arbil	control, city, attack, involvement $\rightarrow$ tank assault, were manned
0.147	Hutu, Kurdish, Moslem, PKK $\rightarrow$ Tamil Tiger	guerrillas, rebels
0.153	Istanbul, Belgrade, Sofia, Larnaca, London, Stansted	airport, landed
0.228	Berlin, Brussels	meeting, stadium, in $\rightarrow$ prix, organisers

TABLE 3.3 – Sélection de concepts concentrés ayant un gain positif

La quatrième ligne du tableau 3.3 indique par exemple que le concept $c =$ ({Hutu, Kurdish, Moslem, PKK [2], Tamil Tiger}, {guerrillas, rebels}) a une concentration de 0.147. Cette concentration relativement élevée indique que les objets et attributs de c apparaissent rarement dans d'autres concepts. Par ailleurs, c

2. Parti des travailleurs du Kurdistan.

génère la règle d'objets « l'entité nommée *Tamil Tiger* apparaît dans le même contexte que *Hutu, Kurdish, Moslem, PKK* ». Elle est vérifiée par les attributs *guerrillas* et *rebels*.

Les concepts ayant une faible concentration possèdent des attributs et des objets communs avec d'autres concepts et présentent généralement peu de règles d'implication. Les concepts du tableau 3.4 contiennent des éléments polysémiques qui appartiennent à de nombreux concepts et qui sont peu discriminants.

Concentration	ext(c)	int(c)
0.010	Russian, German, Israeli, Dutch	government, police, results, told
0.010	Britain, Iraq, U.S.	said, says, for, government
0.010	China, Baghdad, Iraq, India, Moscow	said, says, in

TABLE 3.4 – Sélection de concepts peu concentrés ayant un gain positif

La deuxième ligne du tableau 3.4 indique que le concept ({Britain, Iraq, U.S.}, { said, says, for, government}) possède une concentration de 0.01. Les attributs *said*, *says* et *told* sont polysémiques et peuvent être associés à tous les types d'entités nommées. Ils contribuent peu à la désambiguïsation des EN présentes dans les concepts et font baisser leur concentration. En revanche, l'observation globale de nos données semble indiquer qu'ils peuvent être combinés à des attributs plus discriminants pour constituer des concepts sémantiquement homogènes. Ces éléments peu discriminants et fréquents couvrent donc à la fois des zones polysémiques et monosémiques du treillis. Ainsi, les recouvrements observés entre les intensions et les extensions des concepts pourraient réaliser des transitions de sens progressives et continues.

La structure hiérarchique des treillis marque aussi cet aspect graduel du sens. Des concepts qui partagent des objets ou des attributs ont nécessairement des parents ou des enfants communs. Ils recouvrent des éléments communs si bien que le passage se fait progressivement entre des concepts partageant des parents ou des enfants dans le treillis. Ainsi, le treillis de la figure 3.2 présente des concepts pour lesquels on passe graduellement d'attributs du domaine de la politique (*government, minister*), à la sécurité civile (*police, city*), puis au domaine sportif (*city, league matches, league standings, coach*).

Ces transitions graduelles sont marquées par des recouvrements et différences plus ou moins forts entre les concepts. Nous allons à présent proposer de quantifier ces recouvrements et différences pour permettre de calculer une distance entre unités de sens et les modéliser dans un espace sémantique muni de propriétés de continuité.

3.2.3 Recouvrement et continuité des sens

Nous proposons ici d'examiner les aspects graduels du sens observés précédemment au regard d'un modèle de polysémie qui quantifie les différences entre des unités de sens associées à une unité lexicale. Cette quantification des différences serait utile pour discriminer le sens en contexte, autrement dit, permettre une désambiguïsation. Une distance pourrait permettre de caractériser ces différences et d'introduire la notion de proximité sémantique. Cette notion est développée dans le cadre du modèle proposé par Victorri et Fuchs [1996] qui représente le sens d'unités polysémiques dans un espace sémantique muni de propriétés de continuité. Jusqu'à présent, une représentation des concepts formels dans un espace sémantique continu n'a jamais été envisagée. Dans cette perspective, nous étudions ici l'applicabilité du modèle de Victorri et Fuchs [1996] au cadre des concepts formels.

3.2.3.1 Métrique distributionnelle pour la construction d'un espace sémantique continu

Dans le cadre théorique de la construction dynamique du sens [Victorri et Fuchs, 1996], des dictionnaires de synonymes sont utilisés pour représenter la structure sémantique d'unités polysémiques dans des espaces sémantiques continus. Le dictionnaire des synonymes est défini comme un graphe où les nœuds sont associés à des entrées lexicales, et deux nœuds sont connectés si leurs unités lexicales associées sont synonymes. Les cliques[3] du graphe sont considérées comme des unités de sens. Une mesure de similarité entre les cliques peut être déterminée en fonction du nombre de synonymes qu'elles partagent. On peut alors considérer la similarité entre cliques comme un critère de proximité sémantique entre des unités de sens.

L'espace sémantique est une projection en deux dimensions du nuage formé par les cliques dans l'espace multidimensionnel engendré par les synonymes d'une unité lexicale polysémique. À titre d'illustration, nous avons repris de [Jacquet, 2005] une représentation cartographique de l'espace sémantique associé à l'unité polysémique « monter » (*cf.* figure 3.5). Cette carte est une représentation continue des variations de sens d'une unité polysémique : deux cliques y sont proches lorsqu'elles partagent des synonymes. Cette représentation permet de rendre compte de l'aspect graduel des variations de sens entre les cliques.

3. Une clique est définie comme un sous-graphe complet maximal, c'est-à-dire un ensemble maximal de sommets tous reliés deux à deux.

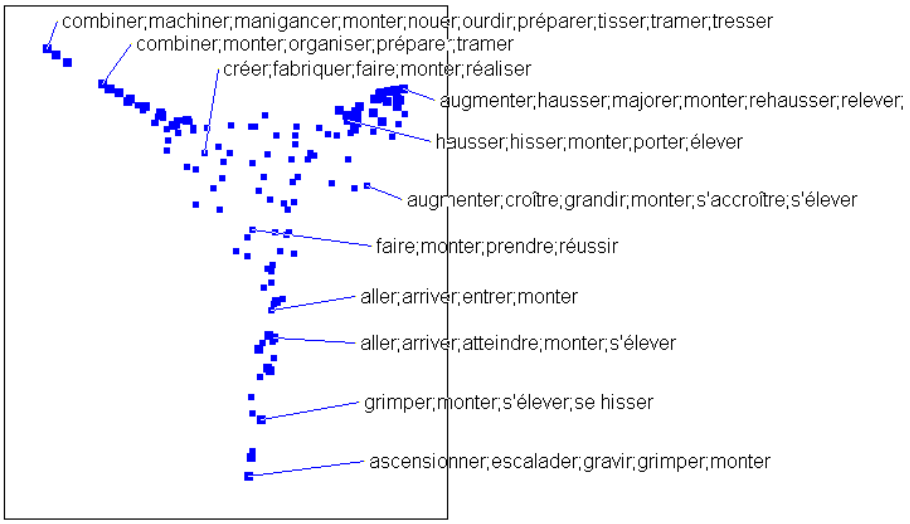


FIG. 3.5 – Cartographie de l'espace sémantique associé à l'unité polysémique « monter »

D'après Jacquet [2005], l'espace sémantique montré en figure 3.5 présente trois axes dominants qui correspondent à trois sens importants de « monter » et qui pourraient correspondre aux définitions d'un dictionnaire : « gravir, escalader une montagne » (cliques situées en bas sur la carte), « augmenter, relever les prix » (en haut à droite) et « organiser, préparer un projet » (en haut à gauche). Des changements progressifs du sens peuvent être observés entre les cliques situées aux extrémités des axes et celles au centre de la carte. Les nuances de sens observées font une transition progressive entre des sens distincts : par exemple, la transition entre « monter la montagne » et « monter à Paris » est matérialisée par les cliques qui s'étendent du bas vers le centre de la carte.

Victorri et Fuchs [1996] font intervenir le χ^2 pour calculer dynamiquement une représentation de l'espace sémantique associé à une unité polysémique. La mesure peut être considérée comme une mesure distributionnelle car elle indique si les différences entre deux cliques sont significatives en mesurant une dissimilarité entre leurs distributions de probabilité. Elle permet d'estimer le degré de dépendance statistique entre deux cliques c_k et c_l constituées d'unités polysémiques u_i. Plus formellement, soit u une unité lexicale associée à des synonymes $u_1, u_2 \dots u_n$ et à des cliques $c_1, c_2 \dots c_p$ Posons $x_{ki} = 1$ si $u_i \in c_k$ et $x_{ki} = 0$ si $u_i \notin c_k$. La mesure δ_{χ^2} est définie de la manière suivante :

$$\delta_{\chi^2}(c_k, c_l) = \sum_{i=1}^{n} \frac{x}{x_{.i}} \left(\frac{x_{ki}}{x_{k.}} - \frac{x_{li}}{x_{l.}}\right)^2 \tag{3.16}$$

$$x_{.i} = \sum_{j=1}^{p} x_{ji} \ , \ x_{k.} = \sum_{i=1}^{n} x_{ki} \text{ et } x = \sum_{i=1}^{n} \sum_{j=1}^{p} x_{ji} \tag{3.17}$$

Cette mesure fait intervenir deux caractéristiques importantes :

1. chaque unité lexicale est pondérée en fonction du nombre de cliques $x_{.j}$ dans lesquelles elle apparaît : le poids est d'autant plus faible que l'unité lexicale est présente dans un plus grand nombre de cliques. Une unité lexicale contenue dans un grand nombre de cliques joue un rôle moins important dans la discrimination du sens ;

2. le nombre d'éléments $x_{k.}$ d'une clique c_k intervient au dénominateur : les cliques composées de beaucoup d'unités lexicales vont être rapprochées les unes des autres et vont avoir tendance à occuper une position plus centrale dans l'espace sémantique.

Ces caractéristiques attachées aux χ^2 peuvent être rapprochées des propriétés déjà examinées dans cette section. Le calcul du degré de monosémie (*cf.* formules 3.11 et 3.12) et celui de $x_{.j}$ font intervenir le nombre d'unités de sens (respectivement des concepts et des cliques) contenant une unité lexicale. Par ailleurs, la mesure de gain (*cf.* formule 3.13 proposée par Maddouri [2005]), qui combine le nombre d'objets et d'attributs d'un concept, est assimilable au calcul du nombre d'éléments dans un clique $x_{k.}$. Ces propriétés nous encouragent à faire une analogie entre les concepts formels et les cliques.

3.2.3.2 Des cliques aux concepts formels

Jusqu'à présent, une représentation des concepts formels dans un espace sémantique continu n'a jamais été envisagée ; la notion de concept formel est pourtant très proche de celle de clique. Lorsque la relation binaire R d'un contexte formel $\mathbb{K} = (O, A, R)$ est représentée par un graphe biparti G, chaque concept formel correspond à une biclique maximale [4][Choi, 2006].

Dans la figure 3.6, le tableau (à gauche) définit le contexte formel (O, A, R) muni d'un ensemble d'objets $O = \{a, b, c, d\}$ et d'un ensemble d'attributs $A = \{1, 2, 3, 4\}$: les relations $(o, a) \in R$ sont indiquées par des « $\times$ ». Ce contexte formel peut être représenté par un graphe biparti (au milieu) dans lequel les bicliques maximales sont les concepts formels du treillis de Galois (à droite).

4. Un graphe est biparti s'il existe une partition de son ensemble de sommets en deux sous-ensembles U et V telle que chaque arête ait une extrémité dans U et l'autre dans V. Une biclique ou (graphe biparti complet) est une sorte de graphe biparti dans lequel chaque nœud d'un premier ensemble $X \subseteq U$ est connecté à chaque nœud du second ensemble $Y \subseteq V$. Cette biclique est dite maximale lorsqu'elle n'est contenue dans aucune autre biclique.

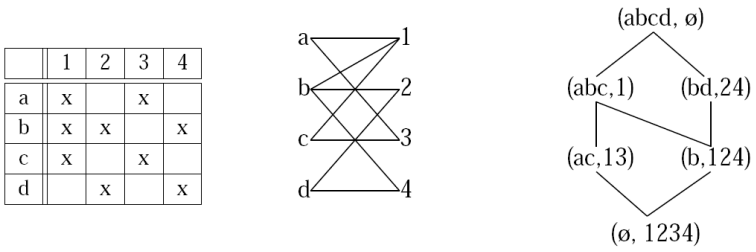


FIG. 3.6 – Équivalences entre un contexte formel, un graphe biparti et un treillis de Galois

Les concepts formels peuvent donc être manipulés comme des unités de sens au même titre que des cliques de synonymes. Cependant, ils sont constitués à partir d'usages et non à partir de ressources préétablies : nos concepts formels issus d'une extraction automatique introduisent certainement davantage de bruit que des cliques d'un dictionnaire des synonymes constitué manuellement. Il faut noter aussi que les concepts formels, en regroupant des relations entre des EN et leurs dépendances, conservent une composante syntagmatique que l'on ne retrouve pas dans des cliques de synonymes. Par ailleurs, les intensions des concepts sont constituées de groupes verbaux, groupes nominaux, prépositions n'entretenant pas de relations de synonymie contrairement aux cliques de Victorri et Fuchs [1996].

Dans la section suivante, nous étendons le modèle de la polysémie de Victorri et Fuchs [1996] dont la caractéristique essentielle est de proposer une représentation continue des variations de sens d'une unité polysémique. La représentation d'un espace sémantique de concepts formels pourrait nous aider à établir une modélisation de la polysémie à l'aide des treillis. Cette représentation permet de tenir compte de l'aspect graduel des différences entre concepts tout en se conjuguant avec les propriétés structurelles des treillis (*cf.* section 1 de ce chapitre). De plus, ce type de représentation réalise une quantification des différences entre concepts formels définissant alors une mesure de proximité sémantique potentiellement utile pour discriminer le sens en contexte.

3.3 Représentation géométrique de treillis de Galois

Les treillis de Galois sont généralement interprétés comme des structures symboliques dans lesquelles les concepts formels se différencient de manière qualitative. Dans cette section, nous complétons ce point de vue par une nouvelle interprétation géométrique des treillis en introduisant des distances entre concepts qui expriment une différenciation quantitative. À cet égard, nous reprenons le modèle cartographique de Victorri et Fuchs [1996] s'inscrivant dans un cadre

« continuiste » de la sémantique lexicale. Les cartes de ce modèle sont des espaces sémantiques continus dans lesquels chaque point correspond à une unité de sens (clique, concept formel). Un calcul de distance entre des points traduit une proximité sémantique plus ou moins grande entre des unités de sens. Cette notion de proximité sémantique est utile pour faire correspondre l'usage d'une unité lexicale avec un concept formel, ce qui pourrait être assimilable à une étape de désambiguïsation de cette unité. D'un point de vue mathématique, il s'agit de géométriser la topologie de l'espace des concepts formels structuré par un treillis.

Cette section développe les points suivants. Une représentation vectorielle des concepts formels est d'abord proposée. Nous présentons ensuite des techniques de réduction de la dimensionnalité pour visualiser cette représentation. Pour terminer, ces techniques sont utilisées pour projeter un ensemble de concepts formels comme les points d'une carte munie d'une métrique que nous interprétons comme un critère de proximité sémantique.

3.3.1 Représentation vectorielle des concepts formels

Nous associons ici chaque concept formel d'un ensemble C à un vecteur d'objets et un vecteur d'attributs. Ces vecteurs s'instancient dans deux espaces vectoriels dont les bases respectives sont les objets et les attributs des concepts de C. Ces espaces sont respectivement associés à une matrices des objets M_{O_C} et une matrice des attributs M_{A_C}.

Les vecteurs lignes de la matrice d'objets M_{O_C} correspondent aux concepts formels de C et ses colonnes aux objets $O_c = \bigcup_{x \in C} \text{ext}(x)$. Dans cette matrice binaire de dimension $(|C| \times |O_C|)$, chaque coefficient $m_{ij}^{O_C}$ indique (à la ligne i et la colonne j) la présence ou l'absence d'un objet o_j dans l'extension d'un concept formel c_i :

$$m_{ij}^{O_C} = \begin{cases} 1 & \text{si } o_j \in \text{ext}(c_i) \\ 0 & \text{sinon} \end{cases}$$

De manière analogue, on peut définir une matrice d'attributs M_{A_C} dont les lignes correspondent aux concepts formels de C et les colonnes aux attributs $A_c = \bigcup_{x \in C} \text{int}(x)$. Dans cette matrice binaire de dimension $(|A_C| \times |C|)$, chaque coefficient $m_{ij}^{A_C}$ indique (à la ligne i et la colonne j) la présence ou l'absence d'un attribut a_j dans l'extension d'un concept formel c_i :

$$m_{ij}^{A_C} = \begin{cases} 1 & \text{si } a_j \in \text{int}(c_i) \\ 0 & \text{sinon} \end{cases}$$

Pour éclaircir cette formalisation, examinons un ensemble de concepts formels

$C = \{c_0, c_1, c_2, c_3\}$ tel que :

$$\begin{array}{lll} c_0 = & (\{o_0, o_1, o_2, o_3\} & , \{a_0\}) \\ c_1 = & (\{o_0, o_1\} & , \{a_0, a_1\}) \\ c_2 = & (\{o_0, o_2, o_3\} & , \{a_0, a_2, a_3\}) \\ c_3 = & (\{o_0\} & , \{a_0, a_1, a_2, a_3\}) \end{array}$$

Chaque concept formel de C est représenté par un vecteur pour ses extensions et un vecteur pour ses intensions. Ces vecteurs sont exprimés dans deux matrices M_{O_C} et M_{A_C}, appelées respectivement matrice des extensions et matrice des intensions, dans lesquelles chaque ligne correspond à un concept formel de C. Les colonnes de M_{A_C} correspondent aux intensions des concepts et de manière similaire, les colonnes de la matrice des extensions M_{O_C} sont assignées aux extensions des concepts.

$$M_{O_C} = \begin{array}{c} \left[\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{array}\right] \begin{array}{c} c_0 \\ c_1 \\ c_2 \\ c_3 \end{array} \\ o_0 \ o_1 \ o_2 \ o_3 \end{array} \tag{3.18}$$

$$M_{A_C} = \begin{array}{c} \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{array}\right] \begin{array}{c} c_0 \\ c_1 \\ c_2 \\ c_3 \end{array} \\ a_0 \ a_1 \ a_2 \ a_3 \end{array} \tag{3.19}$$

Nous pouvons aussi considérer la matrice des concepts M_C qui place les attributs et les objets des concepts comme composantes des vecteurs associés aux concepts formels : la matrice M_C est une fusion des matrices M_{O_C} et M_{A_C}.

$$M_C = \begin{array}{c} \left[\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right] \begin{array}{c} c_0 \\ c_1 \\ c_2 \\ c_3 \end{array} \\ o_0 \ o_1 \ o_2 \ o_3 \ a_0 \ a_1 \ a_2 \ a_3 \end{array} \tag{3.20}$$

De manière à prendre en compte le nombre de cooccurrences entre les objets et les attributs qui pourraient apparaître dans un corpus, nous considérons le contexte formel pondéré $\tilde{\mathbb{K}} = (O, A, \tilde{R})$ dans lequel les éléments de $\tilde{R}$ sont des relations associées à des réels positifs ou nuls : plus formellement, $\tilde{R} : (O \times A) \rightarrow$

$\mathbb{R}^+$. Les coefficients $m_{ij}^{O_C}$ et $m_{ij}^{A_C}$ des matrices M_{O_C} et M_{A_C} peuvent ensuite être définis de la manière suivante :

$$m_{ij}^{O_C} = \sum_{a \in \text{int}(c_i)} \tilde{R}(o_j, a) \tag{3.21}$$

$$m_{ij}^{A_C} = \sum_{o \in \text{ext}(c_i)} \tilde{R}(o, a_j) \tag{3.22}$$

Pour un concept c_i, décrit en ligne i, le coefficient $m_{ij}^{O_C}$ (à la ligne i et à la colonne j) correspond à la somme du nombre de cooccurrences entre un objet $o_j \in \text{ext}(c_i)$, associé à la colonne j, et chaque attribut $a \in \text{int}(c_j)$ dans l'intension du concept c_j. De manière analogue, le poids $m_{ij}^{A_C}$ (à la ligne i et à la colonne j) correspond à la somme du nombre de cooccurrences entre un attribut $a_j \in \text{int}(c_i)$ (représenté en colonne j) et chaque objet $o \in \text{ext}(c_i)$ dans l'extension du concept c_i. Pour nos données, notons que les coefficients $m_{ij}^{O_C}$ et $m_{ij}^{A_C}$ des concepts $\top$ et $\bot$ sont nuls car $\text{ext}(\bot) = \emptyset$ et $\text{int}(\top) = \emptyset$.

Bělohlávek [2000] a déjà proposé une représentation matricielle analogue pour mesurer la similarité entre concepts formels. La notion de similarité apparaît également dans le cadre de l'analyse de concepts formels par similarité [Messai, 2009; Messai *et al.*, 2010] qui permet de former des concepts en regroupant des objets autour d'attributs ayant des valeurs similaires.

La suite de nos travaux s'appuient sur notre représentation matricielle pour établir la notion de similarité entre concepts formels. Cette similarité pourrait tout à fait être déterminée à l'aide de la distance euclidienne, du cosinus ou encore de la mesure du χ^2 dont nous avons déjà montré l'intérêt dans ce chapitre. Notre représentation matricielle est maintenant exploitée à l'aide de techniques de réduction de dimensionnalité. Ces techniques nous aident à interpréter les concepts formels dans des espaces de faible dimension.

3.3.2 Projection des concepts formels vers un espace réduit

Lorsque le volume de données à traiter est important, l'ensemble des règles d'association ou celui des concepts formels extrait peut être difficile à interpréter ou à manipuler. Plusieurs types de travaux sont partis de ce constat pour synthétiser un ensemble de règles d'association ou de concepts formels en une représentation compressée à l'aide de méthodes de *clustering* statistique. L'intérêt est de conserver l'expressivité des règles d'association ou des concepts en tirant avantage d'une représentation synthétique fournie par ces méthodes. Pour appliquer ces méthodes statistiques, la plupart de ces approches s'appuient sur des distances ou des mesures de similarité entre des règles d'association ou des concepts formels. Certaines approches se restreignent à regrouper les règles en fonction de la similarité de leur condition ou de leur conséquence (partie gauche ou droite d'une

règle) : les règles sont alors constituées uniquement d'attributs. Lent *et al.* [1997] formalisent le problème du *clustering* de règles d'association munies de deux attributs pouvant prendre des valeurs continues. Ce travail a été généralisé pour traiter des règles comportant un nombre quelconque d'attributs [Gupta *et al.*, 1999]. Han *et al.* [1997] proposent un partitionnement d'un ensemble de règles d'association structuré par un hypergraphe. Dans cette structure, les attributs (ou *items*) correspondent à des nœuds connectés par hyper-arêtes lorsque ces attributs interviennent dans une même règle. Dans le cadre de l'analyse de concepts formels, Pensa [2006] réalise des regroupement de concepts formels similaires en utilisant un algorithme de classification ascendante hiérarchique. L'auteur définit une distance entre concepts formels (bi-ensembles) qui est calculé avec une mesure de différence symétrique ensembliste (voir formule 4.1 dans le chapitre suivant). Contrairement aux méthodes dédiées aux règles d'association, cette mesure est appliquée à la fois sur les extensions et les intensions des concepts.

Nous employons ici des techniques de réduction de la dimensionnalité qui nous servent à construire et visualiser un espace de concepts formels associés à une unité polysémique. Les concepts formels (ou du moins, leurs extensions et intensions) sont désormais associés à une représentation vectorielle multidimensionnelle. La réduction de dimensionnalité a pour objectif de créer un petit nombre de variables qui décrivent des données aussi bien que le font les variables « brutes » décrivant ces données, habituellement en grand nombre. Un processus réduisant la dimensionnalité d'une base de données agit comme un compresseur de l'espace des variables. Ce processus non supervisé fait en général perdre de l'information. Ces techniques ont cependant plusieurs avantages. Elles réduisent la quantité d'information que les algorithmes auront à traiter, diminuant ainsi, parfois fortement, les temps de calcul et l'encombrement mémoire. Par ailleurs, en limitant le nombre de variables à 2 ou 3, elles fournissent des coordonnées pour une représentation visuelle plane ou tridimensionnelle. Enfin, elles généralisent les données observées en proposant un modèle (prédictif ou descriptif). Parmi les nombreuses techniques existantes, nous en avons retenu deux [5].

La principale méthode de réduction de dimensionnalité choisie est l'analyse en composantes curvilignes (ACC ou CCA : *curvilinear component analysis*) [Hérault *et al.*, 1999] qui s'appuie sur des réseaux de neurones non supervisés analogues aux cartes auto-organisatrices de Kohonen [1988]. La CCA cherche une représentation des données dans un espace de faible dimension qui garde au mieux

5. Trois librairies Python *open source* on été utilisées pour nos expérimentations : *scikits.ManifoldLearning* [Brucher *et al.*, 2008] pour l'analyse en composantes curvilignes `http://www.scipy.org/scipy/scikits/wiki/MachineLearning/ManifoldLearning` ; Orange [Demsar *et al.*, 2004] pour l'analyse factorielle des correspondances (`http://www.ailab.si/orange/doc/modules/orngCA.htm`) ; *modular data processing toolkit* [Zito *et al.*, 2009] notamment pour son implémentation du *Growing Neural Gas Network* [Fritzke, 1995].

les distances entre observations. L'algorithme effectue une projection non linéaire des observations (vecteurs d'entrée) dans un espace de faible dimension p réglée par l'utilisateur. La projection étant le plus souvent effectuée dans un espace de dimension réduite, il est impossible de faire correspondre exactement les points projetés et les observations. La CCA essaye de respecter au moins localement la topologie de l'entrée en favorisant la qualité de projection des observations séparées par des courtes distances plutôt que de conserver les grandes.

À titre de comparaison, nous avons utilisé l'analyse factorielle des correspondances qui est une méthode statistique conçue pour l'analyse des données textuelles [Benzécri, 1973, 1981]. Cette méthode a été reprise par [Victorri et Fuchs, 1996; Jacquet *et al.*, 2005] pour la cartographie des synonymes en mettant en lumière les propriétés du χ^2 pour mesurer la proximité sémantique d'unités se sens.

Nous présentons maintenant des cartes sémantiques de concepts formels produites à l'aide de ces deux techniques de réduction.

3.3.3 Cartographie sémantique des concepts formels

Nous avons restreint cette étude à l'ensemble des concepts formels C contenant l'entité nommée *Iranian* lorsqu'elle est en relation avec *troops*. Cet ensemble est associé à la matrice des concepts M_C munie de coefficients pondérés. Les individus (lignes) de cette matrice sont des concepts formels, et ses variables (colonnes) correspondent à l'ensemble des attributs et des objets contenus dans les intensions et les extensions des concepts. Une projection en deux dimensions par analyse en composantes curvilignes de M_C est représentée en figure 3.7.

L'ordonnancement général/spécifique des concepts est globalement respecté. Ici le concept $\top$ est situé dans la partie supérieure de la carte alors qu'à l'opposé, le concepts $\bot$ est placé dans la zone inférieure. Les concepts intermédiaires sont également répartis du plus général au plus spécifique. Cette projection conserve donc globalement la topologie du treillis s'appuyant sur la relation d'inclusion ensembliste. Nous avons effectué d'autres essais de projection qui ont aussi montré que cette structuration hiérarchique du treillis est respectée (voir par exemple, la figure 3.2 présentée en début du chapitre pour l'entité nommée *Australian* en relation avec l'unité lexicale *results*. Des concepts partageant des parents/enfants possèdent des objets/attributs communs et tendent à être séparés par de faibles distances et être proches sur la carte. La proximité des points de la carte peut donc être interprétée comme un signe de similarité sémantique entre concepts formels. Par ailleurs, le recouvrement des intensions/extensions se traduit par des propriétés de continuité sur la carte.

À titre de comparaison, nous avons cartographié les concepts du même ensemble C en se limitant cette fois aux objets (matrice des extensions M_{O_C}) ou

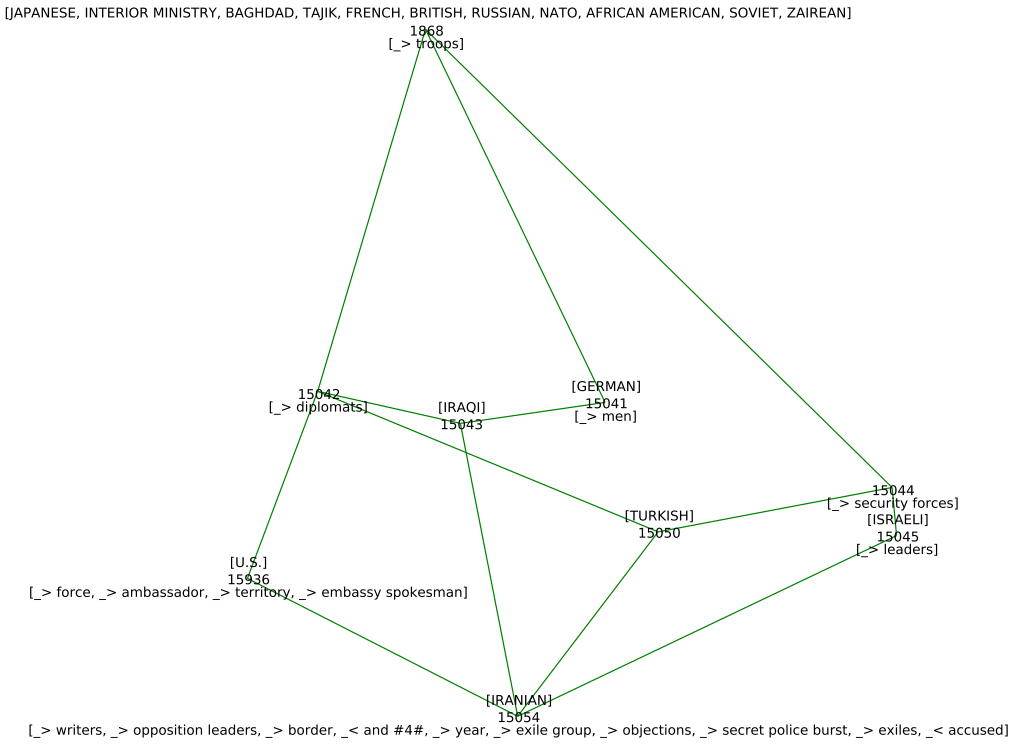


FIG. 3.7 – Projection 2D de la matrice M_C munie de coefficients pondérés avec une analyse en composantes curvilignes

aux attributs (matrice des intensions M_{A_C}) pour la projection des concepts. Ces projections, en figures 3.8 et 3.9, ne sont visiblement pas d'aussi bonne qualité que lorsque tous les objets et attributs de C sont pris en compte. On peut toutefois remarquer sur cet exemple que l'exploitation des attributs avec M_{A_C} est la seule à respecter globalement l'organisation hiérarchique du treillis.

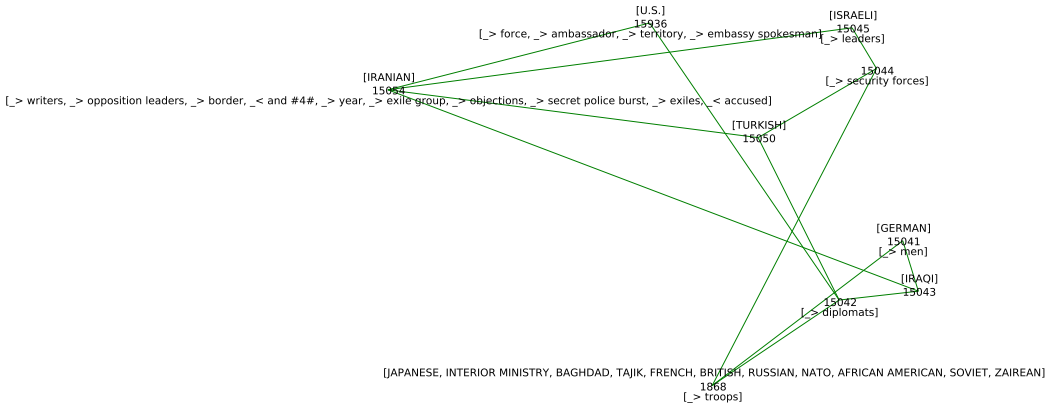


FIG. 3.8 – Exemples de projections avec la matrice des extensions M_{O_C}

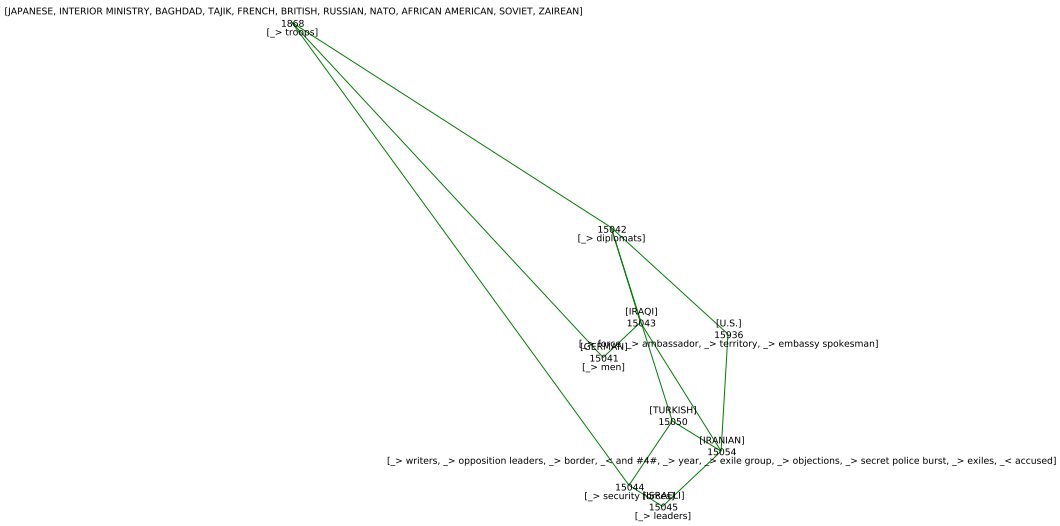


FIG. 3.9 – Exemples de projections avec la matrice des intensions M_{A_C}

À titre de comparaison, la figure 3.10 montre une projection 2D des concepts de M_C avec une analyse factorielle des correspondances.

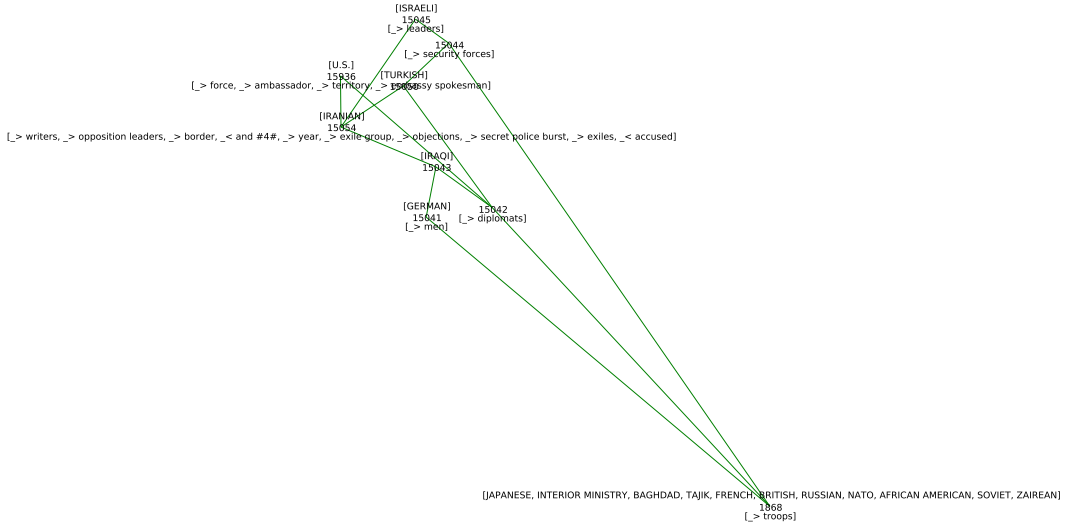


FIG. 3.10 – Projection 2D de la matrice M_c munie de coefficients pondérés avec une analyse factorielle des correspondances

Proportionnellement, les distances sur la figure 3.10 ne sont pas identiques à celles de la figure 3.7. Les deux techniques employées ont un comportement comparable pour les longues distances mais le concept le plus général c_{1868} est encore plus éloigné des autres concepts avec l'analyse factorielle des correspondances. Sur notre exemple, l'analyse factorielle des correspondances semble moins à l'aise que la CCA pour respecter la topologie locale de zones où les concepts sont très rapprochés. L'analyse factorielle paraît être moins appropriée que l'analyse en composantes curvilignes pour visualiser les concepts de C. Nous ne pouvons cependant pas généraliser et nous pensons que d'autres essais devraient être menés avec d'autres données pour privilégier la technique la plus adaptée.

3.3.4 Discussion

L'utilisation d'un modèle vectoriel des concepts ouvre la perspective de coupler notre approche symbolique de la sémantique, fondée sur l'analyse de concepts formels, avec différents outils d'analyse statistique. Il devient notamment possible d'utiliser des outils de *clustering* statistique pour regrouper des concepts formels similaires et synthétiser le contenu d'un treillis volumineux devenu difficile à manipuler ou à interpréter.

Le couplage proposé entre les treillis et les outils de compression statistique met en œuvre une représentation hybride symbolique-numérique. Cette hybridation pourrait en plus reprendre la structure d'un treillis pour construire le réseau

de neurones utilisé en CCA, alors que ce réseau est habituellement constitué en connectant un élément représenté dans l'espace $\mathbb{R}^n$ avec ses k plus proches voisins.

Dans une autre perspective, cet outil de cartographie pourrait aider à visualiser l'évolution de treillis de Galois pendant l'intégration de nouvelles données observées dans un flux. Pour chaque évolution d'un treillis, son état serait matérialisé par un ensemble de points sur une carte. La dynamique de structuration d'un treillis pourrait ainsi être étudiée en suivant l'évolution temporelle de sa cartographie. À cet égard, Rastier [2000] suggère de considérer des unités de sens en évolution comme un système dynamique : *« Si en synchronie les relations au sein des classes lexicales peuvent être caractérisées par des oppositions sémiques discrètes, le caractère graduel des évolutions diachroniques peut être représenté par des modèles dynamiques qui, sans contredire l'analyse sémique, caractérisent les sémèmes comme des points remarquables sur des dynamiques. »*. Ce point de vue nous conforte dans l'idée d'une représentation hybride symbolique/numérique dans laquelle des unités de sens discrètes (concepts formels), caractérisées pour nous par leur contexte d'usage (assimilables à des sèmes), s'instancie dans un espace sémantique continu en évolution.

3.4 Conclusion

Dans ce troisième chapitre, nous avons montré que le treillis construit dynamiquement à partir de nos données présentait des propriétés sémantiques remarquables que nous proposons d'exploiter pour la modélisation de la polysémie lexicale. Notre démarche a été dirigée par deux hypothèses que nous avons examinées par l'analyse de données issues de notre corpus. À cet égard, nous avons cherché à établir des critères pour déterminer dans quelles conditions les concepts formels pouvaient être assimilés à des unités de sens. Par ailleurs, nous avons regardé comment un treillis de Galois organise la différenciation de ces unités de sens pour modéliser la polysémie lexicale.

Les travaux de Toussaint *et al.* [2000] ont été particulièrement éclairants pour comprendre les différences de granularité entre les concepts du treillis. Nous avons alors pu faire apparaître des règles d'implication formalisables par des règles d'association entre des objets (EN) ou entre des attributs. En mesurant le degré de fiabilité de ces règles, nous avons pu identifier des regroupements d'objets ou d'attributs apparaissant fréquemment dans le même contexte. Nous avons observé que tous ces regroupements ne constituaient pas pour autant des unités de sens sémantiquement homogènes.

Nous avons alors cherché à mettre en correspondance les usages d'un corpus avec les concepts du treillis. Cette première tentative nous a amené à élaborer une mesure du degré de polysémie des objets, des attributs et des concepts. Nous avons

remarqué que certains concepts monosémiques possédaient suffisamment d'objets et d'attributs pour constituer des extensions et des intensions sémantiquement homogènes. À cet égard, ils vérifient l'hypothèse distributionnelle selon laquelle le partage de contextes est un signe de proximité sémantique. Nous avons également observé que la répartition des objets et des attributs entre les concepts pouvait être interprétée comme une transition graduelle entre les unités de sens. Ces observations mettent l'accent sur des propriétés de continuité complémentaires de la structuration des treillis habituellement envisagée sous l'angle de modèles discrets et symboliques en analyse de concepts formels.

Nous avons ainsi été conduit à proposer une représentation originale de la répartition des concepts formels par l'expression de leurs objets et attributs sous une forme vectorielle. Les concepts peuvent alors être représentés dans un espace sémantique muni de propriétés de continuité, nous permettant d'estimer la proximité sémantique entre des unités de sens. De plus, de nombreuses méthodes statistiques non supervisées (méthodes de *clustering* ou de réduction de la dimensionnalité) deviennent applicables aux concepts pour proposer de nouveaux niveaux d'interprétation aux treillis.

Les représentations structurées et continues présentées dans ce chapitre sont capables de différencier le sens d'UL par des calculs différents. Dans le modèle de structuration des concepts, les différences entre les concepts appartenant à des niveaux distincts sont exprimables par des règles d'implication entre les unités lexicales. Dans le modèle formalisé par une représentation vectorielle des concepts, les variations de sens sont abordées comme un phénomène relevant du continu et les différences entre les concepts sont cette fois exprimées par des distances.

Dans un objectif de désambiguïsation lexicale, nous serons amené dans le prochain chapitre à combiner les représentations structurées et continues pour annoter les EN d'un corpus par une sélection de concepts qui sont représentatifs des usages de ces EN. Cette annotation réalisée de manière non supervisée est considérée comme un enrichissement de corpus. Nous évaluons alors la pertinence de cette annotation dans le cadre d'une tâche de classification supervisée d'EN.

Chapitre 4

Exploitation de treillis de Galois en désambiguïsation non supervisée

Dans les chapitres précédents, nous avons présenté un modèle de base lexicale hiérarchisée par un treillis de Galois constituée automatiquement à partir de corpus. À l'instar des méthodes de classification non supervisées habituellement utilisées en analyse distributionnelle, l'analyse de concepts formels permet de constituer des regroupements d'unités lexicales correspondant à des usages observés sur corpus. L'intérêt, nous semble-t-il, de notre approche est d'apporter un niveau de structuration supplémentaire adapté à la modélisation de la polysémie lexicale. Cette structuration développe les sens sur plusieurs niveaux conceptuels : le passage d'un niveau à un autre indique un changement de granularité. Les différences observées entre les niveaux mettent en évidence des règles d'implication. Celles-ci présentent un degré de généralisation (ou de spécialisation) pertinent pour la description et la désambiguïsation des unités lexicales. Par ailleurs, les treillis sont des structures de données suffisamment flexibles pour permettre l'intégration d'UL identifiées dans un nouveau corpus. Cette propriété est particulièrement intéressante pour des systèmes de désambiguïsation sémantique ou de reconnaissance d'EN qui doivent traiter des mots inconnus. Toutefois, il faut noter que chaque unité polysémique pouvant être associée à plusieurs concepts formels, sa désambiguïsation n'est pas immédiate. C'est pourquoi nous proposons ici l'élaboration d'une stratégie de sélection des concepts appropriés à la désambiguïsation d'EN en contexte.

Dans ce quatrième chapitre, nous présentons une nouvelle méthode d'annotation automatique des EN d'un corpus de test. Cette méthode exploite des concepts formels inférés par ACF de manière non supervisée sur un corpus d'apprentissage. Cette « annotation conceptuelle » est considérée comme un enrichissement de corpus exploitable dans le cadre d'une tâche de classification supervisée d'EN. Le plan de ce chapitre est le suivant. La première section examine des méthodes de

désambiguïsation d'unités lexicales qui mettent à profit des ressources constituées manuellement : elles ont servi de point de départ à l'élaboration de notre méthode d'annotation, méthode qui s'appuie sur une exploration des données dans un treillis. La deuxième section présente deux stratégies d'annotation conceptuelle des unités lexicales d'un corpus mettant à profit les concepts formels apparaissant dans un treillis. Une première stratégie recourt à des règles d'association qui permettent de choisir un couple de concepts annotant à la fois une EN et un de ses attributs en relation de dépendance. La seconde permet de prendre en compte plusieurs attributs en dépendance avec une entité nommée en déterminant un élément représentatif d'un ensemble de concepts candidats pour l'annotation de cette EN. La dernière section est consacrée à la validation expérimentale de notre seconde stratégie d'annotation conceptuelle dans le cadre d'une tâche de classification supervisée d'EN dont on sait mesurer les performances. Dans cette perspective, nous employons un protocole expérimental d'évaluation en cascade [Candillier *et al.*, 2006] qui se déroule en deux étapes : la première repose sur l'enrichissement d'un corpus en exploitant les résultats d'une annotation conceptuelle ; la seconde consiste à vérifier si un algorithme d'apprentissage supervisé bénéficie de cet enrichissement pour une tâche de classification en comparant ses résultats avec ceux d'un apprentissage et d'une classification réalisés sur des données non enrichies. Pour terminer, les résultats de cette évaluation en cascade sont analysés et discutés.

4.1 La désambiguïsation vue comme une sélection d'unités de sens prédéfinies

Certaines méthodes de désambiguïsation sémantique exploitent des ressources prédéfinies (dictionnaires, thésaurus...) pour étiqueter les unités lexicales d'un corpus. Cette tâche peut être dirigée par des critères de similarité entre des unités de sens prédéfinies (définitions, concepts) et le contexte d'usage des unités polysémiques.

Cette section bibliographique se penche sur plusieurs approches s'appuyant sur différentes représentations du sens. Pour chacune de ces méthodes, un rapprochement avec des travaux existants en ACF est proposé. En premier lieu, nous étudions l'algorithme de Lesk [1986] qui permet de sélectionner les définitions d'un dictionnaire pour désambiguïser l'ensemble des mots d'un énoncé. Nous examinons ensuite les travaux de Resnik [1999] portant sur une désambiguïsation utilisant à la fois la taxonomie de WordNet [Miller, 1995] et une analyse fréquentielle des occurrences d'un corpus. À l'issue de cette section, nous discutons des adaptations possibles de ces différentes méthodes à notre problème d'annotation conceptuelle d'entités nommées.

4.1.1 Désambiguïsation par définitions de dictionnaire

À l'aide d'un dictionnaire électronique, Lesk [1986] propose un algorithme de désambiguïsation qui applique l'hypothèse que des mots situés dans un même voisinage tendent à partager un sujet commun. Le principe de l'algorithme de Lesk repose sur la mesure du recouvrement entre les différentes définitions, dans le dictionnaire, d'un mot ambigu et les définitions de ses voisins immédiats, situés dans une fenêtre de mots de taille fixée (en général une dizaine de mots). La procédure employée (*cf.* algorithme 6) s'appuie sur la désambiguïsation mutuelle des unités polysémiques. L'intérêt de cet algorithme dans notre cadre serait d'en utiliser une variante non supervisée qui serait dirigée, non pas sur un dictionnaire traditionnel, mais par les résultats d'une analyse de concepts formels.

Algorithme 6: Algorithme de Lesk

Entrées : Dictionnaire, mots d'un énoncé
Sorties : mots de l'énoncé annotés par des définitions

```
pour chaque définition s_ti d'un mot w_t faire
    score_i = 0
    pour chaque mot w_j ∈ voisinage(w_t) faire
        si j == t alors break
        pour chaque définition s_jk de w_j faire
            temp_score[i] = sim(s_ti, s_jk)
        meilleur_score = max(temp_score)
        si meilleur_score > θ alors
            score_i = score_i + meilleur_score
    retourner i, tel que score_i ≥ score_j, ∀j, 1 ≤ j ≤ n avec n, le nombre
    de mots dans une phrase
```

La mesure de recouvrement utilisée dans l'application de l'algorithme de Lesk peut se traduire par une distance entre les définitions s_{ti} d'un mot w_t à désambiguïser et les définitions s_{jk} de chaque mot w_j du contexte d'usage $voisinage(w_t)$ de w_t. Plus le recouvrement entre deux définitions est grand, plus la distance qui les sépare diminue.

Dans un treillis de Galois, les descriptions des concepts formels — en intension et en extension — pourraient être assimilées aux définitions d'un dictionnaire. Pour évaluer la distance entre deux concepts formels, Pensa [2006] propose une mesure de dissimilarité manipulant des opérations ensemblistes (à savoir l'union et la différence symétrique[1]). Pour deux concepts formels $c_1 = (E_1, I_1)$ et $c_2 =$

1. La différence symétrique entre deux ensembles A et B est définie par $A\Delta B = (A \cup B) - (A \cap B)$.

(E_2, I_2), la distance δ_Δ entre c_1 et c_2 est définie par :

$$\delta_\Delta(c_1, c_2) = \frac{1}{2}\left(\frac{|E_1 \Delta E_2|}{|E_1 \cup E_2|} + \frac{|I_1 \Delta I_2|}{|I_1 \cup I_2|}\right) \tag{4.1}$$

On peut noter que le calcul de δ_Δ considère les extensions autant que les intensions pour mesurer la distance entre deux concepts. Une telle mesure serait opportune dans l'algorithme de Lesk afin de sélectionner des concepts formels. Cependant, δ_Δ ne permet pas de tenir compte des définitions hiérarchisées telles qu'elles peuvent l'être dans un thésaurus ou un treillis de Galois. Par ailleurs, cette distance n'est pas capable de prendre en compte le nombre d'occurrences des unités lexicales en corpus. Pour résoudre ces problèmes, nous examinons des mesures de similarité qui ont été élaborées pour prendre en compte une structure taxonomique.

4.1.2 Mesures de similarité taxonomique

En partant de l'algorithme de Lesk, Patwardhan *et al.* [2003] et Vasilescu *et al.* [2004] présentent une technique de désambiguïsation exploitant des mesures de similarité capables d'évaluer la proximité sémantique entre deux nœuds d'une base lexicale taxonomique telle que WordNet [Miller, 1995]. Ce type de mesure permet de mettre en correspondance des UL observées en corpus avec les entrées lexicales décrites dans la base lexicale. Le calcul de similarité repose principalement sur la longueur du plus court chemin entre deux concepts dans la hiérarchie. L'intuition sous-jacente est de déterminer le plus direct hyperonyme commun à deux concepts dans la taxonomie. Ce concept est le subsumant le plus spécifique (SPS, ou en anglais LCS pour *least common subsumer*) : le SPS d'une collection de concepts $c_1 \ldots c_n$ est le concept le plus spécifique D qui subsume $c_1 \ldots c_n$ tel que $c_i \leq d$ pour $i = 1 \ldots n$; et si $c_i \leq E$ pour $i = 1 \ldots n$ alors $D \leq E$.

Une mesure de similarité conceptuelle hybride, proposée par Resnik [1999], combine des contraintes structurelles avec des critères probabilistes estimés par une analyse des occurrences dans un corpus. Lorsque plusieurs unités polysémiques cooccurrent, le sens le plus probable pour chacune de ces unités lexicales est celui qui maximise les relations sémantiques entre chaque sens choisi : le concept le plus probable pour annoter une unité polysémique est celui qui minimise la longueur du chemin avec les autres unités du concept.

L'approche proposée par Resnik ne se limite pas à déterminer le SPS entre deux concepts, c'est-à-dire le plus court chemin qui les sépare. La distance calculée prend aussi en compte la distribution des unités lexicales observées dans un corpus de référence. Cette connaissance n'est pas disponible dans un corpus ne disposant pas d'un étiquetage sémantique : il n'existe pas d'association entre les mots du corpus et les concepts de la base lexicale. Resnik suggère alors de compter le

nombre d'occurrences d'une unité polysémique dans le corpus, puis de diviser ce compte par le nombre de concepts différents qui lui sont associés dans la base lexicale. Par exemple, supposons que l'unité lexicale *bank* apparaisse 20 fois dans un corpus. Les concepts associés à cette unité dans la taxonomie (en l'occurrence, *river bank* et *financial bank*) reçoivent chacun un compte de 10, ce qui présuppose une équiprobabilité d'appartenance des sens possibles.

La similarité entre deux concepts c_1 et c_2 est finalement définie comme l'entropie (quantité d'information) fournie par les lexicalisations (les unités lexicales du concept) du SPS dans un corpus, où $P(c)$ est la probabilité de retrouver une instance du concept c.

$$sim(c_1, c_2) = -\log P(SPS(c_1, c_2))$$

Ces travaux ont été adaptés à la structure de treillis de Galois pour permettre de mesurer la similarité entre des concepts formels. Dans le cadre d'une logique de descriptions, Maala *et al.* [2007] présentent une mesure de similarité entre deux concepts formels qui s'appuie sur la longueur des chemins entre ces concepts dans le graphe que constitue le treillis. Cette mesure fait appel aux notions de concept subsumant le plus spécifique (SPS) et de concept subsumé le plus général (SPG, en anglais MGS pour *most general subsumee*) pour une collection de concepts. Le SPG d'une collection de concepts $c_1 \dots c_n$ est le concept le plus général D qui est subsumé par $c_1 \dots c_n$ tel que $D \leq c_i$ pour $i = 1 \dots n$; et si $E \leq c_i$ pour $i = 1 \dots n$ alors $E \leq D$.

La distance est calculée en comptant le nombre d'arcs entre les concepts. Cette mesure peut être considérée comme une distance d'édition car elle est calculée sur la base d'un chemin minimal de transformations entre éléments (objets et/ou attributs). Ce type de similarité taxonomique est capable de tirer parti de l'aspect structurel des treillis. En revanche, contrairement au modèle de Resnik [1999], l'aspect fréquentiel (fréquences d'occurrence d'UL en corpus) n'est pas utilisé pour pondérer les arcs du treillis. Dans notre cadre d'exploitation de treillis en désambiguïsation lexicale, il nous semble plutôt approprié de combiner ces aspects structurels et fréquentiels : nous n'utiliserons donc pas la mesure proposée par Maala *et al.* [2007].

4.1.3 Conclusion

Les approches examinées dans cette section mettent en œuvre des techniques de désambiguïsation d'unités polysémiques à partir de ressources décrivant une sémantique prédéfinie du lexique. Dans le cadre de l'utilisation de concepts formels pour cette tâche, une variante non supervisée de l'algorithme de Lesk pourrait s'appuyer, non pas sur un dictionnaire traditionnel, mais sur un treillis de Galois.

Plutôt que de mesurer le taux de recouvrement entre des définitions, les critères de sélection d'unités de sens pourraient être dirigés par une mesure de similarité ou une distance entre des concepts formels [Maala *et al.*, 2007]. Dans cette perspective, des mesures de similarité taxonomiques pourraient tirer avantage de l'aspect structurel des treillis de Galois autant que des aspects fréquentiels observés dans des corpus. Enfin, en complément à cet aspect structurel de la polysémie, il paraît opportun d'exploiter le modèle continu du sens présenté dans le chapitre 3.

4.2 Annotation conceptuelle d'unités lexicales

Certaines méthodes que nous venons d'examiner sont capables d'exploiter des ressources sémantiques prédéfinies pour désambiguïser les unités lexicales d'un corpus. L'objectif de ces méthodes est d'associer des unités lexicales observées en corpus avec des unités de sens d'une ressource. Dans la plupart des cas, cette mise en correspondance est déterminée par un score indiquant la similarité entre les descriptions des entrées lexicales et le contexte d'usage des unités lexicales dans le corpus.

Dans cette section, nous proposons d'adapter les principes de ces méthodes au cadre de l'ACF en vue d'exploiter des treillis en désambiguïsation lexicale. Ici, le treillis de Galois est considéré comme un modèle de structuration d'une base lexicale. L'objectif est d'identifier des concepts formels (inférés par ACF sur un corpus d'apprentissage) potentiellement utiles pour l'annotation des entités nommées d'un corpus de test. Cette tâche d'annotation conceptuelle est définie de la manière suivante. Soit S l'ensemble des exemples à désambiguïser dans un corpus et $\mathfrak{T}$ l'ensemble des concepts formels d'un treillis $\underline{\mathfrak{T}}$. Une annotation conceptuelle consiste à associer chaque exemple $s \in S$ avec un concept $c \in \mathfrak{T}$. L'espace des annotations $\mathcal{A} = S \times \mathfrak{T}$ représente des couples d'exemples annotés par des concepts formels. Chaque unité lexicale $s \in S$ à désambiguïser peut être annotée par un grand nombre de concepts dans $\mathfrak{T}$. Nous proposons donc de réduire le choix des annotations possibles au sous-ensemble des concepts candidats $\mathfrak{T}(s, \mathit{contexte}(s)) \subseteq \mathfrak{T}$ qui décrit uniquement les concepts associés à un exemple s considéré dans son contexte $\mathit{contexte}(s)$. Les éléments de $\mathit{contexte}(s)$ correspondent ici aux unités lexicales qui entretiennent une relation de dépendance avec un exemple s. Finalement, l'opération d'annotation conceptuelle consiste à déterminer un unique concept formel $c \in \mathfrak{T}(s, \mathit{contexte}(s))$ en optimisant un score pour chaque unité lexicale s à désambiguïser dans un corpus. Ce score pouvant être calculé de différentes manières, plusieurs techniques sont envisageables pour résoudre une tâche d'annotation conceptuelle.

Nous avons mis au point deux stratégies pour résoudre la tâche d'annotation conceptuelle d'EN. La première s'intéresse à la désambiguïsation d'une relation

entre une entité nommée et un attribut en procédant à un calcul des confiances des règles d'association en intension et en extension. La seconde méthode généralise ce principe en prenant en compte plusieurs attributs en dépendance avec une entité nommée : elle consiste à sélectionner des concepts qui maximisent une distance sémantique dans un espace métrique associé aux concepts formels. Nous les présentons successivement dans cette section.

4.2.1 Annotation d'un objet en relation avec un attribut

Nous présentons ici un premier modèle d'exploitation des treillis de Galois en désambiguïsation lexicale [Girault, 2008b]. Chaque concept structuré dans un treillis construit à partir d'un corpus est considéré comme une annotation sémantique potentiellement utile pour désambiguïser une entité nommée de ce même corpus. Le treillis de Galois est exploré pour sélectionner un concept formel, plutôt que de sélectionner des unités de sens dans une ressource constituée manuellement. La méthode tire profit de la double structuration du treillis de Galois pour sélectionner les concepts formels désambiguïsant à la fois les EN et un de leurs attributs en relation.

4.2.1.1 Espace de recherche associé à un objet en relation avec un attribut

Dans un énoncé, on suppose qu'une EN $o \in O$ est en relation avec un attribut $a \in A$. L'objectif est d'annoter o avec un concept X du treillis en sachant que $\mathit{contexte}(o) = \{a\}$. De manière analogue, l'attribut a peut être annoté par un concept Y en considérant que $\mathit{contexte}(a) = \{o\}$. Notre proposition est la suivante. L'ensemble des concepts candidats pour l'annotation est le sous-treillis $\underline{\mathfrak{T}}(o, a)$ qui contient les concepts ayant l'objet o dans leur extension et a dans leur intension.

$$\underline{\mathfrak{T}}(o, a) = \{c \in \underline{\mathfrak{T}} | o \in \mathrm{ext}(c) \wedge a \in \mathrm{int}(c)\} \tag{4.2}$$

L'extension $E_x = \mathrm{ext}(X)$ du concept X qui annote o possède des objets similaires aux objets de l'extension de a (*i.e.*, E_x doit être proche de $E_a = \{a\}'$). L'intension $I_y = \mathrm{int}(Y)$ du concept Y qui annote a possède des attributs similaires aux attributs de l'intension de o (*i.e.*, I_y doit être proche de $I_o = \{o\}'$).

Les deux concepts (X, Y) à déterminer se situent sur des chemins entre le concept $C_o = (\{o\}'', \{o\}')$ associé à o et le concept $C_a = (\{a\}', \{a\}'')$ associé à a. Plus formellement, on a $C_o \leq X \leq C_a$ et $C_o \leq Y \leq C_a$. Cette notion de proximité entre les extensions ou les intensions peut se traduire par une combinaison de règles d'association.

4.2.1.2 Règles d'association pour la sélection des concepts

Dans le chapitre précédent, nous avons montré des liens existant entre l'apprentissage de règles d'association probabilistes et l'analyse de concepts formels. L'inclusion entre les intensions et les extensions appartenant à des niveaux conceptuels différents peut être formalisée par des règles d'implication notées $\alpha \rightarrow \beta$, avec $\alpha \subseteq \beta$. Le degré d'implication d'une règle $\alpha \rightarrow \beta$ peut être mesuré par la confiance et correspond à une probabilité conditionnelle.

$$conf(\alpha \rightarrow \beta) = P(\beta|\alpha) = \frac{P(\alpha,\beta)}{P(\alpha)} = \frac{|\beta'|}{|\alpha'|} \tag{4.3}$$

Chaque arc du treillis peut être associé à une mesure de confiance pour pondérer les règles d'association entre les objets des extensions.

$$x = \underset{C_a \leq X \leq C_o}{\operatorname{argmax}} \quad conf(E_o \rightarrow E_x) + conf(E_x \rightarrow E_a) \tag{4.4}$$

$$= \underset{C_a \leq X \leq C_o}{\operatorname{argmax}} \quad \frac{|I_x|}{|I_o|} + \frac{|I_a|}{|I_x|} \tag{4.5}$$

La double relation d'inclusion entre les niveaux conceptuels d'un treillis permet d'obtenir des règles d'objets ou d'attributs.

$$y = \underset{C_a \leq Y \leq C_o}{\operatorname{argmax}} \quad conf(I_a \rightarrow I_y) + conf(I_y \rightarrow I_o) \tag{4.6}$$

$$= \underset{C_a \leq Y \leq C_o}{\operatorname{argmax}} \quad \frac{|E_y|}{|E_a|} + \frac{|E_o|}{|E_y|} \tag{4.7}$$

Pour illustrer cette stratégie d'annotation, un exemple issu du corpus est maintenant examiné.

4.2.1.3 Exemple d'annotation conceptuelle

Dans l'énoncé proposé en figure 4.2, notre objectif est d'annoter l'expression « *champion Lindsay Davenport* » avec un des concepts contenant le couple (o, a) =(*Lindsay Davenport*, *champion*). Comme le montre la figure 4.1, l'espace de recherche $\underline{\mathfrak{T}}(o, a)$ est muni d'une relation d'ordre partiel et possède une structure de treillis.

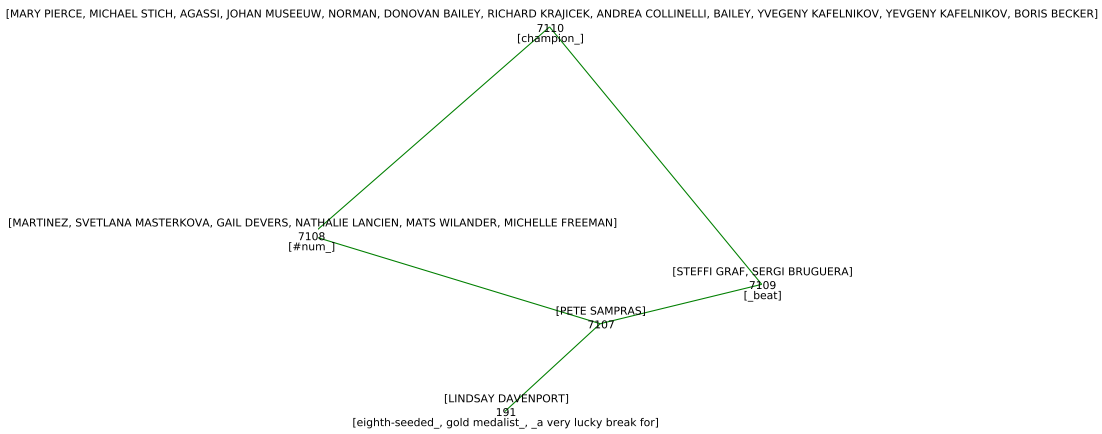


FIG. 4.1 – Concepts contenant la relation (*Lindsay Davenport*,*champion*)

Dans un premier temps, notre objectif n'est pas de faire une classification supervisée des EN mais plutôt d'en proposer une annotation fine issue d'une analyse non supervisée. L'exemple présenté (*cf.* figure 4.1) n'est pas ambigu. Dans d'autres cas, les éléments de l'intension pourraient décrire plusieurs « facettes » de l'EN. Conformément à notre méthode d'annotation conceptuelle, il convient de choisir une branche de sens afin de sélectionner un concept approprié précisant la facette retenue pour annoter l'EN. Sur ce point, les extensions des concepts sont constituées essentiellement de sportifs et plus particulièrement de joueurs de tennis. Les intensions des concepts décrivent précisément les usages d'entités nommées liées à des joueurs de tennis dans le corpus. L'intension du concept C_{191}, borne inférieure du treillis associé à *Lindsay Davenport*, est décrite par les attributs {*gold medalist_, _beat, eigth-seeded_, _a very lucky break for, champion_, #num_*}. L'application des formules 4.5 et 4.7 permet de sélectionner les concepts C_{7109} et C_{7108} pour annoter (o, a) :

- C_{7109} =({Steffi Graf, Sergi Bruguera, Lindsay Davenport}, {champion, beat})
- C_{7108} =({Svetlana Masterkova, Gail Devers, Michelle Freeman, Martinez, Nathalie Lancien, Mats Wilander, Lindsay Davenport, Pete Sampras}, {champion_, #num_})

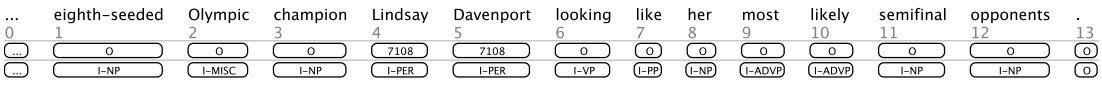


FIG. 4.2 – Exemple d'annotation conceptuelle sur le corpus CoNLL

La limitation principale de cette stratégie d'annotation est qu'elle n'est pas capable de prendre en compte plusieurs attributs du contexte pour annoter un

objet. Afin de dépasser cette limitation, nous proposons maintenant une généralisation de notre méthode.

4.2.2 Annotation d'entités nommées en relation avec plusieurs attributs

Nous proposons ici une autre stratégie d'annotation conceptuelle [Girault, 2008a] capable de prendre en compte plusieurs attributs. Cette proposition est justifiée par le fait que nous avons employé un analyseur en dépendances (*cf.* chapitre 2) capable d'extraire plusieurs dépendants (attributs) en relation syntaxique avec une même entité nommée (objet) dans un énoncé. Nous reprenons l'hypothèse énoncée dans [Yarowsky, 1995] (*one sense per discourse*) qui suppose qu'une unité lexicale a de bonnes chances d'avoir le même sens dans un même document. Nous supposons que chaque EN est monosémique dans une dépêche. La description d'une EN est alors étendue à toutes ses dépendances observées au niveau d'une dépêche. Cette hypothèse permet d'augmenter le nombre d'attributs, de réduire l'espace de recherche et le temps de traitement nécessaire pour annoter les EN d'une dépêche.

Nous proposons ici d'adapter l'espace de recherche des concepts candidats pour l'annotation de manière à considérer plusieurs attributs. Nous reprenons la représentation continue des concepts formels proposée dans le chapitre 3 pour identifier un concept représentatif de l'espace de recherche utilisé pour l'annotation conceptuelle. Pour terminer, nous adaptons l'algorithme de Lesk à notre cadre d'annotation conceptuelle des EN.

4.2.2.1 L'annotation comme une recherche de concepts candidats dans un treillis

La question qui se pose est de déterminer les concepts potentiellement intéressants pour annoter une entité nommée en relation avec plusieurs attributs. Dans une dépêche, nous supposons que les occurrences d'une EN $o \in O$ et ses dépendances $A_i = \mathit{contexte}(o) =\subseteq A$ ont été identifiées à l'aide d'une analyse en dépendances (*cf.* chapitre 2). Dans une tâche de désambiguïsation lexicale, on considère que le sens de o dépend du sens de ses dépendances A_i qui apparaissent dans son contexte. Notre objectif est d'annoter l'EN o avec un concept formel x du treillis en fonction des concepts associés aux attributs A_i en relation avec o dans une dépêche.

Dans le treillis $\underline{\mathfrak{T}}$, l'objet o est associé à $Co = (\{o\}'', \{o\}')$ et, de manière similaire, les concepts pour les attributs de A_i sont les éléments de $C_{A_i} = \{(\{a_i\}', \{a_i\}'') | a_i \in A_i\}$. Le concept x est recherché parmi les concepts de $\underline{\mathfrak{T}}(o, A_i)$ qui contiennent o

dans leurs extensions et au moins un dépendant a_i dans leurs intensions.

$$\underline{\mathfrak{T}}(o, A_i) = \{(E, I) \in \underline{\mathfrak{T}} | o \in E \wedge A_i \cap I \neq \emptyset\} \tag{4.8}$$

L'ensemble des concepts candidats est partiellement ordonné et possède une structure de treillis. Lorsque l'ensemble A_i possède plusieurs éléments, l'ensemble des concepts candidats $\underline{\mathfrak{T}}(o, A_i)$ est l'union des sous-treillis contenant l'objet o et un des attributs $a_i \in A_i$.

Nous n'avons pas besoin de reconstruire ce sous-treillis $\underline{\mathfrak{T}}(o, A_i)$ car il peut être directement extrait du treillis complet $\underline{\mathfrak{T}}$. La procédure pour constituer $\underline{\mathfrak{T}}(o, A_i)$ (*cf.* algorithme 7) est un parcours en largeur du graphe associé à $\underline{\mathfrak{T}}$.

Algorithme 7: Algorithme de détermination du sous-treillis $\underline{\mathfrak{T}}(o, A_i)$

```
   Entrées : 𝔗, o, A_i
   Sorties : 𝔗(o, A_i)
 1 concepts = ∅
 2 queue = [(o'', o')]
 3 tant que queue ≠ ∅ faire
 4 |   x = queue.dequeue() // sort le concept de la file et l'assigne à x
 5 |   si x ∈ concepts alors
 6 |   |   continue
 7 |   si int(x) ∩ A_i ≠ ∅ alors
 8 |   |   concepts = concepts ∪ {x}
 9 |   |   pour chaque parent ∈ parents(x) faire
10 |   |   |   // mettre les parents du concept x dans la file queue
11 |   |   |   queue.enqueue(parent)
12 retourner concepts
```

À l'instar de l'espace des versions en classification supervisée, le treillis est considéré comme un espace de recherche dans lequel on veut trouver des concepts (c'est-à-dire des hypothèses de regroupement entre des unités lexicales) qui fournissent une généralisation intéressante pour expliquer et désambiguïser une EN. Il s'agit de trouver des concepts, ni trop spécifiques, ni trop généraux, c'est-à-dire des concepts qui possèdent plus d'objets que le concept C_o et plus d'attributs que les concepts de C_{A_i}. Afin d'identifier un concept représentatif de l'espace de recherche $\underline{\mathfrak{T}}(o, A_i)$, nous exploitons maintenant la représentation vectorielle des concepts à laquelle (*cf.* chapitre 3) nous associons la métrique du χ^2.

4.2.2.2 Détermination du prototype d'un ensemble de concepts

Notre objectif est d'annoter une EN o en fonction de ses dépendances $\{a_0, a_1 \ldots\}$ dans une dépêche par un concept représentant $\underline{\mathfrak{T}}(o, A_i)$. Il nous semble difficile de

généraliser la formule 4.5, d'autant qu'elle ne permet pas de prendre en compte le nombre d'occurrences des unités lexicales du corpus qui constituent les objets et les attributs du treillis. Notre stratégie alternative consiste à sélectionner un concept représentant l'ensemble $\underline{\mathfrak{T}}(o, A_i)$.

Nous allons déterminer ce concept dans la représentation matricielle $M_{\underline{\mathfrak{T}}(o,A_i)}$, associée à $\underline{\mathfrak{T}}(o, A_i)$. La matrice $M_{\underline{\mathfrak{T}}(o,A_i)}$ représente l'espace sémantique continu muni d'une mesure du χ^2 qui est considérée comme une métrique. On cherche à y identifier un concept se situant en une position intermédiaire entre le concept le plus spécifique (o'', o') et les concepts généraux $(a', a''), \forall a_i \in A_i$. Nous appelons ce concept le prototype ou médoïde de C. Il est défini comme le concept $x \in C$ dont la dissimilarité avec les autres concepts de C est minimale :

$$prototype(C) = \underset{x \in C}{\operatorname{argmin}} \quad \sum_{y \in C} \delta(x, y) \tag{4.9}$$

Le calcul du prototype intervient dans l'algorithme de désambiguïsation des entités nommées d'une dépêche. Nous examinons maintenant le détail de cette procédure d'annotation conceptuelle.

4.2.2.3 Algorithme de désambiguïsation

Kilgarriff et Rosenzweig [2000] ont proposé une version simplifiée de l'algorithme de Lesk reposant sur le calcul du partage des mots entre la définition et le contexte d'un mot à désambiguïser. Leur nouvelle version de l'algorithme évite ainsi une comparaison de toutes les définitions associées à tous les mots du contexte, nécessaire avec la méthode initialement proposée par Lesk [1986]. Cette approche permet de réduire l'espace de recherche de manière significative lorsque le contexte à prendre en compte contient plus de deux unités lexicales. De plus, une amélioration des résultats est observée par rapport à la méthode de Lesk [Vasilescu *et al.*, 2004].

Notre algorithme de désambiguïsation exploite cette simplification pour éviter d'aboutir à des parcours complexes dans le treillis. La méthode de Kilgarriff et Rosenzweig [2000] est ici adaptée pour manipuler des concepts formels plutôt que des définitions d'un dictionnaire. L'Algorithme 8 exploite ainsi les concepts d'un treillis de Galois $\underline{\mathfrak{T}}$ constitué à partir d'un corpus pour annoter les EN d'une dépêche d.

Algorithme 8: Algorithme d'annotation conceptuelle

Entrées : Treillis de Galois $\underline{\mathfrak{T}}$, dépêche d
Sorties : mots de l'énoncé annotés par des définitions

$\mathbb{K}_d = (O_d, A_d, R_d)$
pour chaque $o \in O_d$ **faire**
    $A_i = \{a \in A_d | (o, a) \in R_d\}$
    $annotation[o] = prototype(\underline{\mathfrak{T}}(o, A_i))$
pour chaque *énoncé* $e \in d$ **faire**
    pour chaque *entité nommée* o *de l'énoncé* **faire**
        Récupérer le concept formel $c = annotation[o]$ pour l'entité nommée o
retourner *Retourner la dépêche annotée*

Trois étapes sont nécessaires pour cet algorithme d'annotation conceptuelle. La première consiste à lire la dépêche d pour construire son contexte formel $\mathbb{K}_d = (O_d, A_d, R_d)$ qui représente l'ensemble de ses entités nommées en relation avec leurs dépendants dans d. Pour la deuxième, on construit l'espace de recherche $\underline{\mathfrak{T}}(o, A_i)$ pour chaque objet $o \in O_d$ et l'ensemble de ses dépendants A_i dans la dépêche d, de manière à déterminer le prototype de o, en identifiant le concept qui minimise sa dissimilarité avec les autres concepts de $\underline{\mathfrak{T}}(o, A_i)$. La troisième étape nécessite une relecture de la dépêche d pour associer à chacune de ses entités nommées son concept prototype. Cette dernière étape est considérée comme une désambiguïsation non supervisée.

Notre travail ne se focalisera pas sur l'étude formelle de la complexité de l'algorithme d'annotation conceptuelle. En revanche, nous pouvons donner quelques tendances à propos du comportement de l'Algorithme **??** sur le corpus de test. La taille de l'espace de recherche $\underline{\mathfrak{T}}(o, A_i)$ varie selon le degré de polysémie des unités lexicales o et A_i. Le nombre de concepts dans $\underline{\mathfrak{T}}(o, A_i)$ a forcément tendance à croître lorsque o et les éléments de A_i sont très polysémiques ; pour une majorité de cas cependant, $\underline{\mathfrak{T}}(o, A_i)$ possède moins de trois concepts. La construction de la matrice $M_{\underline{\mathfrak{T}}(o,A_i)}$ et la détermination de son prototype doivent être faites pour chaque objet d'une nouvelle dépêche.

En pratique, pour un treillis de concepts $\underline{\mathfrak{T}}$ de 10000 concepts, le nombre de concepts dans $\underline{\mathfrak{T}}(o, A_i)$ est généralement inférieur à 100. Sur l'ordinateur utilisé pour nos expérimentations [2], le chargement du treillis en mémoire prend environ 5 secondes et l'exécution de l'algorithme de désambiguïsation sur le corpus de test dure moins de 3 secondes. Les temps de traitement sont encore tout à fait acceptables avec un treillis de 50000 concepts.

2. Configuration utilisée : Linux 2.6.28, Python 2.6.4, processeur Intel Core 2 Duo T8300 2.4GHz et 2Go de mémoire vive.

L'utilisation de l'hypothèse proposée par [Yarowsky, 1995] (*one sense per discourse*) réduit significativement le temps de calcul. Il pourrait être intéressant d'exploiter cette hypothèse sur les attributs extraits d'une dépêche. Il faut noter toutefois que, pour certaines unités lexicales très polysémiques, l'hypothèse n'est plus valable : par exemple, l'unité lexicale *said* peut prendre plusieurs sens dans une dépêche et constitue rarement une variable discriminante.

4.2.3 Annotation conceptuelle : amélioration de la robustesse

Le processus d'annotation conceptuelle que nous avons élaboré n'est pas suffisamment robuste pour étiqueter des EN inconnues. Nous avons considéré jusqu'ici qu'il existe toujours au moins un concept dans $\underline{\mathfrak{T}}(o, A_i)$ pour annoter une EN o en dépendance avec les attributs de A_i ; plus formellement, nous supposons que $\underline{\mathfrak{T}}(o, A_i) \neq \emptyset$. Cette hypothèse est vérifiée lorsque le treillis manipulé pour l'annotation conceptuelle d'un corpus contient les relations extraites de ce corpus. Cependant, pour nos expérimentations à venir, nous utiliserons des corpus séparés pour la phase d'apprentissage (*train*) et de test (*testb*) et ces derniers ne contiennent pas les mêmes relations. Dans ces expérimentations, le treillis employé pour l'annotation conceptuelle est uniquement constitué à partir du corpus d'apprentissage. Notre méthode actuelle exploitant ce treillis ne nous permet donc pas d'annoter le corpus de test.

Pour améliorer la robustesse du processus d'annotation conceptuelle, une stratégie possible est de constituer un treillis $\underline{\mathfrak{T}}_{train}$ à partir du corpus d'apprentissage uniquement, puis de réutiliser ses concepts sur le corpus de test lorsque ceci est possible. Notons toutefois que pour de nouvelles relations identifiées entre un objet et des attributs, les concepts du treillis $\underline{\mathfrak{T}}_{train}$ ne sont pas utilisables. L'espace de recherche $\underline{\mathfrak{T}}(o, A_i)$ pourrait être substitué par un ensemble de concepts de $\underline{\mathfrak{T}}_{train}$ caractérisant les éléments connus de la nouvelle relation. Ce nouvel espace de recherche, noté $\hat{\underline{\mathfrak{T}}}(o, A_i)$, correspond à l'ensemble des concepts qui seraient modifiés si les nouveaux éléments en cours d'annotation devaient être intégrés à $\underline{\mathfrak{T}}_{train}$.

Lors de l'insertion d'une nouvelle relation (o, a), les concepts $c_o = (o'', o')$ et $c_a = (a', a'')$ devront être modifiés de manière à ce que, suite à l'intégration, on ait $c_o = (o'', o' \cup \{a\})$ et $c_a = (a' \cup \{o\}, a'')$, une fusion entre les deux concepts étant possible. Les autres concepts qui doivent être modifiés possèdent des objets a' associés à l'attribut a ainsi que les ensembles d'attributs o' associés à l'objet o. Finalement, l'ensemble des concepts modifiés lors de l'insertion d'une nouvelle relation (o, a) s'écrit de la manière suivante :

$$\hat{\underline{\mathfrak{T}}}(o, a) = \{(E, I) \in \underline{\mathfrak{T}} | E \subseteq (o'' \cup a') \text{ et } I \subseteq (o' \cup a'')\} \tag{4.10}$$

À partir de $\hat{\underline{\mathfrak{T}}}(o, a)$, on peut exprimer l'ensemble des concepts $\hat{\underline{\mathfrak{T}}}(o, A_i)$ modifiés lors de l'insertion des relations entre un objet o et des attributs de l'ensemble A_i par :

$$\hat{\underline{\mathfrak{T}}}(o, A_i) = \bigcup_{a \in A_i} \hat{\underline{\mathfrak{T}}}(o, a) \tag{4.11}$$

L'espace de recherche $\hat{\underline{\mathfrak{T}}}(o, A_i)$ peut être déterminé à l'aide d'une procédure similaire à l'Algorithme 7 employé pour le calcul de $\underline{\mathfrak{T}}(o, A_i)$. Dans nos expérimentations à venir, l'espace de recherche $\hat{\underline{\mathfrak{T}}}(o, A_i)$ sera utilisé uniquement dans le cas où $\mathfrak{T}(o, A_i)$ ne contient aucun concept. Nous présentons dans la section suivante les expérimentations validant l'apport de notre stratégie d'annotation conceptuelle pour une tâche de classification supervisée d'EN.

4.3 Évaluation en cascade d'annotations conceptuelles

L'objectif de cette section est d'évaluer notre méthode d'annotation conceptuelle à l'aide d'expérimentations sur corpus. Notre méthode est capable d'étiqueter automatiquement des EN d'un corpus de test avec des concepts formels inférés par ACF de manière non supervisée sur un corpus d'apprentissage. La pertinence des concepts employés pour notre annotation conceptuelle est difficile à établir en l'absence d'un corpus de référence (vérité terrain) disposant d'une annotation conceptuelle manuelle. Une annotation manuelle de ce type serait d'ailleurs difficile à réaliser étant donné que plusieurs concepts formels d'un treillis peuvent être appropriés pour désambiguïser une EN.

Pour éviter ces problèmes et favoriser une évaluation automatique, notre technique d'annotation conceptuelle (non supervisée) est validée dans le cadre d'un protocole d'évaluation en cascade [Candillier *et al.*, 2006]. Ce protocole se déroule en deux étapes : la première repose sur l'enrichissement d'un corpus en exploitant les résultats d'une annotation conceptuelle ; la seconde consiste à vérifier si un algorithme d'apprentissage supervisé bénéficie de cet enrichissement pour une tâche de classification en comparant ses résultats avec ceux d'un apprentissage et d'une classification réalisés sur des données non enrichies. Notre approche est validée sur une tâche de classification supervisée d'EN dont on peut mesurer automatiquement les performances. Nous évitons ainsi d'avoir recours à une expertise.

Dans cette section, nous spécifions en premier lieu le déroulement de nos expérimentations : nous décrivons la tâche supervisée de classification d'EN, une

méthode existante d'apprentissage pour la résoudre, ainsi que les mesures de performances employées dans le cadre de l'évaluation en cascade. Notre démarche générale est alors illustrée par l'annotation conceptuelle des EN d'un énoncé : cette annotation est produite automatiquement à partir d'un treillis constitué à partir d'un corpus d'apprentissage non étiqueté. Nous examinons ensuite expérimentalement si l'exploitation de ce treillis par annotation conceptuelle permet d'améliorer la classification supervisée d'EN dans un corpus de test. À l'issue de cette section, les limites de notre approche sont discutées.

4.3.1 Protocole d'évaluation en cascade

Nous employons ici un protocole d'évaluation en cascade [Candillier *et al.*, 2006] pour valider l'apport de l'annotation conceptuelle (non supervisée) à une tâche de classification supervisée des EN sur le corpus CoNLL-2003 (voir sous-section suivante). Le terme cascade signifie ici que des systèmes d'apprentissage artificiel sont utilisés successivement : l'apprentissage non supervisé est considéré comme un prétraitement d'une tâche supervisée que l'on sait évaluer. Le système non supervisé employé en prétraitement met en œuvre la construction du treillis de Galois par ACF, suivie d'une annotation conceptuelle produisant l'enrichissement des EN, par des concepts formels, dans des corpus d'entraînement et de test. Cet enrichissement peut être quantifié en employant deux systèmes A et B qui fonctionnent avec le même algorithme de classification. Le système de référence (ou témoin) A est un classifieur supervisé ayant appris un modèle prédictif sur le corpus d'apprentissage étiqueté (étiquettes sémantiques prédéfinies : PER, LOC, ORG et MISC). Le système en cascade B est aussi un classifieur supervisé, mais cette fois, il est entraîné et testé sur des corpus enrichis par annotation conceptuelle issue d'une analyse non supervisée. Les concepts formels employés pour l'annotation fournissent alors des informations supplémentaires exploitées pour l'apprentissage supervisé du système B. Les performances des systèmes A et B peuvent être ensuite être comparées automatiquement par rapport à la tâche de classification supervisée. Cette comparaison permet alors de vérifier si le gain apporté par une annotation conceptuelle est significatif pour cette tâche.

4.3.1.1 Classification supervisée sur un corpus enrichi

L'évaluation de notre technique d'annotation conceptuelle repose sur une tâche de classification supervisée d'EN qui peut être présentée de la manière suivante. Soit X l'ensemble des exemples, $Labels$ l'ensemble des étiquettes sémantiques possibles des exemples et L (pour *Labelled*) l'espace $X \times Labels$ (les éléments de L sont des paires constituées d'un exemple et de son étiquette sémantique). Les étiquettes sémantiques possibles pour un mot sont $C = \{$PER, LOC,

ORG, MISC}. Pour la tâche de classification des EN, un exemple est représenté par un mot (ou *token*) et ses caractéristiques principales, à savoir, l'étiquette morphosyntaxique (*part of speech* : nom commun, verbe, préposition...) et l'étiquette du *chunk*[3] associé au mot. Dans le cas du classifieur B, on ajoute une annotation conceptuelle issue d'un treillis de Galois. Cette annotation correspond à un identifiant numérique créé lors de la construction du treillis. De nombreuses techniques d'apprentissage supervisé sont utilisables pour résoudre une tâche de classification d'EN. Rappelons que l'approche choisie nous permet de mesurer les différences de performance entre un système témoin A et un système B ayant bénéficié de notre annotation conceptuelle.

La technique des champs conditionnels aléatoires (*Conditional Random Fields* ou CRF) a été employé pour la tâche de classification des EN. Les champs conditionnels aléatoires sont des automates à états finis probabilistes ayant pour objectif d'étiqueter et segmenter les séquences de données [Lafferty *et al.*, 2001]. Nous n'allons pas détailler ici le fonctionnement des CRF car les d'outils mathématiques complexes dont ils font usage ne sont pas nécessaires pour comprendre le déroulement de notre approche. N'importe quelle autre méthode d'apprentissage supervisé aurait pu être employée à cette tâche. Au-delà des performances sur notre tâche de classification d'EN, nos choix ont été motivés par la disponibilité d'outils libres et accessibles[4].

Nous précisons maintenant les mesures de performance utilisées habituellement en reconnaissance d'EN. Elles permettent de faire une comparaison quantitative entre deux systèmes CRF ayant appris leur modèle prédictif respectivement sur des corpus bruts et enrichis.

4.3.1.2 Critère de performance pour l'évaluation des systèmes de reconnaissance d'entités nommées

Pour évaluer et comparer quantitativement les sorties produites par les différents systèmes, les critères de performance utilisés sont le rappel, la précision et la F-mesure [van Rijsbergen, 1975]. Dans notre cadre, les exemples d'apprentissage et de test peuvent être classés dans plusieurs classes $Labels = \{$`PER`, `LOC`, `ORG`, `MISC`$\}$. Pour l'évaluation, nous considérons chaque étiquette $l \in Labels$ de manière indépendante. L'évaluation de notre système de classification multi-classes se rapporte alors à l'évaluation de plusieurs classifieurs binaires. Chaque classe $l \in Labels$ est attribuée à un classifieur binaire considérant qu'un exemple étiqueté par l est positif et qu'un exemple non étiqueté par l est négatif. Les mesures

3. Cette étiquette indique à la fois la catégorie du *chunk* (nominal, verbal...) et la position du mot dans le *chunk* (« B » pour indiquer le premier mot du chunk et « I » pour les mots suivants du *chunk*).

4. Le logiciel CRF++ a été utilisé pour cette tâche : `http://crfpp.sourceforge.net/`.

de rappel et précision peuvent être expliquées avec une matrice de confusion [Kohavi et Provost, 1998] qui contient l'information sur les classifications réelles et prédites par un système de classification. Chaque classifieur binaire attribué à une étiquette $l \in Labels$ est associé à une matrice de confusion représentée dans la table 4.1 :

Vérité \ Prédiction	Négatif	Positif	Total
Négatif	VN_l	FP_l	$VN_l + FP_l$
Positif	FN_l	VP_l	$FN_l + VP_l$
Total	$VN_l + FN_l$	$FP_l + VP_l$	$VN_l + FP_l + FN_l + VP_l$

TABLE 4.1 – Entrées d'une matrice de confusion pour un classifieur binaire attribué à une étiquette $l \in Labels$

Les mesures de précision et rappel pour une étiquette l peuvent être exprimées par des rapports entre les valeurs VP_l, FP_l et FN_l, où VP_l (vrais positifs) est le nombre d'exemples positifs classés positivement ; FP_l (faux positifs) est le nombre d'exemples négatifs classés positivement ; FN_l (faux négatifs) est le nombre d'exemples positifs classés négativement ; de plus, on note VN_l (vrais négatifs) le nombre d'exemples négatifs classés négativement.

La précision attribuée à une étiquette $l \in Labels$ exprime le nombre d'EN correctement étiquetée par l rapporté au nombre d'EN classées par l dans l'ensemble du corpus. Le principe est le suivant : l'objectif d'un algorithme de classification est de classer correctement les EN qu'il parvient à classer. Toutes les EN mal classées constituent du « bruit ». La précision s'oppose au bruit. Si elle est élevée, cela signifie que peu d'EN sont mal classées par le système et que ce dernier peut être considéré comme « précis ». On calcule la précision pour une étiquette $l \in Labels$ avec la formule suivante :

$$P_l = \frac{VP_l}{VP_l + FP_l} = \frac{\text{Nombre de prédictions correctes attribuées à la classe } l}{\text{Nombre de prédictions de classe } l} \tag{4.12}$$

Le calcul de la précision globale exprime le nombre total d'EN correctement classées rapporté au nombre total d'EN classées du corpus.

$$P_{global} = \frac{\sum_{l \in Labels} VP_l}{\sum_{l \in Labels} VP_l + FP_l} \tag{4.13}$$

Cette précision globale correspond en fait à une micro-moyenne qui reprend les entrées des matrices de confusion attribuées à chaque étiquette de $Labels$.

Contrairement à une moyenne (appelée macro-moyenne) des précisions P_l, cette micro-moyenne est capable de prendre en compte le nombre d'exemples associés à chaque classe.

Le rappel attribué à une étiquette $l \in Labels$ est défini par le ratio entre le nombre d'EN correctement étiquetées par l et le nombre d'EN classées par l que possède le corpus de test. Si cette adéquation entre les EN classées correctement et le nombre d'EN du corpus est importante alors le taux de rappel est élevé. À l'inverse, si le corpus possède de nombreuses EN qui n'ont pas été identifiées et classées, on parle de silence. Le silence s'oppose au rappel. Le rappel pour l'étiquette l peut être mesuré de la manière suivante :

$$R_l = \frac{VP_l}{VP_l + FN_l} = \frac{\text{Nombre de prédictions correctes attribuées à la classe } l}{\text{Nombre d'exemples de classe } l} \tag{4.14}$$

Le rappel global est défini par la proportion d'EN correctement étiquetées par rapport au nombre d'EN classées que possède le corpus de test.

$$R_{global} = \frac{\sum_{l \in Labels} VP_l}{\sum_{l \in Labels} VP_l + FN_l} \tag{4.15}$$

Le rappel global est une micro-moyenne qui utilise les entrées des matrices de confusion attribuées à chaque étiquette, permettant ainsi de prendre en compte le nombre d'exemples associés à chaque étiquette.

La précision et le rappel sont des mesures intimement liées. Si plus d'exemples sont classés, le rappel augmente mais la précision peut diminuer parce que tous les exemples ne sont pas forcément correctement classés. La F-mesure considère à la fois la précision et le rappel. Elle peut être interprétée comme une moyenne pondérée entre la précision et le rappel. La F-mesure est calculée de la manière suivante :

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 R + P} \tag{4.16}$$

Cette F-mesure peut être calculée avec des mesures de P et R globales ou spécifiques à une étiquettes $l \in Labels$.

De nombreux systèmes sont optimisés pour maximiser le score $F_{\beta=1}$. Cependant, pour certaines applications cette pratique n'est pas sytématiquement recommandée [5]. Par ailleurs, la différence entre deux F-mesures globales (sur l'ensemble des étiquettes) correspondant à l'évaluation de deux systèmes n'est pas un indicateur suffisant pour prouver que les améliorations (ou dégradations) apportées par

5. Dans un billet sur *blog* de Hal Daumé en août 2006, Christopher D. Manning explique pourquoi ne pas optimiser les systèmes de reconnaissance des entités nommées pour la mesure F_β (`http://nlpers.blogspot.com/2006/08/doing-named-entity-recognition-dont.html`).

un des deux systèmes ne sont pas dues au hasard. À cet égard, nous présentons à présent deux tests statistiques indiquant si les différences de performances entre deux systèmes sont significatives.

4.3.1.3 Tests statistiques pour la comparaison de performances

Dans le cadre d'une évaluation en cascade, nous sommes amené à comparer les résultats d'un système « témoin » A (*baseline*) avec ceux d'un système B ayant subi un traitement particulier (en l'occurrence, l'apprentissage à partir d'un corpus ayant bénéficié d'une annotation conceptuelle). Pour une interprétation plus rigoureuse de nos résultats, nous décrivons ici des tests indiquant si les différences de performances entre les systèmes A et B sont statistiquement significatives. Ces tests consistent à rejeter ou à accepter l'hypothèse que les différences observées sont non significatives, appelée l'hypothèse nulle H_0, en fonction d'un ensemble de mesures établies sur des données. Les tests que nous employons estiment la probabilité $P(H_0)$ que l'hypothèse H_0 soit vérifiée. Une probabilité faible $P(H_0)$ indique que les différences de performances entre deux systèmes ne sont pas dues au hasard et sont donc statistiquement significatives : nous pouvons ainsi établir quel est le « meilleur » système.

Les n mesures examinées par ces tests statistiques correspondent aux scores de précision et de rappel associées à chaque étiquette (PER, LOC, ORG, MISC) pour un système donné : chaque système est donc caractérisé par $n = 8$ scores notés m_i. Soit $\delta_i = m_i^A - m_i^B$, la différence entre deux scores associés à deux systèmes A et B à comparer et la moyenne de ces différences $\overline{\delta} = \frac{1}{n}\sum_{i=1}^{n}\delta_i$.

Le test-t de Student (dans sa version *paired t-test*) est un test statistique paramétrique qui repose sur la comparaison de moyennes entre deux échantillons statistiques. Cette comparaison utilisant l'écart type corrigé $s(\delta_i) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\delta_i - \overline{\delta})^2}$ se calcule de la manière suivante :

$$t_s = \frac{\overline{\delta}}{s(\delta_i)}\sqrt{n} \tag{4.17}$$

Sous l'hypothèse H_0, la valeur t_s suit une loi de Student avec $n-1$ degrés de liberté qui détermine la probabilité $P_a(H_0)$ que la différence de performance entre deux systèmes est non significative : avec 7 degrés de liberté, on a par exemple $P_a(H_0) = 0.01$ pour $t \simeq= 3.499$ et $P_a(H_0) = 0.5$ pour $t \simeq 0.718$.

Le test de Wilcoxon (*Wilcoxon signed-rank test*) est une hypothèse statistique non paramétrique considérée comme une alternative au test-t de Student. Le test de Wilcoxon procède à des comparaisons de différences entre des mesures, mais contrairement au test-t de Student, il ne fait aucune hypothèse sur la loi statistique suivie par les mesures étudiées. Le calcul effectué pour ce nouveau test emploie le signe de la différence δ_i et $rang(|\delta_i|)$ le rang occupé par $|\delta_i|$ dans le classement de la plus petite à la plus grande valeur absolue des différences :

$$t_w = \frac{\sum_i^n R_i}{\sqrt{\sum_i^n R_i^2}} \tag{4.18}$$

$$\text{avec} \quad R_i = signe(\delta_i) * rang(|\delta_i|) \tag{4.19}$$

On peut ensuite déterminer la probabilité $P_w(H_o)$ en fonction de n et t_w à partir d'une table de Wilcoxon.

Nous utilisons dans nos expérimentations un seuil $\theta = 0.1$, tel que si $P(H_0) < \theta$, l'hypothèse H_0 est rejetée : nous pouvons de cette manière déterminer si les améliorations (ou les dégradations) des performances de notre système d'annotation conceptuelle sont statistiquement significatives.

4.3.2 Exemple d'annotation conceptuelle

À titre d'illustration, nous examinons ici un exemple d'annotation conceptuelle produit par notre système. Dans le corpus d'apprentissage (*train*), l'entité nommée $o = Iranian$ apparaît dans un document avec les attributs $A_i = \{border, opposition, leaders, exile\ group, troops\}$. L'espace de recherche $\underline{\mathfrak{T}}(o, A_i)$ (*cf.* figure 4.3) contient l'ensemble des concepts possédant des relations entre o et chaque attribut de A_i.

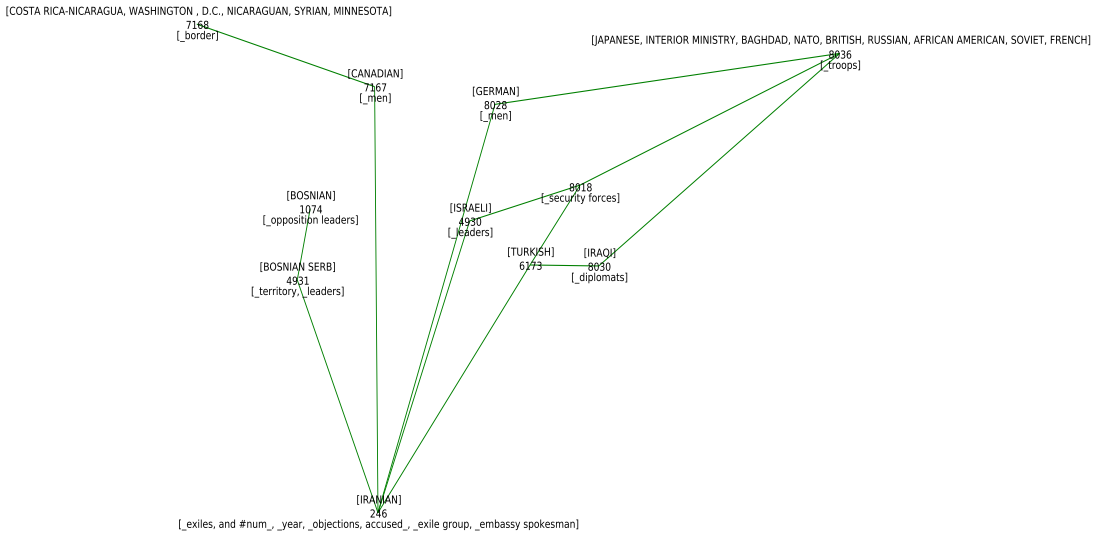


FIG. 4.3 – Concepts contenant des unités lexicales en relation avec l'EN *Iranian* dans un document du corpus d'apprentissage *train*

Comme nous pouvons l'observer, des attributs n'appartenant pas à A_i apparaissent dans différentes branches de $\underline{\mathfrak{T}}(o, A_i)$: les branches qui contiennent l'attribut *troops* font apparaître les UL {*security forces, diplomats, men*}. Les extensions de ces concepts correspondent principalement à des nationalités ({*Iraqui, German, Israeli, Turkish*}) : ces entités nommées sont toutes classées avec l'étiquette `MISC` dans le corpus. Les concepts de $\underline{\mathfrak{T}}(o, A_i)$ comportent donc des objets qui généralisent l'usage de o à d'autres usages sémantiquement proches. Cette généralisation est pour nous une première étape vers la désambiguïsation.

La figure 4.4 présente un exemple d'énoncé extrait de notre corpus qui a été enrichi par une annotation conceptuelle. Plusieurs types d'étiquettes sont placés sous chaque mot de l'énoncé. Les étiquettes numériques (première couche d'annotation) correspondent aux identifiants des concepts dont la description est détaillée dans la figure 4.5 : elles apparaissent uniquement pour les mots constituant des EN. Par ailleurs, sur la seconde couche d'annotation, les EN sont étiquetées par une catégorie sémantique (`I-MISC`, `I-LOC`) prédite par l'algorithme de classification supervisée. Les mots qui ne sont pas considérés comme des EN ne sont pas annotés conceptuellement (ce qui est marqué ici par `O` sur la première couche d'annotation) et leur étiquette sémantique est ici substituée par l'étiquetage représentant le *chunking* de l'énoncé (`I-NP`, `I-VP`, `I-PP`...) pour la seconde couche d'annotation.

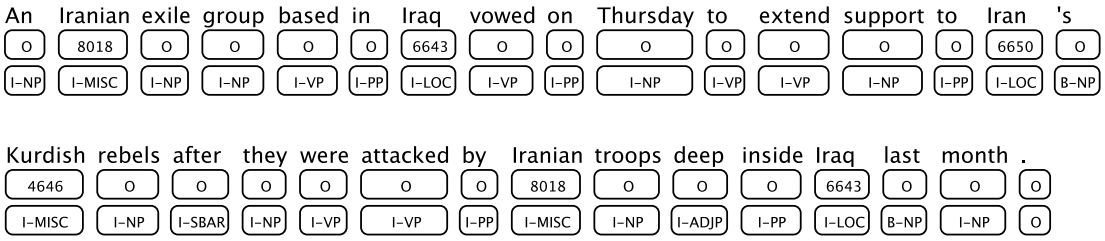


FIG. 4.4 – Exemple d'annotation conceptuelle sur un énoncé

- C_{4646} =({Kurdish, PKK, Moslem, Tamil Tiger}, {_rebels, _guerrillas})
- C_{6643} =({Afghanistan, Israel, Italy, India, Costa Rica, France, Iraq}, {and_, northern_, NUM_})
- C_{6650} =({China, Pakistan, Iran, Haarlem, Yeltsin}, {and_, on_})
- C_{8018} =({Israeli, Iranian, Turkish}, {_security forces, _troops})

FIG. 4.5 – Concepts associés à la figure 4.4

Nous pouvons remarquer que les résultats de l'annotation conceptuelle sur cet énoncé sont mitigés. Les extensions des concepts $C_{6643}, C_{6650}, C_{8018}$ regroupent des EN qui possèdent en majorité la même étiquette sémantique (respectivement,

LOC, LOC et MISC sur la figure 4.4). En revanche, l'extension du concept C_{4646} contient des objets dont la catégorie sémantique est différente : en effet, *Kurdish* et *Moslem* correspondent normalement à l'étiquette `MISC` alors que *PKK* et *Tamil Tiger* sont assimilés à la catégorie `ORG`. En dépit de cette divergence, on peut noter que les UL de l'intension (*rebels, guerrillas*) sont sémantiquement proches.

Par ailleurs, l'EN *Iranian* possède deux occurrences qui désignent des référents différents. La première occurrence *Iranian exile group* fait référence à un groupe d'expatriés opposants au régime iranien, alors que *Iranian troops* désigne plutôt les troupes militaires de l'Iran s'attaquant à ses détracteurs. Ces différentes occurrences pourraient être annotées par des concepts différents si l'hypothèse de Yarowsky [1995] (*one sense per discourse*) n'avait pas été appliquée. Le relâchement de cette contrainte n'est toutefois pas recommandé dans le cadre de notre tâche de classification supervisée. Bien que les deux occurrences de l'EN *Iranian* soient associées à des référents différents, elles possèdent la même étiquette sémantique `MISC`. Nous avons calculé que 97% des EN du corpus d'apprentissage possèdent une même étiquette sémantique étant donné un document : l'hypothèse de Yarowsky [1995] est donc raisonnable à appliquer dans notre cas.

Suite à ces remarques d'ordre qualitatif, nous présentons maintenant les résultats de l'évaluation en cascade nous fournissant une validation quantitative.

4.3.3 Résultat de l'évaluation en cascade

Nous comparons ici les performances de trois systèmes de classification supervisée fondés sur des champs conditionnels aléatoires. Le système CRF a été entraîné sur le corpus *train* et testé sur le corpus *testb* : ces corpus correspondent aux jeux de données originaux de la campagne d'évaluation CoNLL-2003. Conformément au protocole d'évaluation en cascade, ces même corpus ont été enrichis par annotation conceptuelle avant l'entraînement et le test des systèmes CRF_ACF_t et CRF_ACF_{ta}. Le système CRF_ACF_t emploie un treillis construit à partir du corpus *train* pour l'annotation conceptuelle des corpus *train* et *testb* alors que le treillis du système CRF_ACF_{ta} est constitué avec les corpus *train* et *testa*. Le système CRF constitue une référence dont les performances sont comparées aux systèmes CRF_ACF_t et CRF_ACF_{ta}.

Le tableau 4.2 présente les résultats de l'évaluation en cascade sur le corpus de test (*testb*) pour ces trois systèmes. Les colonnes représentent les différents types d'entités nommées et les lignes correspondent aux scores de précision, rappel et $F_{\beta=1}$ pour chaque système.

Système CRF	LOC	MISC	ORG	PER	Moyenne
Précision	84.37%	76.85%	77.55%	80.93%	80.62%
Rappel	83.81%	59.12%	71.52%	83.74%	77.11%
$F_{\beta=1}$	84.09	66.83	74.41	82.31	78.82
Système CRF_ACF_t					
Précision	86.51%	77.31%	81.32%	83.41%	82.95%
Rappel	84.95%	72.79%	74.95%	86.15%	81.07%
$F_{\beta=1}$	85.72 %	74.98	78.01	84.76	82.00
Système CRF_ACF_{ta}					
Précision	86.47%	77.34%	80.72%	83.87%	82.87%
Rappel	85.07%	72.93%	75.14%	86.15%	81.12%
$F_{\beta=1}$	85.77	75.07	77.83	84.99	81.98

TABLE 4.2 – Résultats de l'évaluation en cascade sur le corpus *testb*

Pour le système CRF_ACF_t, tous les scores ont augmenté par rapport au système CRF. La différence des moyennes indique une progression moyenne de 2.33% en précision, 3.96% en rappel et 3.18 pour $F_{\beta=1}$. Cette amélioration est statistiquement significative car les tests de Wilcoxon ($p_w < 0.008$) et de Student ($p_s < 0.042$) donnent des probabilités faibles pour l'hypothèse H_0. Les progressions apportées par le système CRF_ACF_{ta} sont comparables ($p_w < 0.008$ et $p_s < 0.041$). Les systèmes de classification d'entités nommées s'appuyant sur une annotation conceptuelle apportent donc des améliorations, légères mais significatives, sur l'ensemble des performances. En revanche, la comparaison entre les systèmes CRF_ACF_t et CRF_ACF_{ta} montre que la taille du treillis influence peu les résultats : les différences entre les deux systèmes n'est pas significative ($p_w > 0.44$ et $p_s > 0.73$ pour l'hypothèse H_0). Les scores obtenus pour les deux systèmes sont sensiblement les mêmes alors que l'annotation conceptuelle produite par CRF_ACF_{ta} pourrait être de meilleure qualité étant donné qu'elle tire parti d'un treillis plus volumineux.

Système CRF	LOC	MISC	ORG	PER	Moyenne
Précision	85.27%	87.30%	81.91%	85.60%	84.90%
Rappel	83.83%	69.31%	75.99%	86.48%	80.63%
$F_{\beta=1}$	84.55	77.27	78.84	86.04	82.71
Système CRF_ACF_t					
Précision	88.35%	88.01%	84.02%	87.54%	86.80%
Rappel	87.10%	78.85%	79.19%	88.87%	85.29%
$F_{\beta=1}$	87.72	83.18	81.54	88.20	86.04

TABLE 4.3 – Résultats de l'évaluation en cascade sur le corpus *testa*

L'évaluation des systèmes CRF et CRF_ACF_t sur le corpus *testa* (*cf.* tableau 4.3) montre une plus grande progression sur le corpus *testa* que sur *testb*. Le corpus *testa* est réputé pour avoir présenté de meilleurs résultats lors de la campagne CoNLL-2003. Les différences observées entre les systèmes CRF et CRF_ACF_t sont ici encore statistiquement significatives ($p_w < 0.008$ et $p_s < 0.011$).

Les différents résultats de ces évaluations en cascade montrent que les connaissances issues d'un treillis de Galois permettent d'améliorer les performances d'un système de classification d'entités nommées de manière significative. Nous suspectons cependant que des dégradations pourraient être observées avec des treillis beaucoup plus volumineux. Nous justifions maintenant cette dernière remarque en montrant les limites de notre adaptation de l'évaluation en cascade.

4.3.4 Problèmes liés à l'évaluation en cascade

Dans le cas d'une annotation conceptuelle, l'algorithme de classification supervisée (en l'occurrence, les CRF) n'a pas connaissance de la structure de treillis. Le contenu sémantique des concepts formels produits n'est pas non plus pris en compte. Pour le moment, l'annotation conceptuelle d'une unité lexicale correspond à un identifiant numérique associé au concept employé pour l'annotation. Le choix de cet identifiant est arbitraire[6] et ne tient compte ni de la structure, ni de la sémantique des concepts produits par l'ACF. En réalité, les objets et les attributs d'un concept formel constituent des variables qu'un classifieur supervisé devrait considérer.

La structure hiérarchique du treillis n'est pas non plus prise en compte. Deux concepts c_a et c_b ayant une relation de parenté ou de voisinage (parents communs)

6. Dans le programme que nous avons mis en œuvre, les identifiants des concepts indiquent l'ordre dans lequel ils ont été créés lors de la construction du treillis.

ont une chance non négligeable de partager des ensembles de variables proches pour la classification supervisée. Cependant, si le classifieur supervisé ne rencontre que c_a pendant l'apprentissage, c_b ne pourra pas être exploité lors de la phase de test.

Par ailleurs, l'architecture séquentielle de la cascade impose la manipulation d'une structure figée qui ne permet pas d'exploiter les propriétés incrémentales du treillis. Les identifiants des concepts formels peuvent évoluer lorsque l'on intègre de nouvelles relations. Par conséquent, le classifieur supervisé ne peut pas considérer de nouveaux concepts pourtant en relation de subsomption avec des concepts connus par le classifieur.

Il serait intéressant que l'algorithme d'apprentissage supervisé puisse comprendre la structure de treillis. Le langage des exemples du classifieur supervisé pourrait ainsi évoluer pour manipuler les ensembles d'objets et d'attributs constituant les concepts. Une autre possibilité serait de convertir les concepts formels vers un langage des exemples compréhensible pour l'algorithme d'apprentissage. Les solutions proposées dans le chapitre 5 vont dans ce sens.

4.4 Conclusion

Nous avons consacré ce quatrième chapitre à l'exploitation d'un treillis constitué à partir d'un corpus d'apprentissage pour annoter des UL extraites d'un corpus de test. Dans le processus d'annotation élaboré, les concepts formels sont considérés comme des unités de sens contextualisées qui désambiguïsent l'usage spécifique d'une UL en le généralisant à d'autres usages qui lui sont proches. La méthode d'annotation conceptuelle proposée fournit un enrichissement du corpus dont l'intérêt est valorisé à travers une tâche de classification supervisée d'EN, dans le cadre d'une évaluation en cascade [Candillier *et al.*, 2006].

Notre technique d'annotation conceptuelle contribue à faire progresser notre problématique d'exploitation d'une base lexicale acquise automatiquement. Notre démarche a consisté à adapter aux treillis des méthodes de désambiguïsation connues exploitant habituellement des ressources lexicales électroniques préexistantes. L'approche développée revient d'une certaine manière à choisir un niveau de granularité approprié dans une BL en mettant à profit la complémentarité des aspects symboliques et numériques pour déterminer un ensemble de concepts candidats et y sélectionner le plus représentatif. Ce choix peut être aussi appréhendé comme un problème de classification analogue à l'inférence d'hypothèses vérifiant un compromis spécifique/général dans l'espace des versions [Mitchell, 1997].

Nous avons validé l'intérêt des treillis pour notre problématique de désambiguïsation en évaluant le gain apporté par l'annotation conceptuelle pour une tâche de classification d'EN. Bien que valorisée par l'expérimentation, la procédure en

cascade n'exploite pourtant pas au mieux les propriétés des treillis. Il est notable que les différentes granularités de la BL sont mal considérées, surtout lorsque sa taille est importante : un nombre de concepts élevé réduit les possibilités de généralisation pour la phase d'apprentissage supervisé exploitant l'annotation conceptuelle. Par ailleurs, en imposant de juger une BL figée, l'architecture séquentielle de la cascade limite fortement la prise en compte de l'incrémentalité de la BL : une séquence de processus lourde doit être relancée pour évaluer l'influence d'une nouvelle intégration dans la BL sur les performances en classification supervisée. Nous avons donc validé la pertinence de l'annotation conceptuelle mais nous devons encore améliorer notre protocole d'évaluation pour mieux observer le gain apporté par notre approche.

Une alternative à cette architecture en cascade est présentée dans le cinquième et dernier chapitre. La procédure de classification supervisée est appliquée aux concepts du treillis plutôt qu'au corpus enrichi par une annotation conceptuelle à partir du treillis. L'étiquetage des concepts peut alors être propagé à de nouvelles EN observées dans un flux lors de l'intégration de ces EN au treillis. L'incrémentalité du treillis est alors exploitée pour que l'on puisse facilement examiner l'influence de la structure du treillis étiqueté à partir d'un corpus d'apprentissage sur la classification d'EN d'un corpus de test.

Chapitre 5

Propagation d'étiquettes sémantiques dans une base lexicale en construction

Dans le chapitre précédent, nous avons proposé une procédure de désambiguïsation reposant sur des apprentissages et des classifications réalisées successivement. Elle est amorcée par la construction du treillis, se poursuit avec une annotation conceptuelle et s'achève par une classification supervisée (dans le cadre d'une évaluation en cascade). Cette architecture séquentielle présente plusieurs limites. Une perte d'information marque chaque étape de la procédure, notamment entre l'annotation conceptuelle et la classification supervisée à cause de modes de représentation incompatibles ne permettant notamment pas de tirer profit de la granularité variable modélisée par le treillis. Par ailleurs, des problèmes calculatoires peuvent survenir : la construction d'un treillis volumineux nécessite des traitements lourds lors d'un apprentissage ou d'une classification supervisée réalisée *a posteriori*. Enfin d'un point de vue pratique, cette architecture séquentielle manque de flexibilité : les procédures exécutées après la construction du treillis doivent l'être à nouveau si on souhaite modifier les connaissances de ce dernier.

Pour dépasser les contraintes de notre architecture en cascade, nous proposons de réaliser de manière incrémentale et simultanée la construction d'un treillis et la classification supervisée d'EN (en relation avec leurs dépendants) observées dans un flux. La stratégie développée ici emploie un ensemble d'apprentissage L (L pour *labelled*) représentant les entités nommées étiquetées d'un corpus d'apprentissage et un autre ensemble U (U pour *unlabelled*) associé aux EN qui apparaissent non étiquetées dans le flux. À partir de données de L, nous construisons un treillis $\underline{\mathfrak{T}}$ dont les concepts formels sont enrichis avec les étiquettes des EN de l'ensemble L. Une classification supervisée des nouvelles EN de U peut ensuite être réalisée à l'aide de l'étiquetage des concepts enrichis de $\underline{\mathfrak{T}}$ auxquels ces EN

se rattachent lors de leur intégration dans $\underline{\mathfrak{T}}$. Par ailleurs, les étiquettes attachées aux concepts de $\underline{\mathfrak{T}}$ peuvent être employées pour la classification supervisée d'EN d'un corpus de test lors d'une annotation conceptuelle de ces EN avec $\underline{\mathfrak{T}}$.

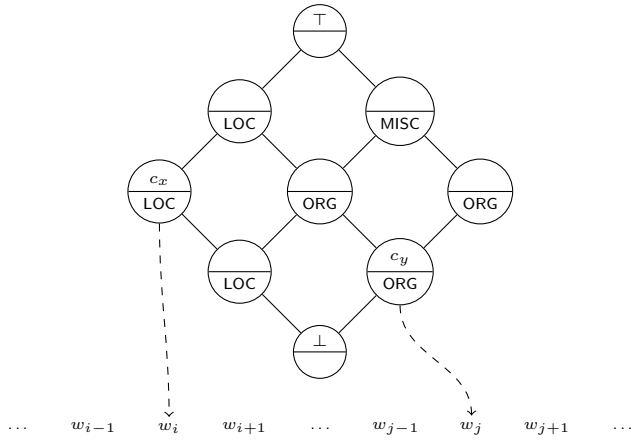


FIG. 5.1 – Annotation conceptuelle d'unités lexicales à l'aide d'un treillis de Galois disposant de concepts formels étiquetés

La figure 5.1 illustre la notion de treillis étiqueté. Les concepts formels de ce treillis sont symbolisés par des cercles dont la partie inférieure contient une étiquette (LOC, MISC ou ORG) et la partie supérieure est consacrée à leur identification (par exemple c_x, c_y ou $\top$). La figure 5.1 montre aussi une suite $\ldots w_{i-1}, w_i, w_{i+1} \ldots w_{j-1}, w_j, w_{j+1} \ldots$ qui représente une séquence d'unités lexicales. L'annotation conceptuelle des unités lexicales w_i et w_j est ici représentée par les flèches partant des concepts c_x et c_y. Les étiquettes portées par ces concepts sont utilisées pour la classification supervisée des unités lexicales w_i et w_j. Notre technique d'annotation conceptuelle peut ainsi être évaluée sur une tâche de classification d'EN d'un corpus de test en employant un treillis dont les concepts sont étiquetés à l'aide d'un corpus d'apprentissage.

Ce chapitre est consacré à la mise au point de techniques de classification supervisée permettant d'étiqueter l'ensemble des concepts formels d'un treillis ainsi que des exemples (entités nommées en relation avec leurs dépendants) issus d'un corpus qui s'intègrent à ce treillis. Notons que cette classification des concepts est considérée ici comme un enrichissement du treillis *a posteriori* et n'en affecte pas la structure qui reste déterminée sans supervision. De manière à limiter au maximum la supervision nécessaire pour étiqueter les concepts, nous adoptons une stratégie d'apprentissage semi-supervisé qui consiste à propager l'étiquetage connu d'un petit nombre d'exemples vers des données non étiquetées. Afin d'éti-

queter directement de nouveaux exemples issus d'un corpus lors de leur intégration dans le treillis, nous cherchons à coupler cette approche semi-supervisée à la construction du treillis dans un cadre incrémental.

Ce chapitre présente successivement deux techniques d'apprentissage semi-supervisé que nous adaptons au cadre des treillis de Galois. La première s'appuie sur l'utilisation d'un algorithme de co-apprentissage [Blum et Mitchell, 1998] (de l'anglais *co-training*) que nous utilisons pour propager la classification d'éléments connus à travers la structure de graphe biparti associé au contexte formel $\mathbb{K}$. La seconde emploie l'algorithme *label propagation* [Zhu et Ghahramani, 2002] qui s'appuie sur la structure du treillis ainsi que sur les distances entre concepts formels (*cf.* espace sémantique continu des concepts formels, chapitre 3). La validation de ces stratégies est proposée à travers des expérimentations sur corpus montrant l'évolution de leurs performances en fonction du nombre d'exemples (étiquetés et non étiquetés) utilisés pour une tâche de classification supervisée d'EN.

5.1 Classification de concepts par co-apprentissage

Le co-apprentissage (ou *co-training*) fait référence à une famille d'algorithmes semi-supervisés capables de propager l'étiquetage d'un ensemble d'apprentissage restreint vers un jeu de données non étiquetées. La technique incrémentale initialement proposée par Blum et Mitchell [1998] laisse envisager un couplage avec la construction du treillis à partir de données présentées en séquences. Par ailleurs, l'algorithme de co-apprentissage et notre méthode de construction de treillis sont capables de manipuler une représentation des données commune. Les objets et les attributs des concepts formels d'un treillis correspondent à des variables manipulées par l'algorithme de co-apprentissage. L'approche développée ici consiste à étiqueter des concepts formels d'un treillis construit progressivement à partir de données traitées à l'aide d'un algorithme de co-apprentissage.

Notre approche est appliquée à un cadre de classification supervisée d'EN. Notre démarche générale est la suivante. Nous disposons d'un corpus d'apprentissage qui comporte un ensemble restreint d'exemples étiquetés L (EN étiquetées PER, LOC... en relation avec leurs dépendants) et un ensemble d'exemples non étiquetés U. Les données étiquetées sont alors utilisées pour construire un treillis et initialiser l'algorithme de co-apprentissage. L'algorithme incrémental de co-apprentissage réalise alors séquentiellement la classification supervisée de nouveaux exemples issus de U. Chaque fois qu'un nouvel exemple est étiqueté, celui-ci est directement intégré au treillis. L'étiquetage de ces nouveaux exemples peut alors être transmis aux concepts lors de l'intégration de ces exemples au treillis. Notre technique d'annotation conceptuelle peut ensuite être testée sur tâche de

classification d'EN d'un corpus de test en employant les concepts étiquetés du treillis.

Cette section comporte quatre parties. Pour commencer, nous examinons le fonctionnement de l'algorithme de co-apprentissage proposé par Blum et Mitchell [1998]. Nous décrivons ensuite comment étiqueter des concepts formels en s'appuyant sur un modèle prédictif entraîné par co-apprentissage. Nous expérimentons alors l'algorithme initial de co-apprentissage pour le comparer à notre approche exploitant des concepts formels étiquetés. Pour terminer, nous discutons des limites et des perspectives de notre approche.

5.1.1 Co-apprentissage à partir de données fournies en séquence

Le co-apprentissage repose sur l'idée que des apprentissages indépendants peuvent être combinés sur des données décrites selon des points de vue complémentaires. Dans le cas de deux points de vue complémentaires, les exemples d'apprentissage sont décrits par deux ensembles de variables disjoints appelés « vues ». L'idée du co-apprentissage est que deux classifieurs entraînés séparément sur ces deux vues doivent étiqueter de manière identique la même donnée. Ces classifieurs peuvent alors être entraînés indépendamment sur chaque vue d'un ensemble de données étiquetées L pour étiqueter les exemples d'un ensemble de données non étiquetées U. En principe, ces classifieurs doivent s'améliorer entre eux en déplaçant de U vers L les exemples qui ont obtenu un étiquetage fiable avec ces classifieurs.

5.1.1.1 Bipartition des données d'apprentissage

Comme nous l'avons déjà indiqué, le co-apprentissage manipule des exemples décrits par deux ensembles de variables disjoints appelées « vues ». Ces exemples peuvent être représentés par un graphe biparti[1] tel que celui qui représente les données issues de nos corpus (*cf.* figure 2.4 au chapitre 2). Nous conservons ici la terminologie de l'ACF avec une vue sur les objets O et une vue sur les attributs A interconnectés par une relation $r \in R$. Comme l'illustre la figure 5.2 dans notre cas particulier, l'ensemble d'objets $O = \{o1, o2 \cdots om\}$ correspond à une vue sur les entités nommées et l'ensemble d'attributs $A = \{a1, a2 \cdots an\}$ à une autre vue sur leurs dépendants. Un exemple (o, a) exploitable pour un co-apprentissage est donc constitué d'une entité nommée (vue sur les objets) en relation avec un de

1. La bipartition de l'ensemble des variables n'est pas toujours naturelle. Cependant, selon Nigam et Ghani [2000], un ensemble de variables peut être décomposé en deux partitions qui minimisent une mesure d'information mutuelle conditionnelle.

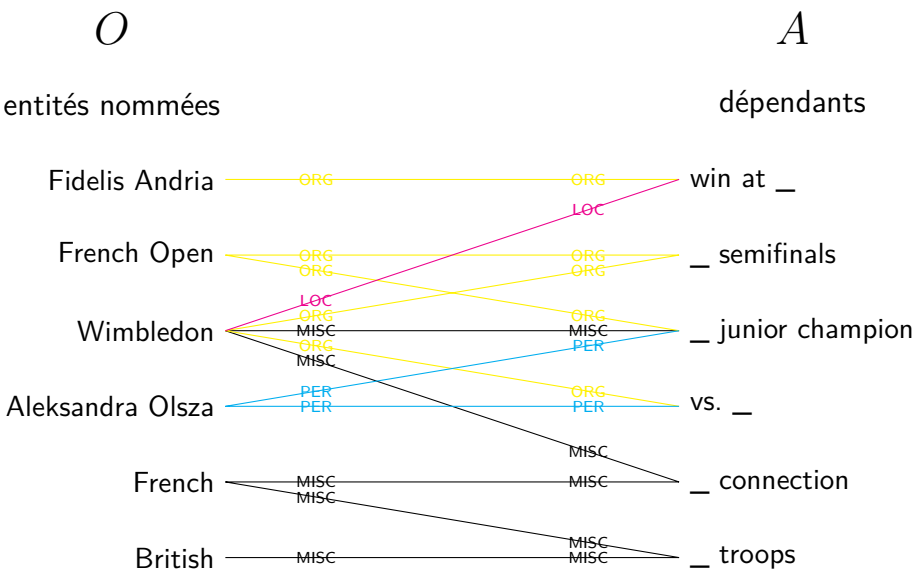


FIG. 5.2 – Graphe biparti constitué de relation étiquetées entre des EN et leurs dépendants

ses dépendants [2].

L'ensemble des données d'apprentissage est divisé en deux ensembles de relations : les relations $L = \{((o_1, a_1), l_1) \dots ((o_{|L|}, a_{|L|}), l_{|L|})\}$ annotées par des étiquettes $l_i \in Labels$, et les relations non étiquetées $U = \{(o_{|L|+1}, a_{|L|+1}) \dots (o_{|L|+|U|}, a_{|L|+|U|})\}$. Sur le graphe de la figure 5.2, l'étiquetage des relations de L est marqué aux extrémités des liens entre les entités nommées et leurs dépendants. Des couleurs symbolisent aussi cet étiquetage : cyan pour les personnes, magenta pour les lieux, jaune pour les organisations et noir pour les autres types d'entités nommées.

5.1.1.2 Hypothèse d'indépendance pour des apprentissages sur des vues séparées

Représentés par un graphe biparti, les exemples d'apprentissage de L sont exploités pour entraîner un modèle prédictif $\mathcal{M}_{cotrain}$ à l'aide d'un algorithme de co-apprentissage. Le modèle $\mathcal{M}_{cotrain}$ peut être vu comme une fonction $h : (O \times A) \rightarrow Labels$ prédisant l'étiquette d'un exemple (o, a). Dans le cadre du co-apprentissage, cette fonction h combine les estimations fournies par deux fonctions $h_o : O \rightarrow Labels$ et $h_a : A \rightarrow Labels$ entraînées indépendamment sur les

2. Notons qu'il est tout à fait envisageable de considérer plusieurs dépendants : ceux situés à gauche ou à droite de l'entité nommée et éventuellement ceux observés dans l'ensemble des phrases d'une dépêche.

objets et les attributs de L. Dans une perspective probabiliste, il s'agit d'estimer indépendamment $h_o(o) = \underset{l \in Labels}{\operatorname{argmax}} P(l|o)$ et $h_a(a) = \underset{l \in Labels}{\operatorname{argmax}} P(l|a)$ plutôt que de calculer directement $h((o,a)) = \underset{l \in Labels}{\operatorname{argmax}} P(l|o,a)$. Ces estimations reposent sur une hypothèse d'indépendance entre deux variables aléatoires o et a (*cf.* formule 5.1) qui stipule que la probabilité jointe $P(o,a|l)$ est égale au produit de probabilités $P(o|l)P(a|l)$:

$$\forall o \in O, \forall a \in A, \forall l \in Labels, P(o,a|l) = P(o|l)P(a|l) \tag{5.1}$$

Cette hypothèse d'indépendance et le théorème de Bayes permettent d'écrire la probabilité $P(l|o,a)$ de la façon suivante.

$$\begin{aligned}
P(l|o,a) &= P(l) \quad \frac{P(o,a|l)}{P(o,a)} && (5.2)\\
&= P(l) \quad \frac{P(o|l)}{P(o)}\frac{P(a|l)}{P(a)} && (5.3)\\
&= P(l) \quad \frac{P(l|o)P(o)}{P(l)P(o)}\frac{P(l|a)P(a)}{P(l)P(a)} && (5.4)\\
&= \tfrac{1}{P(l)} \quad P(l|o)P(l|a) && (5.5)
\end{aligned}$$

Selon Blum et Mitchell [1998], l'étiquette $l \in Labels$ d'un exemple (o,a) est celle qui maximise le produit des probabilités $P(l|o)$ et $P(l|a)$, estimées par deux classifieurs entraînés indépendamment sur chaque vue. Les auteurs font l'hypothèse implicite que les classes $l \in Labels$ sont équiprobables *a priori* ($\forall l_i, l_j \in Labels, P(l_i) = P(l_j)$), ce qui implique que le terme $\frac{1}{P(l)}$ de l'équation 5.5 soit constant et qu'il puisse être supprimé pour la maximisation de $P(l|o,a)$:

$$\begin{aligned}
\hat{h}((o,a)) &= \underset{l \in Labels}{\operatorname{argmax}} \quad P(l|o,a) && (5.6)\\
\hat{h}((o,a)) &= \underset{l \in Labels}{\operatorname{argmax}} \quad P(l|o)P(l|a) && (5.7)
\end{aligned}$$

5.1.1.3 Algorithme de co-apprentissage

Dans le cadre d'un apprentissage semi-supervisé, seules les relations de l'ensemble L sont étiquetées. L'étiquetage partiel de l'ensemble des relations doit être propagé progressivement d'une vue vers l'autre à l'aide des sommets partagés par des relations étiquetées L et les relations non étiquetées U dans le graphe biparti. Cette propagation est réalisée par l'algorithme de co-apprentissage (*cf.* Algorithme 9).

Algorithme 9: Co-entraîner(L,U)

Entrées : U : le jeu de données non étiqueté
L : le jeu de données étiqueté
$U' = \emptyset$: ensemble d'exemples en attente d'une prédiction fiable
tant que $U \neq \emptyset$ **faire**
    Entraîner un classifieur h_o sur les objets de L
    Entraîner un classifieur h_a sur les attributs de L
    Déplacer des échantillons de U vers U'
    Étiqueter les objets de U' avec h_o
    Étiqueter les attributs de U' avec h_a
    Déplacer les exemples de U' obtenant les meilleures prédictions vers L
retourner h_o *et* h_a

En traitant progressivement les exemples non étiquetés, l'algorithme de co-apprentissage emprunte une procédure incrémentale. Pour chaque nouvel itération, deux classifieurs h_o et h_a sont entraînés indépendamment sur chaque vue des données d'apprentissage L. L'ensemble U' initialement vide contient des exemples de U étiquetés à l'aide de h_o et h_a. Les exemples de U' ayant les prédictions les plus fiables sont déplacés vers l'ensemble L alors que les autres exemples sont conservés jusqu'à ce que la fiabilité de leur étiquetage soit suffisante. La fiabilité d'une prédiction l pour un élément $x = o$ ou $x = a$ correspond à la probabilité $P(l|x)$. L'ordre des exemples examinés dans le flux n'est pas nécessairement le même que l'ordre des exemples étiquetés car le choix des exemples déplacés de U' vers L dépend de cette mesure de fiabilité. La structure de données incarnée par U' se comporte en fait comme une file d'attente contenant des exemples en demande d'une connexion fiable avec de futurs exemples étiquetés. Lorsqu'il n'existe plus d'exemples non étiquetés dans U, la procédure retourne deux classifieurs h_o et h_a.

Cette procédure incrémentale est compatible avec l'algorithme 5 (*cf.* chapitre 2) d'ajout d'une relation quelconque à un treillis. Il devient ainsi envisageable de construire conjointement le modèle de co-apprentissage et le treillis. Par ailleurs, les classifieurs h_o et h_a retournés manipulent les variables qui constituent les concepts formels, permettant ainsi d'étiqueter ces concepts.

5.1.2 Co-classification de concepts dans un treillis en construction

Nous décrivons ici notre modèle de classification supervisée des concepts formels capable de tirer parti d'estimations acquises avec deux classifieurs co-entraînés. Cette classification des concepts formels présente plusieurs intérêts. En premier lieu, l'enrichissement de la représentation des concepts avec les étiquettes issues

d'un corpus d'apprentissage unifie les procédures d'annotation conceptuelle (non supervisé) et d'étiquetage supervisé en produisant une double annotation des EN. En effet, une classification supervisée d'une unité lexicale est réalisée lorsqu'un concept lui transmet son étiquette. D'autre part, dans son langage des exemples, la technique supervisée étudiée ici manipule directement les variables (objets et attributs) qui constituent les concepts, contrairement à une méthodologie (*cf.* évaluation en cascade, chapitre 4) où seuls des identifiants des concepts sont pris en compte. Nous évitons ainsi de perdre des informations lors de l'enrichissement pour un apprentissage supervisé par une classification non supervisée.

Nous présentons ici comment procéder à la construction incrémentale du treillis en y intégrant des relations étiquetées lors d'un co-apprentisage. Nous montrons alors une technique de classification de concepts formels combinant des classifications issues de deux modèles prédictifs co-entraînés.

5.1.2.1 Intégration de relations co-classifiées à un treillis

Nous cherchons ici à rendre compatibles l'algorithme de co-apprentissage et notre algorithme incrémental d'ajout de relations à un treillis (*cf.* algorithme 5, chapitre 2). Le graphe biparti manipulé en co-apprentissage correspond en fait à un contexte formel dont certaines relations sont associées à des étiquettes (en l'occurrence les étiquettes sémantiques attachées aux entités nommées). Pour nous situer dans un cadre d'apprentissage semi-supervisé, nous pouvons distinguer le contexte formel $\mathbb{K}_L = (O_L, A_L, R_L)$ correspondant aux exemples étiquetés de L et le $\mathbb{K}_U = (O_U, A_U, R_U)$ qui est associé aux exemples non encore étiquetés de U. Considérons un treillis de Galois $\underline{\mathfrak{T}}_{\mathbb{K}_L}$ qui représente les exemples de l'ensemble L. Les relations R_L correspondent aux exemples étiquetés de L employés à l'initialisation de l'algorithme de co-apprentissage. Rappelons que l'étiquetage des relations de R_L n'influence pas la structure de $\underline{\mathfrak{T}}_{\mathbb{K}_L}$. À chaque cycle de co-apprentissage, l'ensemble L est augmenté de nouveaux exemples provenant des données étiquetées U. Plus précisément, la ligne 8 de l'algorithme 5 met en œuvre une sélection de relations (o, a) dans l'ensemble U' qui obtiennent les meilleures prédictions pour les déplacer vers l'ensemble des relations étiquetées L. Notre démarche consiste simplement à ajouter au treillis chacune des relations (o, a) sélectionnées par un appel à l'algorithme $\underline{\mathfrak{T}}.add_relation(o, a)$. La construction du treillis et le co-apprentissage peuvent ainsi être réalisés conjointement selon un mode incrémental. Il est important de noter que la construction du treillis n'a aucune influence sur le déroulement du co-apprentissage. En revanche, les concepts formels de ce treillis sont employés pour l'annotation conceptuelle des EN du corpus de test. En fait les classifieurs h_o et h_a mis à jour à chaque itération de l'algorithme permettent d'étiqueter tous les concepts formels d'un treillis en cours de construction. Nous devons à présent préciser comment étiqueter les

concepts à l'aide de ces classifieurs.

5.1.2.2 Classifieur bayésien naïf pour l'étiquetage d'un concept formel

Nous représentons maintenant les concepts formels dans un langage des exemples compréhensible par deux classifieurs co-entraînés. Soit $c = (E, I)$, un concept formel tel que $E = \{o_0, o_1 \dots o_m\}$ et $I = \{a_0, a_1 \dots a_n\}$. Les deux ensembles O et A du contexte formel $\mathbb{K}$ sont supposés conditionnellement indépendants, étant donnée une classe cible $l \in Labels$ (*cf.* hypothèse d'indépendance formulée en 5.1).

Les intensions et les extensions des concepts constituent deux vues indépendantes fournissant des variables à deux classifieurs entraînés conjointement par un algorithme de co-apprentissage. En considérant qu'un concept c est décrit par un ensemble de variables indépendantes $X = \text{ext}(c) \cup \text{int}(c) = \{x_1 \dots x_m\}$, nous pouvons estimer la probabilité d'une étiquette $l \in Labels$ sachant X à l'aide d'un classifieur bayésien naïf :

$$P(l|X) = P(l)\frac{\prod_i P(x_i|l)}{P(x_1 \dots x_m)} \tag{5.8}$$

Pour des questions calculatoires, nous reformulons $P(l|X)$ à l'aide du théorème de Bayes et de l'hypothèse d'indépendance 5.1 :

$$P(l|X) = P(l) \quad \frac{\prod_i P(x_i|l)}{\sum_{l_j \in Labels} P(x_1 \dots x_m, l_j)} \tag{5.9}$$

$$= P(l) \quad \frac{\prod_i P(x_i|l)}{\sum_{l_j \in Labels} P(x_1 \dots x_m|l_j)P(l_j)} \tag{5.10}$$

$$= P(l) \quad \frac{\prod_i P(x_i|l)}{\sum_{l_j \in Labels} \prod_i P(x_i|l_j)P(l_j)} \tag{5.11}$$

L'étiquette $l \in Labels$ associée à un concept est celle qui maximise $P(l|X)$:

$$h(c) = \underset{l \in Labels}{\text{argmax}}\ P(l|X) \tag{5.12}$$

Le treillis représenté en figure 5.3 montre un exemple simple de concepts formels disposant d'une classification probabiliste. Les concepts situés entre $\top$ et $\bot$

disposent d'une classification probabiliste indiquée par des proportions (en pourcentages) pour chaque étiquette (PER, LOC, ORG, MISC). La figure 5.3 montre également une séquence d'unités lexicales $\cdots w_{i-1}, w_i, w_{i+1} \cdots w_{j-1}, w_j, w_{j+1} \cdots$ dans laquelle w_i et w_j sont annotées par deux concepts du treillis.

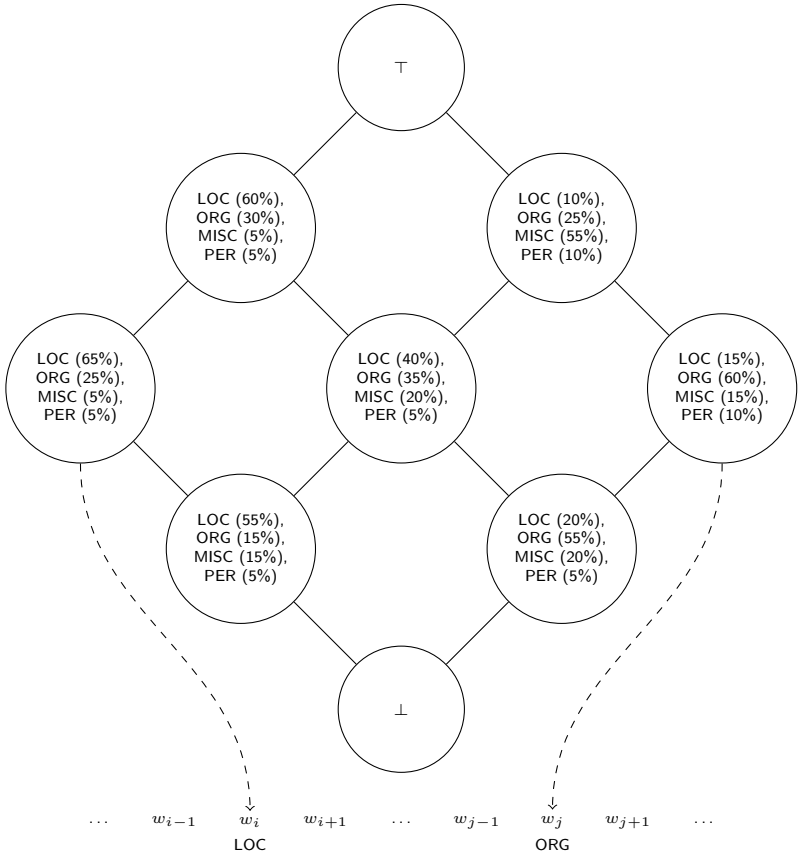


FIG. 5.3 – Annotation conceptuelle d'unités lexicales à l'aide d'un treillis de Galois disposant de concepts formels étiquetés

Ces deux concepts transmettent leur étiquette majoritaire (LOC et ORG) à w_i et w_j. Selon ce principe, un treillis de Galois disposant de concepts étiquetés peut être exploité en classification supervisée. Notre nouveau classifieur peut maintenant être évalué en ayant recours à des expérimentations sur corpus.

5.1.3 Expérimentations et résultats

Les expérimentations réalisées ici sont effectuées sur la tâche de classification supervisée d'entités nommées présentée dans le chapitre 4. Notre objectif est de vérifier l'intérêt de notre stratégie exploitant des concepts formels étiquetés pour une tâche de classification d'EN. Les jeux de données utilisés sont les mêmes que pour les expérimentations précédentes : le corpus *train* sert pour la phase d'entraînement d'un classifieur destiné à étiqueter des EN alors que *testb* est employé pour l'évaluation de ce classifieur. Le nombre d'exemples d'apprentissage disponibles dans le corpus *train* est $|D| = 31755$.

Nous présentons ici successivement deux séries d'expérimentations. À titre de référence, la première permet de comparer les résultats de deux stratégies d'apprentissage supervisé employées sur l'ensemble des données étiquetées du corpus *train*. Ces données étiquetées sont fournies progressivement à un classifieur bayésien naïf (*cf.* équation 5.7). Elles servent également à construire un treillis dont les concepts sont étiquetés par le classifieur bayésien naïf. Nous étudions alors l'influence du nombre d'exemples étiquetés $|L|$ sur les performances de ce classifieur bayésien naïf et de notre approche utilisant des concepts formels étiquetés. La seconde série d'expérimentations évalue deux stratégies semi-supervisées de co-apprentissage. Le nombre d'exemples étiquetés $|L|$ est fixé à 5000 et le nombre d'exemples non étiquetés $|U|$ est compris dans l'intervalle $[0, |D| - |L|]$. Les exemples de U sont progressivement étiquetés par l'algorithme de co-apprentissage et sont également intégrés au treillis $\underline{\mathfrak{T}}$. Nous comparons alors les performances de l'algorithme original de co-apprentissage [Blum et Mitchell, 1998] avec notre approche qui emploie des concepts formels étiquetés.

5.1.3.1 Performances des stratégies supervisées

Nous cherchons ici à établir des résultats de référence en classification supervisée pour permettre une comparaison ultérieure avec différentes stratégies semi-supervisées. Nous utilisons uniquement des exemples étiquetés de L en faisant varier leur nombre $|L|$ dans l'intervalle $[0, |D|]$, où $|D| = 31755$ est le nombre d'exemples disponibles dans le corpus *train* : il n'est donc pas question d'exploiter des exemples non étiquetés ici. Nous proposons de comparer les performances d'un classifieur bayésien naïf (BN) avec celles d'un classifieur conceptuel (BNCF) qui emploie des concepts formels étiquetés. Cette comparaison nous permet de vérifier la validité d'une classification supervisée dirigée par une annotation conceptuelle d'EN.

Les données étiquetées du corpus *train* sont fournies progressivement à un classifieur bayésien naïf (BN). Ces mêmes données servent à la construction incrémentale d'un treillis dont les concepts sont étiquetés par le classifieur BN. Lors d'une annotation conceptuelle, le classifieur BNCF exploite ces concepts pour

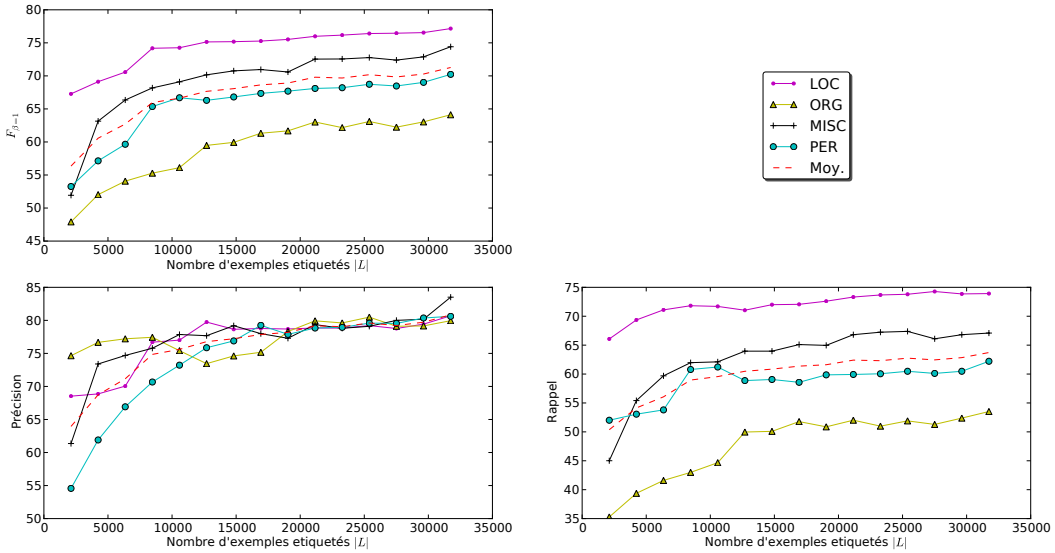


FIG. 5.4 – Évolution des performances du classifieur bayésien naïf (BN) en fonction du nombre d'exemples étiquetés |L| sur le corpus *testb*

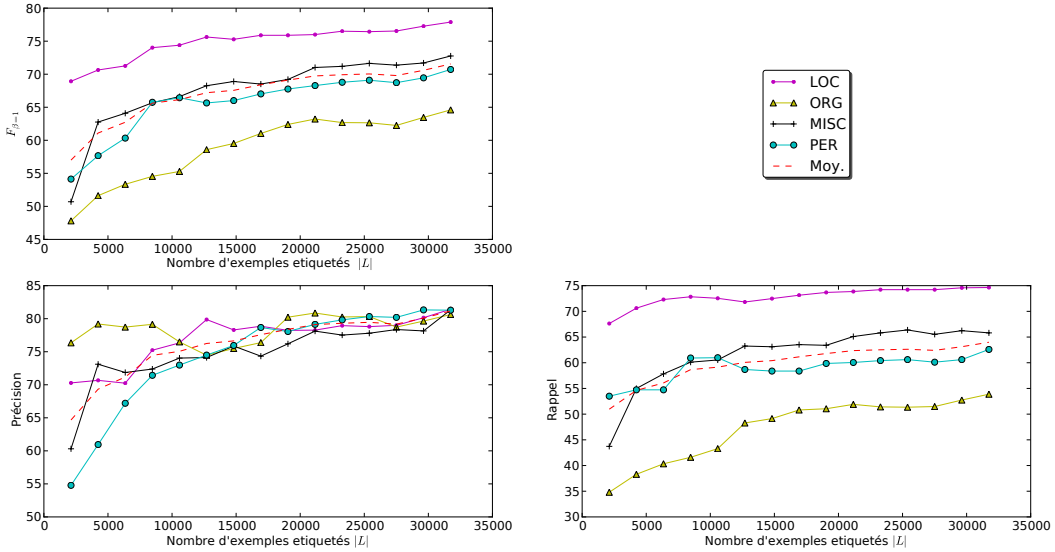


FIG. 5.5 – Évolution des performances du classifieur conceptuel (BNCF) en fonction du nombre d'exemples étiquetés |L| sur le corpus *testb*

étiqueter les EN du corpus de test. Nous étudions alors l'influence du nombre d'exemples étiquetés $|L|$ sur les performances de ce classifieur bayésien naïf et de notre approche utilisant des concepts formels étiquetés. Les figures 5.4 et 5.5 présentent les performances des stratégies BN et BNCF appliquées au corpus *testb*.

Les deux figures sont composées de trois graphiques représentant l'évolution de la précision, du rappel et de la F-mesure en fonction du nombre d'exemples étiquetés observés dans le corpus d'apprentissage *train*. Ces performances correspondent à une évaluation d'une tâche de classification d'EN sur le corpus *testb*. Les courbes de chaque graphique sont associées aux étiquettes sémantiques du corpus CoNLL. Nous avons également ajouté une courbe indiquant une moyenne globale (traits discontinus en rouge) pour l'ensemble des étiquettes.

BN	LOC	MISC	ORG	PER	Moyenne
Précision	80.69%	83.51%	79.95%	80.61%	80.84%
Rappel	73.92%	67.09%	53.52%	62.21%	63.72%
$F_{\beta=1}$	77.16	74.41	64.12	70.23	71.27
BNCF					
Précision	81.43%	81.34%	80.63%	81.29%	81.18%
Rappel	74.64%	65.81%	53.88%	62.59%	63.99%
$F_{\beta=1}$	77.89	72.76	64.6	70.72	71.56

TABLE 5.1 – Comparaison des performances maximales obtenues par le classifieur bayésien naïf (BN) et le classifieur conceptuel (BNCF) évalués sur le corpus *testb*

Les figures 5.4 et 5.5 montrent que le comportement des deux classifieurs est très proche. Les deux systèmes s'améliorent légèrement à mesure que le nombre d'exemples étiquetés croît. Les performances maximales atteintes par les deux systèmes sont aussi très similaires. Comme l'indique le tableau 5.1, le classifieur conceptuel gagne seulement de 0.34% en précision, 0.27% en rappel et 0,29 en F-mesure par rapport au bayésien naïf. Cette amélioration n'est pas statistiquement significative d'après le test de Wilcoxon ($p_w > 0.742$) et test-t de Student ($p_s = 0.973$) pour lesquels la probabilité de l'hypothèse H_0 est élevée. En dehors des intérêts d'interprétabilité, il n'est donc pas clairement avantageux d'utiliser notre approche exploitant des concepts formels pour la classification supervisée.

Sur le même corpus d'apprentissage, les CRF (*cf.* chapitre 4) ont une F-mesure moyenne atteignant un score de 78,82 dépassant d'environ 7 points nos classifieurs BN et BNCF. Un rappel plus haut (77,11% pour les CRF contre 63,99% pour le classifieur BNCF) est à l'origine de cette différence alors que la précision reste

au même niveau (80,62% contre 81,18%). Les variables manipulées par les CRF, représentées notamment par des mots et des traits syntaxiques, sont beaucoup plus nombreuses que celles employées dans ce chapitre. Notre approche fondée sur les treillis exploite en effet des variables issues d'une analyse en dépendances (*cf.* chapitre 2) qui groupent souvent plusieurs mots au sein d'une seule variable et qui ne comportent aucun étiquetage morphosyntaxique. Par ailleurs, il faut noter que certaines entités nommées extraites apparaissent dans le corpus sans présenter de dépendances : elles sont par conséquent absentes des données d'apprentissage manipulées ici. Pour ces raisons, l'entraînement des CRF est pratiqué avec un plus grand nombre de variables, permettant ainsi de couvrir plus d'échantillons que nos méthodes sur le corpus de test. Les mêmes remarques pourront être exprimées pour les classifieurs co-entraînés dont nous présentons maintenant les résultats.

5.1.3.2 Performances des stratégies semi-supervisées

Pour cette seconde série d'expérimentations, nous comparons les performances atteintes par deux stratégies semi-supervisées sur une tâche de classification d'EN du corpus *testb*. Nous utilisons cette fois à la fois des exemples étiquetés et non étiquetés : le nombre d'exemples étiquetés $|L|$ est fixé à 5000 et le nombre d'exemples non étiquetés $|U|$ est compris dans l'intervalle $[0, |D| - |L|]$, où $|D| = 31755$ est le nombre d'exemples disponibles dans le corpus *train*. Les exemples de U sont fournis en séquence : ils sont progressivement étiquetés par l'algorithme de co-apprentissage et sont également intégrés au treillis $\underline{\mathfrak{T}}$ de manière incrémentale. Les concepts de ce treillis peuvent alors être étiquetés par les classifieurs co-entraînés à chaque itération de l'algorithme de co-apprentissage. Nous comparons alors les performances de l'algorithme original de co-apprentissage (appelé CT pour abréger *co-training*) proposé par Blum et Mitchell [1998] avec notre approche employant des concepts formels étiquetés (CTCF).

Nous cherchons ici à observer l'évolution des performances des classifieurs CT et CTCF en fonction du nombre d'exemples non étiquetés $|U|$ traités de manière séquentielle. Les figures 5.6 et 5.7 illustrent cette évolution pour la tâche de classification d'EN sur le corpus *testb*. Comme précédemment, chaque figure présente trois graphiques indiquant la précision, le rappel et la F-mesure pour chaque étiquette sémantique. Environ $\frac{1}{6}$ des exemples du corpus *train* (soit 5000 sur 31755) ont été étiquetés initialement. Le reste des échantillons est examiné progressivement par les deux algorithmes d'apprentissage.

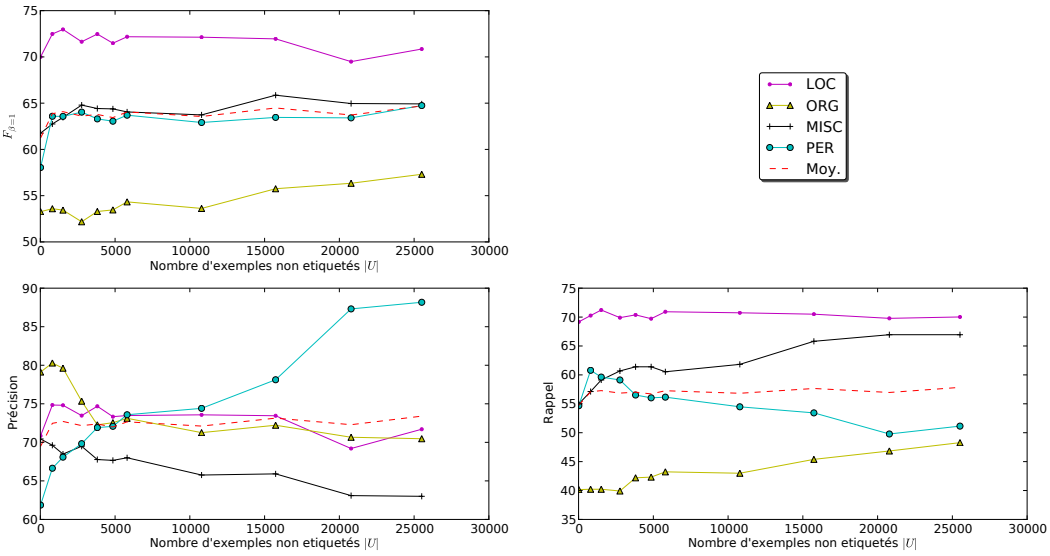


FIG. 5.6 – Évolution des performances de l'algorithme de co-apprentissage (CT) sur le corpus *testb* en fonction du nombre d'exemples non étiquetés $|U|$

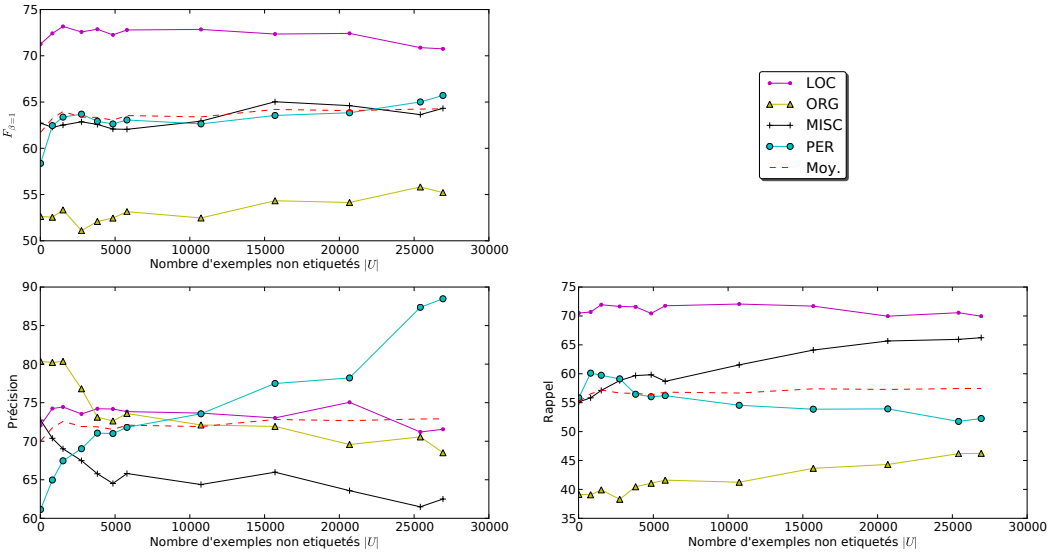


FIG. 5.7 – Évolution des performances du classifieur conceptuel co-entraîné (CTCF) sur le corpus *testb* en fonction du nombre d'exemples non étiquetés $|U|$

Les deux systèmes montrent de légères améliorations des résultats à mesure que des échantillons non étiquetés sont ajoutés. En revanche, certaines courbes

montrent des dégradations difficilement comblées par l'ajout de nouveaux exemples non étiquetés.

Comme l'indique la table 5.2, les performances des deux classifieurs sur l'ensemble des données sont analogues. Notre classifieur conceptuel montre des résultats légèrement inférieurs à ceux de l'algorithme original de co-apprentissage. La différence observée n'est cependant pas statistiquement significative ($p_w = 0.250$ et $p_s > 0.232$).

CT	LOC	MISC	ORG	PER	Moyenne
Précision	71.7%	63.0%	70.47%	88.17%	73.4%
Rappel	70.02%	66.95%	48.28%	51.14%	57.84%
$F_{\beta=1}$	70.85	64.92	57.31	64.74	64.7
CTCF					
Précision	71.55%	62.5%	68.51%	88.48%	72.9%
Rappel	69.96%	66.24%	46.24%	52.26%	57.45%
$F_{\beta=1}$	70.75%	64.32	55.21	65.71	64.26

TABLE 5.2 – Comparaison des performances de l'algorithme de co-apprentissage (CT) et du classifieur conceptuel co-entraîné (CTCF) sur le corpus *testb*

Nous pouvons noter par ailleurs que la classification probabiliste des concepts formels met en évidence des concepts dont l'étiquetage est hétérogène : si certains concepts correspondent à des hypothèses de regroupements consistantes avec les données d'apprentissage, c'est-à-dire des regroupements sémantiquement homogènes vis-à-vis de l'étiquetage des EN du corpus, d'autres concepts regroupent à tort des EN de classes différentes. Par exemple, le concept c_x = ({*Wimbledon*, *Fidelis Andria*},{*win at*}) comporte un lieu (*Wimbledon* dans le contexte de *win at* est étiqueté comme un lieu) et une organisation (*Fidelis Andria* est une équipe de football considérée comme une organisation). L'utilisation de concepts ambigus pour l'annotation conceptuelle risque de se répercuter sur la classification supervisée alors que des classifieurs bayésiens naïfs n'engendreraient pas nécessairement ce type d'erreurs. Par ailleurs, il n'est pas garanti que les concepts sélectionnés (sans supervision) pour l'annotation conceptuelle soient les meilleurs possibles pour optimiser la tâche supervisée.

En dehors du comportement de l'algorithme, nous avons remarqué que les attributs ayant un grand nombre d'occurrences dans le corpus (*stop-words*), tels que « said », « says », « the » ou « a », participent à l'amélioration des performances générales. Bien qu'ils ne soient pas spécifiquement en dépendance avec une étiquette sémantique particulière, ils permettent de connecter davantage de

composantes du graphe biparti et contribuent nettement à la propagation de l'étiquetage. Par ailleurs, certaines EN ne sont en relation qu'avec ces *stop-words* et les supprimer des données d'apprentissage et de test engendrerait une nette baisse des performances.

5.1.4 Discussion

Dans la pratique, le co-apprentissage présente un défaut majeur. Le modèle de classification construit par l'algorithme étiquette de nouveaux exemples grâce à ses prédictions, sans avoir la garantie que ces étiquettes soient correctes. Le co-apprentissage permet normalement de limiter ce problème lorsqu'il est comparé avec d'autres algorithmes de *bootstrapping* qui ne reposent pas sur l'hypothèse d'indépendance des vues. En pratique, des dégradations dans la qualité de données étiquetées lors du co-apprentissage font obstacle à des améliorations supplémentaires : sur le long terme, les classifieurs attachés à chaque vue manipulent des données de moins bonne qualité à chaque nouvelle itération. La propagation d'une seule erreur peut engendrer une nette baisse des performances après quelques itérations. La qualité des données étiquetées à chaque itération de co-apprentissage est cruciale pour la convergence de l'algorithme. Pour limiter ces dégradations, Pierce et Cardie [2001] suggèrent de corriger manuellement les sources d'erreur pendant les itérations du co-apprentissage tout en limitant au maximum le nombre d'exemples à annoter manuellement. Cette stratégie entre dans le cadre d'un apprentissage actif qui fait intervenir un utilisateur humain pour étiqueter manuellement les exemples les plus déterminants pour le déroulement de l'apprentissage.

Le recours à l'apprentissage actif intervient soit lorsque les données sont coûteuses à obtenir, soit au contraire lorsqu'elles sont trop abondantes et qu'il faut être capable de diriger l'apprentissage vers les informations pertinentes. Parmi les différentes stratégies actives existantes, l'échantillonnage sélectif [Roy et McCallum, 2001] s'intéresse à une partie restreinte des exemples d'apprentissage pour en déterminer les caractéristiques et en étiqueter un minimum. Pour [Muslea *et al.*, 2006], l'échantillonnage sélectif a pour objectif d'améliorer le modèle prédictif courant de manière itérative. Il s'applique à des tâches de classification pour lesquelles un grand nombre d'exemples non étiquetés sont accessibles. Dans ce scénario, l'apprenant actif demande à faire étiqueter certains exemples non étiquetés. Ces exemples sont sélectionnés notamment pour leur potentiel à améliorer un modèle prédictif. Le *co-testing* [Muslea *et al.*, 2006] est une technique d'échantillonage sélectif compatible avec le cadre du co-apprentissage dans laquelle les exemples non encore étiquetés sont choisis lorsque leur classification diverge entre les deux vues. Ces exemples sont désignés comme des points de contention, sources d'erreurs potentielles, parmi lesquels une sélection est nécessaire.

Le graphe biparti manipulé implicitement en co-apprentissage (par exemple le graphe de la figure 5.2) possède des propriétés structurelles intéressantes pour identifier les exemples les plus utiles à l'apprentissage. À cet égard, dans un cadre de classification d'entités nommées, Jones [2004, 2005] indique notamment que les caractéristiques liées à la connectivité de ces graphes (degrés des nœuds, coefficient de *clustering* dans les graphes « petit monde » [Watts et Strogatz, 1998]) sont utiles pour déterminer les exemples les plus influents lors de la propagation de l'étiquetage entre les exemples étiquetés et les non étiquetés. Cette propagation est privilégiée par la présence de composantes fortement connexes du graphe biparti [Jones, 2005]. Or, les concepts formels sont des bicliques maximales du graphe biparti et sont à ce titre assimilables à des composantes connexes du graphe biparti. Examiner les propriétés structurelles du graphe biparti et celles du treillis associé pourraient contribuer à mieux comprendre les mécanismes de propagation.

Nous devons rappeler qu'il n'est pas clairement avantageux d'utiliser notre approche exploitant des concepts formels étiquetés pour la classification supervisée. Notre stratégie de co-apprentissage doit être comparée avec d'autres techniques semi-supervisées. Par ailleurs, le co-apprentissage n'explicite pas suffisamment comment se propage un étiquetage des composantes étiquetées du graphe biparti vers les non étiquetées. En expliquant les mécanismes de propagation dans le treillis associé au graphe biparti, il nous serait plus évident d'identifier les exemples les plus influents et les sources d'erreurs. À cet égard, nous étudions dans la section suivante un autre algorithme semi-supervisé s'appuyant sur la structure hiérarchisée du treillis pour propager la classification de quelques exemples étiquetés vers l'ensemble des concepts formels non étiquetés.

5.2 Propagation d'étiquettes sémantiques dans un treillis

Cette section développe une alternative au co-apprentissage qui s'appuie sur les aspects structurels et continus des treillis (*cf.* chapitre 3). Nous examinons ici l'hypothèse que la structure du treillis est utile pour propager l'étiquetage connu de quelques exemples d'un ensemble d'apprentissage L vers des exemples non étiquetés d'un ensemble U. À cet égard, les relations d'héritage qui structurent un treillis seraient pertinentes pour propager l'étiquetage connu de concepts vers leurs concepts plus généraux ou plus spécifiques dont l'étiquetage est inconnu. À titre d'illustration, la figure 5.8 présente des concepts étiquetés (disques colorés) qui pourraient propager leur étiquetage à un concept non étiqueté (point d'interrogation) avec lequel ils partagent des variables (objets et attributs) étant donné qu'ils entretiennent une relation de parenté.

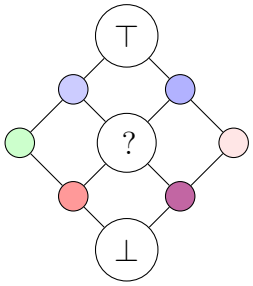


FIG. 5.8 – Treillis de Galois partiellement étiqueté

L'idée de propager un étiquetage à travers la structure d'un graphe est développée dans le cadre de l'algorithme d'apprentissage semi-supervisé *label propagation* (LP) initialement introduit par Zhu et Ghahramani [2002]. À l'origine, les nœuds du graphe représentent des exemples d'apprentissage et ses arêtes sont pondérées par une distance entre ces exemples. L'algorithme LP est alors capable de transmettre à des nœuds non étiquetés la même étiquetage que leur(s) (proche(s)) voisin(s) étiqueté(s)

Nous envisageons ici d'adapter l'algorithme LP au cadre des treillis de Galois représentés dans un espace sémantique continu (*cf.* chapitre 3). Cette adaptation est nécessaire car les exemples d'apprentissage de L (en l'occurrence, des EN étiquetées issues d'un corpus) sont regroupés ou répartis dans différents concepts d'un treillis alors que ces mêmes exemples sont directement associés à des nœuds d'un graphe avec l'algorithme LP originel. Cette différence de représentation des exemples n'affecte pas la procédure de l'algorithme LP ; en revanche son initialisation diffère : nous cherchons donc ici à décrire quels concepts formels choisir et comment les étiqueter pour cette initialisation.

Le plan de cette section est le suivant. Nous présentons en premier lieu l'algorithme d'apprentissage semi-supervisé *label propagation* qui consiste à propager un étiquetage à travers la structure d'un graphe. Nous adaptons ensuite cet algorithme au cadre des treillis, de sorte qu'un petit nombre de concepts formels étiquetés puissent propager leur étiquetage à d'autres concepts proches à travers la structure de treillis. Nous expérimentons alors cet algorithme sur nos données pour comparer ses performances avec les résultats obtenus lors de nos expériences précédentes. Pour terminer, nous proposons des extensions possibles à notre approche.

5.2.1 Propagation d'étiquettes dans un graphe

L'algorithme LP [Zhu et Ghahramani, 2002] vise à propager progressivement l'étiquetage d'un petit ensemble d'exemples vers le reste des exemples non étiquetés. Ces exemples d'apprentissage sont représentés par les nœuds d'un graphe

dont les arêtes connectent des exemples ayant une affinité particulière. Intuitivement, sa démarche consiste à transmettre à des nœuds non étiquetés le même étiquetage que leur(s) (proche(s)) voisin(s) étiqueté(s). Dans sa formulation originale, l'algorithme LP manipule une matrice d'adjacence symétrique W associée à un graphe non orienté G dans lequel les nœuds représentent les exemples d'apprentissage de l'espace $\mathbb{R}^n$ et les arêtes correspondent à des distances w_i entre des exemples voisins.

Afin d'établir une interprétation probabiliste de l'étiquetage des nœuds du graphe, une matrice P normalisant W est définie telle que les éléments P_{ij} sont les probabilités de transition entre les nœuds i à j du graphe. Dans le calcul des P_{ij}, un poids w_{ik} est matérialisé à la ligne i et à la colonne k de la matrice W :

$$P_{ij} = P(i \rightarrow j) = \frac{w_{ij}}{\sum_k w_{ik}} \tag{5.13}$$

Les données d'apprentissage sont divisées en deux ensembles de nœuds : les étiquetés $L = \{(x_1, y_1) \dots (x_{|L|}, y_{|L|})\}$ et les non étiquetés $U = \{x_{|L|+1} \dots x_{|L|+|U|}\}$, où les x_i sont des exemples et les y_i, leur étiquette. Les ensembles L et U sont associés à deux matrices Y_L et Y_U dont les lignes représentent les nœuds de L ou U et les colonnes correspondent aux étiquettes de $Labels = \{l_1, l_2, \cdots, l_{|Labels|}\}$. Une ligne de la matrice peut être interprétée comme l'étiquetage probabiliste d'un nœud : dans la matrice, chaque élément y_{ij} indique la probabilité d'une étiquette l_j sachant un nœud n_i. Par ailleurs, nous définissons une autre matrice $Y = \begin{bmatrix} Y_L \\ Y_U \end{bmatrix}$ de dimension $(|L| + |U|) \times |Labels|$ qui réunit l'étiquetage des nœuds de L et U. Il est important de remarquer que les nœuds du graphe G doivent être associées au même numéros de lignes dans les matrices Y, W et P.

L'objectif de l'algorithme LP est de déterminer Y_U afin de connaître les étiquettes des nœuds de U. Sa procédure est initialisée par une matrice Y_0 dont on cherche à affiner l'estimation au cours des itérations de l'algorithme :

$$Y^0 = \begin{bmatrix} Y_L^0 \\ Y_U^0 \end{bmatrix} = \begin{array}{c} \\ x_1 \\ \vdots \\ x_{|L|} \\ x_{|L|+1} \\ \vdots \\ x_{|L|+|U|} \end{array} \begin{array}{c} \begin{array}{ccc} l_1 & \dots & l_{|Labels|} \end{array} \\ \begin{bmatrix} P(l_1|x_1) & \dots & P(l_{|Labels|}|x_1) \\ \vdots & & \vdots \\ P(l_1|x_{|L|}) & \dots & P(l_{|Labels|}|x_{|L|}) \\ 0 & \dots & 0 \\ \vdots & & \vdots \\ 0 & \dots & 0 \end{bmatrix} \end{array} \tag{5.14}$$

Il est prouvé que l'algorithme converge vers une unique solution $\hat{Y}_U$ qui ne dépend pas de Y_U^0. L'initialisation de Y_U^0 n'est donc pas importante et c'est pour cette raison que tous les éléments de Y_{ij}^0 de Y_U^0 ont été affectés à 0 dans l'équation 5.14.

L'algorithme de propagation peut se formuler de la manière suivante :

Algorithme 10: Propagation d'étiquetages dans un graphe

$Y^0 \leftarrow \begin{bmatrix} Y_L^0 \\ Y_U^0 \end{bmatrix}$
$i \leftarrow 0$
répéter
  $\begin{bmatrix} Y_L^{i+1} \\ Y_U^{i+1} \end{bmatrix} \leftarrow PY^i$ // Propagation
  $Y^{i+1} \leftarrow \begin{bmatrix} Y_L^0 \\ Y_U^{i+1} \end{bmatrix}$ // Réinitialisation des données étiquetées $Y_L^i \leftarrow Y_L^0$
  $i \leftarrow i + 1$
tant que $Y_U^{i-1} \neq Y_U^i$;
pour chaque *nœud* $x_h \in U, (|L| + 1 \leq h \leq |L| + |U|)$ **faire**
  Assigner à x_h son étiquette majoritaire $\hat{y}_h = \underset{l_j \in Labels}{\operatorname{argmax}} Y_{hj}$

À chaque itération de l'algorithme de propagation, l'étiquetage des exemples de Y_L^i est réinitialisé avec leur étiquetage original Y_L^0 car à chaque itération, le produit matriciel PY^i engendre une valeur de Y_L^{i+1} différente de Y^i. Ainsi les exemples de Y_L jouent un rôle de source stable et perpétuelle d'étiquetage qui est diffusée progressivement aux exemples non étiquetés Y_U à chaque itération. Notons que des distances courtes entre les exemples facilitent la propagation de leur étiquetage. Lorsque la valeur de Y_U^i converge, chaque nœud $x_h \in U$ est étiqueté par sa classe majoritaire : celle-ci est associée au poids le plus probable de la ligne Y_h affectée au nœud x_h dans la matrice Y_U.

L'algorithme LP est habituellement appliqué sur des graphes non orientés dont les nœuds correspondent à des exemples d'apprentissage représentables dans $\mathbb{R}^n$. L'étape de création du graphe est très critique car sa topologie détermine ensuite la qualité d'étiquetage des exemples. Une manière simple de déterminer les arêtes de ce graphe est de connecter chaque exemple avec ses k plus proches voisins dans $\mathbb{R}^n$. La pondération des arêtes correspond alors aux distances entre les exemples. Nous ne sommes toutefois pas concerné par ce problème de détermination dans le cadre des treillis de Galois car ces derniers correspondent à déjà à des graphes. Cependant, comme nous l'avons déjà indiqué, les exemples d'apprentissage sont regroupés ou répartis dans les concepts d'un treillis alors que ces mêmes exemples sont directement associés à des nœuds d'un graphe avec l'algorithme LP originel. Nous devons à présent décrire l'impact de cette représentation de treillis sur les

mécanismes de propagation en précisant notamment comment initialiser l'étiquetage des concepts formels.

5.2.2 Adaptation au cadre des treillis

Les treillis de Galois correspondent à des graphes théoriquement manipulables par l'algorithme LP. Dans l'adaptation inédite que nous proposons ici, les concepts formels sont considérés comme les nœuds manipulés par l'algorithme et les relations d'héritage entre ces concepts correspondent aux arêtes du graphe. Pour commencer, nous examinons un exemple de concepts formels cartographiés en montrant que leur étiquetage pourrait être propagé à l'aide de l'algorithme LP. Nous proposons ensuite d'examiner l'hypothèse que des concepts proches dans leur espace sémantique tendent à posséder le même étiquetage. Nous indiquons enfin comment constituer les matrices manipulées par l'algorithme de propagation à partir d'un treillis.

5.2.2.1 Cartographie de concepts formels étiquetés

Le travail effectué sur la classification supervisée de concepts formels nous permet de produire une cartographie de treillis (*cf.* chapitre 3) sur laquelle les concepts sont étiquetés. Nous étudions ici l'entité nommée *Wimbledon* qui présente le plus grand nombre d'étiquettes dans notre corpus. Les liens hiérarchiques organisent les concepts en plusieurs branches correspondant à des étiquetages différents. La carte de la figure 5.9 montre que l'entité nommée *Wimbledon* présente plusieurs facettes sémantiques : en partant du concept c_{11795}, elle se diversifie en effet en plusieurs branches attachées aux types MISC, ORG et LOC. L'EN *Wimbledon* peut correspondre à une compétition de tennis (type MISC en noir : concepts 231, 8642, 11780, 11786 et 11790) mais peut aussi être associées à une équipe de football (type ORG représenté par des triangles en jaune : concepts 164, 3995, 4940 et 11792) ou à un quartier en banlieue londonienne (type LOC associé à des pentagones en magenta : concepts 9864, 11781, 11782 et 11789).

D'autres cartes d'unités polysémiques nous ont montré que la proximité géométrique des concepts formels sur la carte est corrélée à leur étiquetage. En effet des concepts sont proches s'ils possèdent des variables communes : ils tendent donc à avoir un étiquetage proche. Si des concepts sont éloignés par de grandes distances, ils appartiennent à deux zones pouvant être affectées à des étiquetages différents, tandis que des concepts séparés par de faibles distances tendent à posséder le même étiquetage. Cette observation s'accorde avec l'hypothèse sous-jacente à l'algorithme de propagation selon laquelle des exemples appartenant au même *cluster* (inférée par analyse non supervisée) possèdent la même étiquette (cible d'un apprentissage supervisé). Nous regardons maintenant si cette hypothèse est

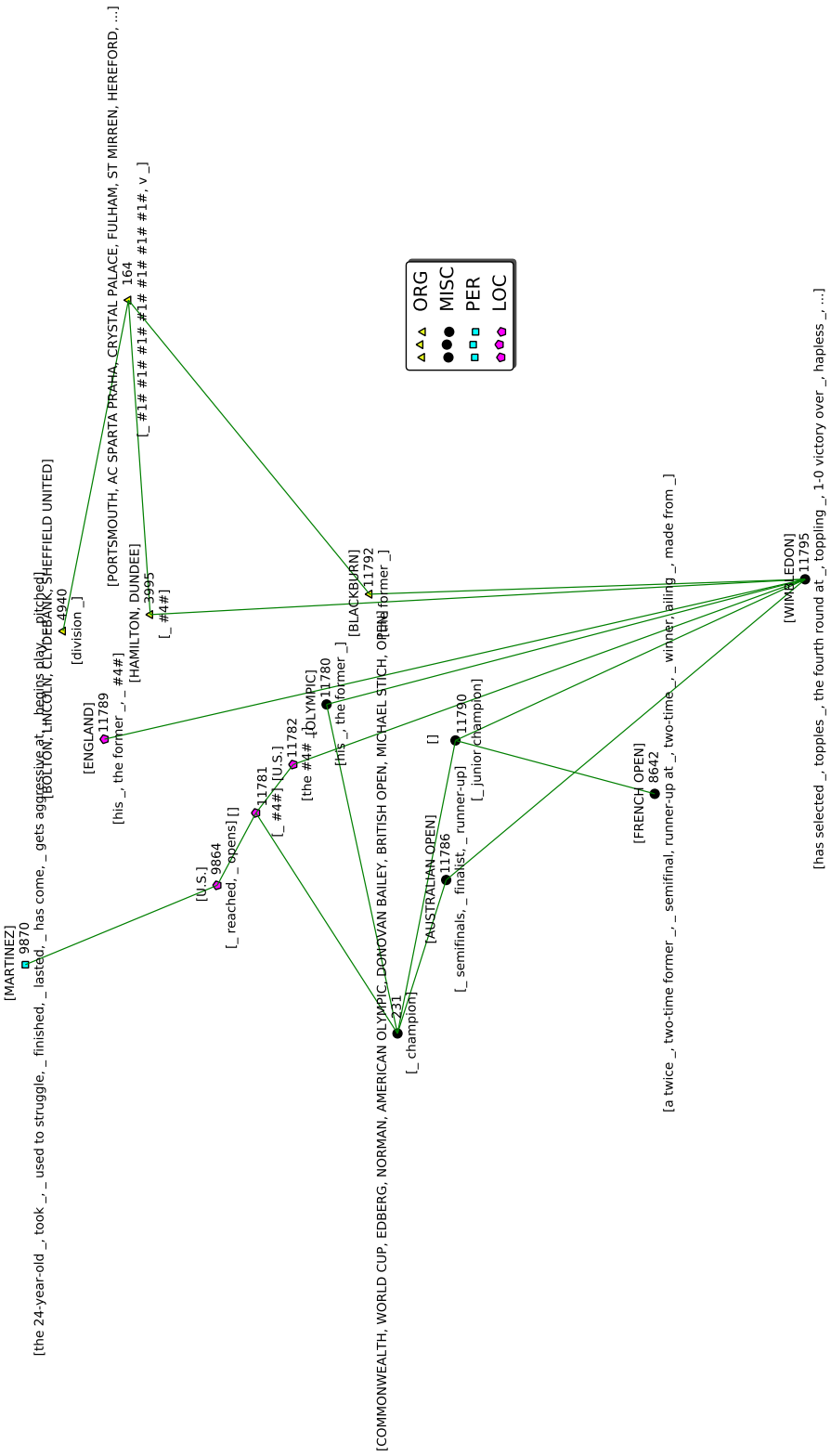


FIG. 5.9 – Cartographie de concepts étiquetés liés à l'entité nommée « Wimbledon »