Consolidation endogène de réseaux lexico-sémantiques : Inférence et annotation de relations, règles d'inférence et langage dédié

Manel Zarrouk

Collège Doctoral du Languedoc-Roussillon

THÈSE

Pour obtenir le grade de

Docteur

Délivré par l'**Université Montpellier**

Préparée au sein de l'école doctorale **I2S**[1]
et de l'unité de recherche **LIRMM**[2]

Spécialité: **Informatique**

Présentée par **Manel Zarrouk**

Consolidation endogène de réseaux lexico-sémantiques :

Inférence et annotation de relations, règles d'inférence et langage dédié

Soutenue le 03/11/2015 devant le jury composé de :

Pr.	Patrice Bellot	Univ. Aix-Marseille	Rapporteur
Pr.	Marianne Huchard	Univ. de Montpellier	Examinatrice
Pr.	Marie-Claude L'Homme	Univ. de Montréal	Rapporteuse
Dr. HDR	Mathieu Lafourcade	Univ. de Montpellier	Directeur de thèse
Pr.	Mme. Anne Laurent	Univ. de Montpellier	Examinatrice
DR.	M. Michael Zock	Univ. Aix-Marseille	Examinateur

[1] École doctorale Information Structures Systèmes

[2] Laboratoire d'Informatique, Robotique et Micro-électronique de Montpellier

Remerciements

Comme le dicte la tradition, je vais tenter de combler la tâche difficile de remplir la page des remerciements, peut-être la tâche la plus épineuse de ces années de thèse. Non qu'exprimer ma gratitude envers les personnes en qui j'ai trouvé un soutien soit contre ma nature, bien au contraire. La difficulté tient plutôt dans le fait de n'oublier personne. C'est pourquoi, je remercie par avance ceux dont le nom n'apparaît pas dans cette page et qui m'ont aidé d'une manière ou d'une autre. Ils se reconnaîtront.

En premier lieu, je tiens à remercier mon directeur de thèse, M. Mathieu Lafourcade, pour la confiance qu'il m'a accordée avec ce travail doctoral, pour ses multiples conseils et pour toutes les heures qu'il a consacrées à diriger cette recherche. J'aimerais également lui dire à quel point j'ai apprécié sa grande disponibilité son encouragement et son respect sans faille des délais serrés de relecture des documents que je lui ai adressés. Nos continuelles discussions, oppositions, contradictions et confrontations ont sûrement été la clé de notre travail commun. Merci sensei.

Je tiens à remercier Mme Marie-Claude L'Homme et M. Patrice Bellot pour avoir accepté d'être rapporteurs de ma thèse ainsi que Mme Marianne Huchard, Mme Anne Laurent et M. Michael Zock pour participer à mon jury. J'éprouve un profond respect pour leurs travaux et leurs parcours. Merci pour le regard critique et avisé que vous porterez.

Une pensée à mes compagnons de fortune (ou d'infortune). On ne

se rend pas toujours compte à quel point ils peuvent être importants dans le travail et dans la vie, jusqu'au jour où nos chemins se séparent.

Merci aux (ex-)membres de l'équipe TEXTE pour les multiples discussions entre autre autour des repas partagés au RA.

Il m'est impossible d'oublier Nicolas Serrurier (le superman du LIRMM), Guylaine Martinoty et Laurie Lavernhe pour leur professionnalisme, leur disponibilité et leur aide précieuse. Ainsi que Ghislaine Takessian, Caroline Ycre, Cécile Lukassik, Katya Lopez...

Je ne manque pas de penser à deux enseignants que j'ai eu pendant mes études en Tunisie et qui ont marqué mon esprit, Dr. Hatem Bettaher – qu'il repose en paix – et M. Mansour Elghoul.

Un énorme merci pour mes parents Lassaad et Sonia pour l'aide morale et la motivation qu'ils m'ont apportées malgré la distance, pour leur soutien, leur confiance et leurs encouragements tout au long de ces années de thèse. Merci beaucoup pour votre amour sans condition auquel je ne trouverai aucun égal. Le fait que vous soyez présents le jour de ma soutenance avec mes frères Mehdi et Baha me remplit de bonheur. نحبّكم برشا و ربّي يخلّيكم ليّا.

Un merci particulier à ma deuxième maman *Mafan* à qui je dédie ce travail et je souhaite tout le bonheur du monde, ainsi qu'à ma tante Rinda en lui souhaitant plein de courage et de sérénité.

Une pensée à mes grands parents Ali, Ammar – qu'il repose en paix –, Khedhireya et Latifa, et au reste de ma famille en Tunisie.

Une pensée spéciale à mon oncle Chokri et à mon P.P.Jacques-Favori qui nous ont quitté cette année. Je vous envoie où que vous soyez l'accomplissement de ce travail enfin, je l'ai fait et je l'ai fini. Merci pour votre présence et que vous reposiez en paix.

Un grand merci à ma deuxième famille pour tout, les superbes balades, les chaleureux repas... Patrick شكراً جزيلاً pour les relectures ainsi que les discussions entomologiques amusantes. Laurence merci pour nos petites conversations et ton attention. Prunelle, merci de nous avoir donné la petite coccinelle, entre autres. Notre virée au Hammam nous attend encore. Merci – vachement – beaucoup à toi Colin *la tête* pour ton soutien quotidien indéfectible, ton enthousiasme contagieux à l'égard de mes travaux comme de la vie en général et tes précieux coups de main

techniques qui m'ont bien aidée à avancer ce travail.

Je tiens finalement à remercier mes amis pour leur amitié tout simplement. Solène, Guillaume, Lilou, Fred, Juanitto, Lionel, Mohamed, Mehdi, Nouha, Manel, etc. J'en oublie certainement encore, qu'ils acceptent mes excuses.

Résumé étendu

Développer des ressources lexicales et sémantiques pour le TAL est un enjeu majeur du domaine. La plupart des ressources existantes ont été construites à la main, comme dans le cas de WordNet. Les approches entièrement automatisées sont généralement limitées à la cooccurrence des termes car l'extraction des relations sémantiques précises entre termes, à partir d'un texte, reste difficile. Bien entendu, des outils sont généralement utilisés pour la vérification de la cohérence, cependant la tâche d'acquisition est longue, coûteuse et demande énormément d'effort. De nouvelles approches impliquant l'externalisation ouverte (crowdsourcing) émergent en TAL. Les ressources sont construites par participation volontaire des contributeurs comme pour Wikipédia et Wikitionnaire ou par GWAP (jeux avec but) où les joueurs sont inconscients de la vraie finalité du jeu.

L'expérience montre que les joueurs/contributeurs complètent le réseau sur ce qui les intéresse. Ce faisant, un grand nombre de relations triviales ne sont pas présentes bien qu'elles soient nécessaires à l'obtention d'un réseau de qualité visant à être utilisé dans diverses applications du TAL, notamment l'analyse sémantique, c'est-à-dire comprendre automatiquement un texte et en tirer des relations fines et pertinentes.

Le cas d'étude de cette thèse est le réseau lexico-sémantique JeuxDeMots qui est construit par crowdsourcing, par le biais d'un outil con-

tributif (Diko) et de multiples GWAP en ligne.

Cette thèse tend à déterminer dans quelle mesure il est possible de consolider automatiquement un réseau lexical, sans avoir recours à une ressource externe. En effet, enrichir, faire évoluer et consolider ce type de ressources est possible en comblant les manques d'information et en détectant les erreurs, la polysémie et le bruit. Ceci pourrait être réalisé principalement par la création de nouvelles relations sémantiques à partir de celles existantes dans le réseau lexical.

Un système d'inférence a été mis en place. Il se base sur les relations déjà existantes dans le réseau et sur des schémas d'inférences comme modèles de comportement pour proposer, à l'instar d'un contributeur humain, de nouvelles relations soumises par la suite au processus de validation/invalidation par les utilisateurs. Pour améliorer la pertinence des inférences produites par ce système, des annotations de relations ont été rajoutées, qui sont autant de méta-informations permettant de contrôler l'activité d'inférence. Ces annotations sont ensuite propagées à travers le système d'inférence.

Une autre facette du travail de recherche effectué est la découverte de règles d'inférences ontologiques réalisée par énumération des formes ($\forall$x, *is-a*(x,B) $\Rightarrow$ *R*(x,C)) implicitement présentes dans le réseau. Une fois identifiées, ces règles sont présentées à la validation/invalidation par l'utilisateur. L'utilité de l'extraction des règles, mis à part l'enrichissement du réseau et le comblement des silences, réside dans le fait de pouvoir enlever toutes les relations inférables par ces règles pour ne garder que les relations utilisées dans les prémisses. Le résultat est un réseau que l'on peut qualifier de *déshydraté*. Il s'agit d'un réseau de taille minimale sur lequel il est possible d'appliquer un processus d'application de règles d'inférence qu'il contient afin de retrouver le réseau dans son état complet. À noter qu'il faut garder les relations ayant un poids négatif même dans l'état minimal du réseau : elles présentent les exceptions et l'information cruciale obtenue des utilisateurs mais ne pouvant être inférée automatiquement. L'objet de ce système de règles est de donner au réseau un aspect plus léger et la possibilité à l'utilisateur (un humain ou un processus automatique) d'appliquer les règles sur des termes précis selon le choix lors de leur utilisation, par exemple dans des processus d'analyse.

Afin de manipuler ce système de règles, un langage spécialisé a

été conçu permettant à des non informaticiens ou des non linguistes de l'utiliser facilement.

Beaucoup de perspectives aussi bien théoriques que pratiques sont envisagées, telles que la limitation de la portée des schémas d'inférences et l'amélioration de la qualité du système de règles d'inférences en lui donnant, par exemple, la capacité de prédire la véracité de la règle proposée.

Sommaire

Liste des tableaux

Liste des Algorithmes

Table des figures

INTRODUCTION GÉNÉRALE

Ressources de connaissances

À l'ère actuelle, le texte numérique est l'un des moyens principaux de représentation et de transmission de l'information comme en témoigne la dominance de l'écrit, e-mails, textes, weblogs, articles, rapports, corpus, etc. L'explosion de ces ressources numériques textuelles a rendu urgent le développement d'une technologie aidant à gérer et à exploiter l'information obtenue.

Le Traitement Automatique de la Langue Naturelle (TALN) ayant pour but de donner aux machines la capacité d'interpréter et *comprendre* [1] le texte, présente de nombreuses approches de traitement sémantique tels que la traduction automatique, l'extraction de mots-clé, l'indexation des textes, l'analyse de texte, l'extraction d'information, le résumé automatique, la génération automatique des textes, etc. Toutes ces approches ont connu un succès indéniable (Chowdhury, 2003). Néanmoins, on constate de plus en plus que de telles approches se livrent à une compréhension très peu profonde (Yvon, 2010).

Afin d'améliorer le niveau de compréhension des approches d'analyse

[1] Notre travail s'intéresse principalement au niveau sémantique de la langue (plusieurs aspects existent: morphologique, syntaxique, pragmatique, etc.) (Yvon, 2010)

sémantique actuelles, de grandes bases de connaissances sémantiques spécialisées mais aussi généralisées sont nécessaires pour avoir la capacité d'automatiser une compréhension de texte plus profonde.

Plusieurs types de connaissances peuvent être le sujet de recherches mais le plus dur à obtenir, qui est également le plus largement applicable, est le type de connaissances possédé par tout le monde ou plus généralement appelé *sens commun*. Tandis que dans un contexte quotidien le terme *sens commun* ou *bon sens* est considéré comme synonyme de *bon jugement*, dans les disciplines scientifiques il renvoie aux millions de faits, de connaissances générales et aux conventions possédées par la plupart des personnes : *Une tortue respire. Un citron est acide. Si on marche sous la pluie on va être mouillé. Si deux personnes se disputent, ils vont être probablement fâchés après, etc.*.

Ce type de connaissances, ainsi identifié, couvre une énorme partie des expériences humaines, englobant la connaissance des aspects spatiaux, physiques, sociaux, temporels et psychologiques de la vie quotidienne. En admettant que chaque personne possède cette connaissance, celle-là est typiquement non explicite dans les communications sociales et en particulier dans les textes tant elle semble aller de soi.

Une extraction de relations fines lors d'une analyse d'un texte ainsi qu'une compréhension assez complète de ce dernier, exige une quantité surprenante des connaissances du monde (linguistiques et terminologiques) que seuls les humains possèdent actuellement.

Néanmoins, dans le TALN une orientation majeure des efforts de recherche consiste à concrétiser ou expliciter les connaissances implicites précédemment mentionnées et ce sous forme de ressources lexico-sémantiques en les mettant ainsi à la portée des machines (Gala and Zock, 2013). Ces ressources sont de formes et conceptions variées et répondent à différents types d'usages.

La constitution de ces ressources est l'une des tâches primordiales en TALN. La plupart des réseaux lexico-sémantiques existants ont été construits manuellement (par exemple WordNet (Miller and Fellbaum, 2007) [2]). Des outils ont été conçus afin d'en vérifier la validité et la complétude, mais leur mise en œuvre nécessite généralement beaucoup de temps, d'argent et d'efforts.

[2]https://wordnet.princeton.edu/

Les approches automatiques, quant à elles, sont le plus souvent limitées aux co-occurrences de termes, et la détermination précise de relations sémantiques entre termes reste délicate. Citons comme exemples EuroWordnet (Vossen, 1998), une version multilingue de WordNet, et WOLF (Sagot and Fiser, 2008), une version française de WordNet, construits automatiquement par croisement de WordNet avec d'autres ressources lexicales. Navigli and Ponzetto (2012) a construit automatiquement BabelNet, un réseau lexical multilingue de grande taille, à partir de co-occurrences de termes dans Wikipédia.

De nouvelles approches, fondées sur l'externalisation ouverte (Lebraty, 2007), apparaissent en TALN, avec en particulier le développement de Amazon Mechanical Turk (Crowston, 2012) ou dans un contexte plus étendu l'émergence de Wikipédia et Wiktionnary. Afin de construire de manière collaborative un réseau lexical (ou n'importe quelle ressource similaire), deux grandes stratégies peuvent être mises en œuvre. La première résulte en un système contributif tel que Wikipédia. Dans la seconde stratégie, les contributions sont faites indirectement au travers de jeux, plus connus sous le nom de Games With A Purpose (GWAP) (von Ahn and Dabbish, 2008).

Quelle que soit la stratégie mise en œuvre, la ressource construite n'est pas exempte d'erreurs qui doivent être corrigées au fur et à mesure de leur découverte. Un grand nombre de relations, pourtant pertinentes, ne sont pas présentes dans le réseau, bien que nécessaires pour une ressource de qualité suffisante pour être utilisée dans diverses applications en TALN, en particulier en analyse sémantique.

Dans cette thèse nous avons pris comme cas d'étude le réseau lexico-sémantique JeuxDeMots [3] construit par le biais d'un outil contributif (Diko) et de multiples GWAP en ligne et nous cherchons à détecter les silences d'information et à les combler, détecter le bruit et l'information fausse et les corriger ou les supprimer, ainsi que déceler la polysémie et enrichir les termes en question afin d'éviter dans la mesure du possible l'ambiguïté.

[3] http://jeuxdemots.org

Objectifs de la thèse

AINSI, les questions de recherche principales auxquelles nous proposons de répondre dans cette thèse sont les suivantes :

Est-il possible, par des approches (semi-)automatiques et sans avoir recours à aucune ressource extérieure, d'améliorer et enrichir un réseau lexical ?

Dans quelle mesure peut-on concevoir une intelligence artificielle autonome en perpétuel raisonnement avec la connaissance dynamique du réseau lexico-sémantique ?

Comment ce système peut-il être facilement manipulable par les utilisateurs (humains ou machines) de manière flexible afin de répondre à leurs besoins spécifiques ?

Organisation du mémoire

Ce mémoire est organisé de la manière suivante.

Chapitre 1 - Connaissances lexico-sémantiques
Ce chapitre présente quelques ressources de connaissance lexicale et sémantique et en particulier les réseaux lexico-sémantiques, ainsi que leurs différentes approches d'acquisition. Nous avons pointé le fait que ces ressources, en particulier celles construites par approches contributives, malgré leur efficacité, ne sont pas exemptes de manque d'information.

Dans les chapitres qui suivent nous avons mis en place une approche sous forme d'un système de consolidation endogène de réseaux lexico-sémantiques (SCEREL) (figure 5.1), en prenant comme exemple d'étude le réseau JeuxDeMots construit par externalisation ouverte.

Chapitre 2 - Inférence de relations sémantiques

Nous décrivons dans ce chapitre la première stratégie d'enrichissement et de consolidation du réseau. Dans ce but, un moteur d'inférence de relations sémantiques à partir des relations existantes dans le réseau, a été mis en place. Ce moteur se compose de plusieurs schémas d'inférences statiques et dynamiques guidant son comportement. Les relations inférées sont présentées aux validateurs humains pour être évaluées. Dans le cas de l'invalidation d'une relation inférée, un processus de réconciliation est lancé pour essayer de cerner la cause de l'erreur et éventuellement la corriger. Le moteur d'inférence et le processus de réconciliation représentent le premier élément du SCEREL.

Chapitre 3 - Annotation de relations

La deuxième stratégie de consolidation consiste à introduire dans le réseau une méta-information sous forme d'annotations des relations sémantiques. Cette annotation fournit une information enrichissante sur les relations, comme la fréquence, la quantité, la pertinence, etc, et vont être d'une aide importante pour améliorer la qualité des inférences et de l'analyse sémantique. Les premières annotations sont introduites manuellement mais propagées automatiquement par la suite avec le moteur d'annotation qui représente la deuxième composante du SCEREL. Le moteur d'annotation utilise les relations proposées par le moteur d'inférence et validées par les utilisateurs.

Chapitre 4 - Extraction de règles d'inférence

Ce chapitre décrit l'approche d'extraction de règles d'inférence par l'inspection du contenu du réseau. Une règle, une fois validée par un utilisateur, sera appliquée afin de produire de nouvelles relations dans le réseau. Si inférer des relations sémantiques nous permet d'inférer des faits, produire des règles enrichit notre réseau par des connaissances. Le moteur d'extraction de règles d'inférence est la dernière composante du SCEREL.

Chapitre 5 - Vers un langage de manipulation du SCEREL

Dans le but de permettre à l'utilisateur de manipuler les composantes ci-dessus décrites du SCEREL, ainsi que d'écrire des programmes plus concis, plus faciles à maintenir et permettant à moindre coût d'avoir

des retours pertinents, nous avons conçu un langage dédié de manipulation du SCEREL. La manipulation consiste, d'une part, à interagir avec le réseau et d'autre part, à appliquer tout type de schéma d'inférence, chercher des termes satisfaisant une ou plusieurs conditions, chercher les règles valides ou visualiser les règles à valider/invalider autour d'un terme donné ou un sous-ensemble de termes.

Chapter 1

CONNAISSANCES LEXICO-SÉMANTIQUES

Sommaire

De nos jours, les ressources numériques textuelles explosent. Des modèles et algorithmes de TALN sont nécessaires pour que les machines soient capables d'interpréter et analyser le texte. Pour *comprendre* et par la suite analyser d'une manière efficace, les machines ont besoin des connaissances lexicales et sémantiques que tout humain possède. Ces connaissances ont été élaborées sous plusieurs formes, allant des terminologies déstructurées (liste de mots) à des glossaires (Velardi et al., 2008), thésaurus (Harabagiu, 1911), dictionnaires numériques (exemple GCIDE (Cassidy, 2002)), ontologies (l'ontologie de la santé mentale (Ceusters and Smith, 2010), l'ontologie des gènes (Ashburner et al., 2000), Cyc (Lenat, 1995), etc.) et réseaux lexico-sémantiques (JeuxDeMots (Lafourcade, 2007), ConceptNet (Liu and Singh, 2004), etc.).

La construction de ces ressources varie selon les motivations et le contexte dans lequel elles étaient élaborées. Ces ressources ont été construites manuellement (WordNet (Miller and Fellbaum, 2007), HowNet (Dong et al., 2010), etc.), (semi)-automatiquement(BabelNet (Navigli and Ponzetto, 2012), MultiWordNet (Pianta et al., 2002), etc.), par externalisation ouverte (Wikitionnaire[1]) ou par des approches mixtes.

1.1 Ressources de connaissances

1.1.1 Lexiques / glossaires / index / vocabulaires

Le lexique d'une langue est un ensemble de lemmes (mots) ou plus précisément d'unités lexicales de cette langue associées à une brève définition ou à son équivalent dans une autre langue pas nécessairement dans un ordre alphabétique. Généralement les lexiques portent sur un domaine précis. Ceux d'un langage de spécialité se présente sous la forme de listes de termes employés dans ce domaine spécialisé (exemple: lexique de terminologie linguistique, de termes d'art, de marine, d'agriculture...).

Exemple de lexique théâtral : (Bouchard, 1878) Les exemples sont extraits du *Mariage de Figaro de Beaumarchais*:

[1] https://fr.wiktionary.org

- *Un aparté* : réplique qu'un personnage dit à part, pour lui-même, sans que les autres ne l'entendent ; l'aparté est toujours signalé par une didascalie : *à part*
  Le Comte, à part .- Ah! voilà mon fripon du billet;

- *Une tirade* : longue réplique qui peut être narrative par exemple un personnage fait le récit d'événements qui se sont passés;

- *La stichomythie* : échange très rapide de répliques très courtes (interjections, phrases très concises, etc.)
  Suzanne - Le cachet, à quoi?
  La Comtesse - A son brevet
  Suzanne - Déjà ;

- *Le monologue*: un personnage seul sur scène dit à voix haute ce qu'il pense ou ce qu'il ressent;

- *La didascalie*: indication scénique qui relève du paratexte auctorial (vous ne devez pas la lire). Elle est utile pour comprendre les jeux de scènes.

Ces ressources sont destinées à être utilisées par les machines dans les applications de TALN, mais aussi par les humains. Elles doivent être formalisées selon deux critères. Un critère linguistique qui impose que le lexique-grammaire de la langue soit explicitement et clairement exprimé (Leclère et al., 2004). L'autre critère est informatique et exige que l'implémentation permette un requêtage utile tout en étant le plus rapide possible.

Dans ce domaine, un lexique se présente sous forme d'entrées lexicales / unités linguistiques élémentaires s'associant généralement aux notions de mots. Il associe à chaque entrée une clé permettant l'accès à l'ensemble d'informations linguistiques décrivant cette entrée. Il doit permettre à la machine l'accès aux informations associées aux mots, l'ajout / suppression / modification d'entrées, la modification de la structure des entrées...

Ci-dessous (figure 1.1) un exemple du Lexique 3[2] (version 3,55) de New et al. (2001).

[2]http://www.lexique.org

ortho	phon	lemme	cgram	genre	nombre	freqlemfilms2	freqlemlivres	freqfilms2	freqlivres	infover	nbhomogr	nbhomoph	islem	nblettres	nbphons	cvcv	p_cvcv
coccinelle	koksinEl	coccinelle	NOM	f	s	1.33	1.82	1.15	1.35		1	2	1	10	8	CVCCVCVCCV	CVCCVCVC
crique	kRik	crique	NOM	f	s	1.10	5.20	1.06	3.78		1	6	1	6	4	CCVCVV	CCVC
méduse	medyz	méduse	NOM	f	s	1.50	3.92	0.66	1.42		2	5	1	6	5	CVCVCV	CVCVC
méduse	medyz	méduser	VER			0.48	1.96	0.01	0.00	ind:pre:3s;	2	5	0	6	5	CVCVCV	CVCVC
tortue	tORty	tortu	ADJ	f	s	0.69	0.68	0.58	0.34		2	6	0	6	5	CVCCVV	CVCCV
tortue	tORty	tortue	NOM	f	s	5.34	6.22	4.00	4.66		2	6	1	6	5	CVCCVV	CVCCV

voisorth	voisphon	puorth	puphon	syll	nbsyll	cv_cv	orthrenv	phonrenv	orthosyll	cgramortho	deflem	defobs	old20	pld20	morphoder	nbmorph
0	0	5	5	kok-si-nEl	3	CVC-CV-CVC	ellenicco	lEniskok	coc-ci-nel-le	NOM	100	16	3.65	3.05	coccinelle	1
7	14	6	4	kRik	1	CCVC	euqirc	kiRk	cri-que	NOM	96	32	1.55	1.10	crique	1
3	1	6	5	me-dyz	2	CV-CVC	esudém	zydem	mé-du-se	NOM,VER	100	27	1.70	1.75	méduse	1
3	1	6	5	me-dyz	2	CV-CVC	esudém	zydem	mé-du-se	NOM,VER	85	21	1.70	1.75	méduser	1
3	2	6	5	tOR-ty	2	CVC-CV	eutrot	ytROt	tor-tue	ADJ,NOM	15	20	1.70	1.85	tortue	1
3	2	6	5	tOR-ty	2	CVC-CV	eutrot	ytROt	tor-tue	ADJ,NOM	100	17	1.70	1.85	tortu	1

Figure 1.1 – Extrait du Lexique 3 pour les termes *coccinelle*, *méduse*, *tortue*, *crique*. Légende: **ortho:** le mot; **phon:** les formes phonologiques du mot; **lemme:** les lemmes de ce mot; **cgram:** les catégories grammaticales de ce mot; **genre:** le genre; **nombre:** le nombre; **freqlemfilms:** la fréquence du lemme selon le corpus de sous-titres (par million d'occurrences); **freqlemlivres:** la fréquence du lemme selon le corpus de livres (par million d'occurrences); **freqfilms:** la fréquence du mot selon le corpus de sous-titres (par million d'occurrences); **freqlivres:** la fréquence du mot selon le corpus de livres (par million d'occurrences); **infover:** modes, temps, et personnes possibles pour les verbes; **nbhomogr:** nombre d'homographes; **nbhomoph:** nombre d'homophones; **islem:** indique si c'est un lemme ou pas; **nblettres:** le nombre de lettres; **nbphons:** nombre de phonèmes; **cvcv:** la structure orthographique; **p-cvcv:** la structure phonologique; **voisorth:** nombre de voisins orthographiques; **voisphon:** nombre de voisins phonologiques; **puorth:** point d'unicité orthographique; **puphon:** point d'unicité phonologique; **syll:** forme phonologique syllabée; **nbsyll:** nombre de syllabes ; **cv-cv :** structure phonologique syllabée; **orthrenv:** forme orthographique inversée; **phonrenv:** forme phonologique inversée; **orthosyll:** forme orthographique syllabée.

1.1.2 Dictionnaires

UN *dictionnaire d'usage* est un recueil de références contenant un ensemble de mots d'une langue spécifique ou d'un domaine de l'activité humaine classés par ordre alphabétique et fournissant chacun des informations relatives à l'emploi et au sens de ce mot.

Une définition plus complète donnée par un dictionnaire de linguistique (Dubois et al., 1973) :

> *Le dictionnaire est un objet culturel qui présente le lexique d'une (ou plusieurs) langues sous forme alphabétique,*

> *en fournissant sur chaque terme un certain nombre d'informations (prononciation, étymologie, catégorie grammaticale, définition, construction, exemples d'emploi, synonymes, idiotismes) ; ces informations visent à permettre au lecteur de traduire d'une langue dans une autre ou de combler les lacunes qui ne lui permettaient pas de comprendre un texte dans sa propre langue.*

Exemple d'entrée dans un dictionnaire informatisé (Larousse en ligne[3]):

aparté
nom masculin

Définitions
• Ce qu'un acteur dit à part soi sur la scène, et qui est censé n'être entendu que des spectateurs.
• Conversation discrète tenue à l'écart, dans une réunion, dans un petit groupe.

Expressions
En aparté, à part soi : je m'en suis fait la réflexion en aparté ; seul à seul avec quelqu'un : il me l'a dit en aparté.

Synonymes
Conversation discrète tenue à l'écart, dans une réunion, ...
Synonyme : tête-à-tête

En TALN, les dictionnaires sont conçus pour une utilisation et compréhension par les machines, d'où nous parlons de dictionnaire exploitable par la machine (DEM) (machine-readable dictionary). Ils se présentent sous une forme électronique qui peut être interrogée par des programmes informatiques.

Ci dessous un exemple d'entrées du dictionnaire morphologique MORFETIK (Grezka et al., 2015)

[3] http://www.larousse.fr/

```
<morphlex forme="auxquels" lemme="auquel"
catgram="D:Rel" cat="DP" genre="M" num="P"
notes="à+LESQUELS"/>

<morphlex forme="tuyaux" lemme="tuyau" catgram="nm"
cat="Nom" genre="M" num="P"/>

<morphlex forme="acculaient" lemme="acculer"
catgram="Verbe" cat="Vrb" num="P" person="3"
temps="Ind_imp"/>
```

Les dictionnaires ayant des entrées (concepts, termes, etc.) structurées hiérarchiquement s'appellent des *taxonomies*. Et s'ils contiennent en plus des relations entre les concepts, on parle alors d'*ontologies*.

1.1.3 Ontologies

SELON Guarino ((1998),

> *une ontologie est constituée par un vocabulaire spécifique utilisé pour décrire une certaine réalité, accompagné d'un ensemble d'hypothèses implicites concernant la signification des mots de vocabulaire.*

Selon Gruber (1993),

> *une ontologie est une spécification explicite d'une conceptualisation.*

Une ontologie organise les concepts indépendamment de la langue et des mots dans lesquels ces concepts s'incarnent. Alors, une ontologie peut être vue comme un modèle de données représentatif de la connaissance au sujet d'un monde ou d'une partie de ce monde.

Le but d'une ontologie est de permettre aux machines de raisonner à propos des objets d'un domaine spécifique (d'Aquin, 2012) et elles ont pour but d'être partageables, cohérentes, normatives et consensuelles.

Elles peuvent être réparties en plusieurs types (ontologie de catégorie, générique et spécialisée) (figure 1.2).

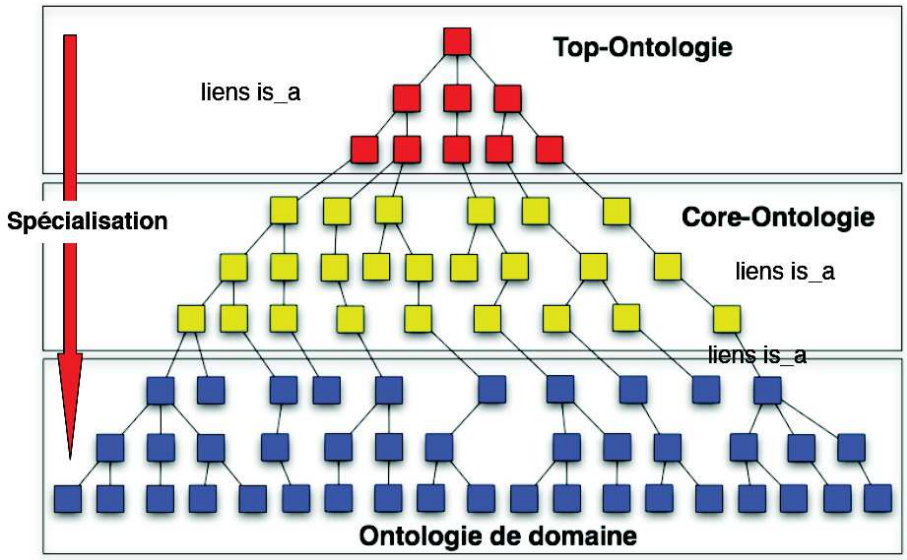

Figure 1.2 – Les différents types d'ontologies par niveau de spécialisation

Ontologie de catégorie

L*'ontologie de catégorie* aussi appelée Top-Ontologie est une ontologie structurant des connaissances de haut-niveau catégorisées selon des notions philosophiques. Ces ontologies globales se spécialisent en définissant les ontologies contenant les objets les plus généraux du domaine (artefact, entité, événement, état, processus...).

Citons quelques exemples de Top-Ontologie :

- SUMO (Suggested Upper Merged Ontoloy) (Niles and Pease, 2001) est une ontologie développée dans le cadre du projet IEE SUO (Standard Upper Ontology) ayant pour objectif de constituer un standard pour mettre l'interopérabilité sémantique entre tous les systèmes d'information, la recherche d'information, l'inférence automatisée et le traitement de langues naturelles (Pease and Niles, 2002). SUMO est à notre connaissance la seule ontologie formelle qui a été mise en correspondance avec tout le lexique de WordNet (figure 1.3);

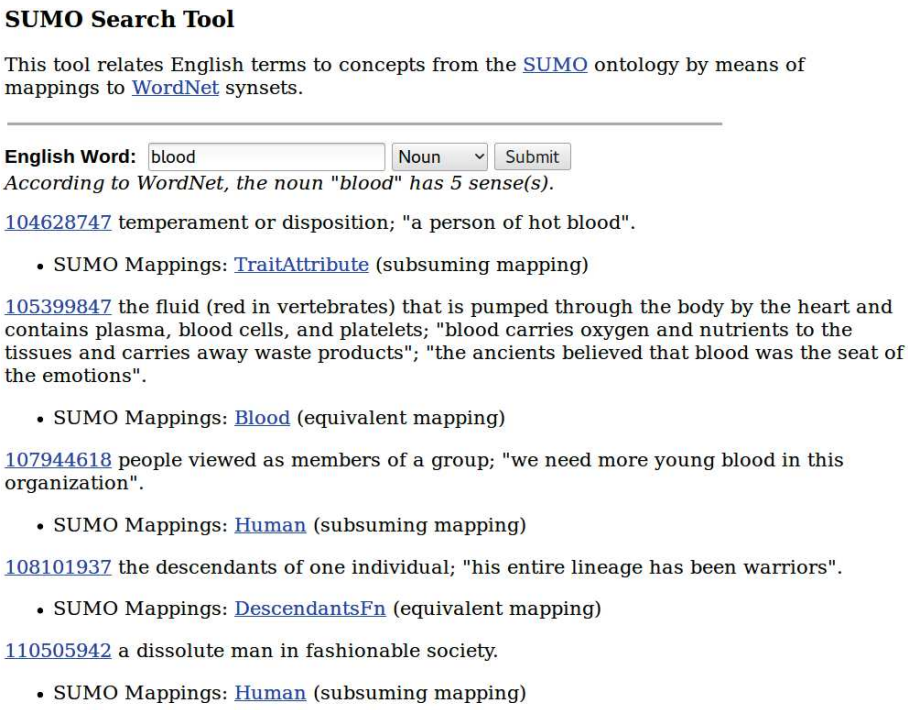

Figure 1.3 – Exemple de résultat pour le terme *blood* (sang) dans l'ontologie SUMO 1.3

- DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) (Gangemi et al., 2002) qui est une méta ontologie pour concevoir des ontologies de domaine.

Ontologie générique

L*'ontologie générique* est une ontologie de type thesaurus qui fournit les concepts structurant un certain domaine et les relations entre les concepts. Elle présente la spécification d'un vocabulaire de référence (médecine, agriculture...).

Par exemple, en médecine :

- Concepts : diagnostic, signe, structure anatomique ...;
- Relations : lieux (cancer-lieux-structure anatomique), synonyme, composition (corps-composition-tête) ...

Exemple d'ontologies génériques:

- CIDOC Conceptual Reference Model (Doerr (2003)) est une ontologie de haut niveau permettant l'intégration des informations pour le patrimoine culturel et leur corrélation avec des données de l'archive;
- Mikrokosmos (Mahesh (1996)) est une ontologie qui entre dans le projet Knowledge Base Machine Translation (KBMT)(Nirenburg, 1989) ayant comme but de fournir un dépôt de concepts avec une abstraction de la langue afin d'aider au processus.

Ontologie spécialisée

UNE *ontologie spécialisée* (ou ontologie de domaine) présente des concepts d'un certain domaine (les composants informatiques, l'immobilier, le droit, la génétique,etc.) tels qu'ils sont définis par le spécialiste de ce domaine. Une telle ontologie est appelée également *ontologie descriptive*, fournit des spécifications riches d'un domaine qui définit les concepts de ce domaine et les relations entre ces concepts.

Prenons quelques exemples d'ontologies de domaine:

- AOS (Agricultural Ontology Service) (Lauser and Sini, 2006) est un centre fédéré pour les termes, définitions et relations dans les secteurs agricoles et domaines d'utilisation connexe de la communauté agricole;
- TOVE (TOronto Virtual Entreprise) (Fox, 1992) est une ontologie qui vise à créer des modèles d'entreprise et modéliser leurs connaissances générales.

Il existe deux grands modèles informatiques d'ontologies: les modèles conceptuels et les modèles logiques. Les modèles conceptuels se basent sur l'approche de bases de données comme l'Entité-Relation, l'UML, etc. Quant aux modèles logiques, le type de modèle qui nous intéresse dans ce rapport, ils utilisent les approches (Intelligence Artificielle) IA et la représentation de connaissances comme les **réseaux sémantiques**, **graphes conceptuels**, **cadres sémantiques**, etc.

1.1.4 Les réseaux lexicaux, lexico-sémantiques!

DANS le domaine du TALN, les ressources fournissant des informations sur les mots ont une grande importance pour les applications telles que la génération du contexte, le système question/réponse, la désambiguïsation, le résumé automatique, la traduction automatique, etc. Ces informations peuvent contenir la catégorie grammaticale des mots, les associations sémantiques, syntagmatiques, paradigmatiques. Avec les autres mots, le(s) sens, gloses, exemple d'usage et inclusion dans de plus larges concepts.

Les réseaux lexico-sémantiques[4] (RLS) se différencient des dictionnaires et des thésaurus : les dictionnaires donnent des définitions aux différents sens d'un mot ainsi que la catégorie grammaticale et quelques exemples, les thésaurus contiennent seulement les relations telles que synonymie et antonymie alors que les RLS fournissent une plus riche information sémantique des mots et des relations plus variées telles que l'hyperonymie et la méronymie. Ceci fait de ces réseaux des structures ontologiques plus élaborées.

La nature des relations liant les termes dans ces réseaux dépend de la motivation derrière la construction de la ressource. Certains réseaux lexicaux sont appelés *réseaux sémantiques* quand ils fournissent une information sémantique riche.

De telles ressources doivent se présenter sous une forme interprétable par les machines dans le but d'être utilisées par des outils de TALN. Assez fréquemment, ces ressources prennent la forme de graphe orienté (GO) avec les termes sous forme de nœuds et les associations entre eux sous forme d'arcs.

Les *nœuds* représentent les éléments du lexique et peuvent être des termes simples ou composés, des informations linguistiques, des expressions, des locutions, des raffinement de sens (tête>anatomie, tête>chef, etc.)

Les *arcs* reliant les nœuds sont des relations orientées, typées (relations ontologiques, lexicales, rôles sémantiques...) et peuvent éventuellement être pondérées. En général, le poids d'une relation peut être inter-

[4]Dans ce mémoire nous parlerons de *réseaux lexico-sémantiques* (RLS) pour englober les différentes catégories

prété comme la force d'association de la relation. Les relations utilisées dans un réseau lexical sont définies par les constructeurs de la ressource selon le but visé.

Ce type de ressource est apparu dans le domaine de l'IA avec Quillian (1968) qui a initié un travail sur les réseaux sémantiques avec pour objectif de modéliser la capacité de la mémoire humaine et sa manière de traitement d'information. Il a ensuite été suivi par Minsky (1975), qui, lui, a conçu les *cadres* et les scripts qui sont une reformulation des réseaux sémantiques.

La construction de telles structures lexicales doit prendre en considération différents aspects comme le domaine de spécification de la ressource, les méthodes de construction, la qualité, le coût de la construction, la couverture etc.

Nous pouvons distinguer deux grandes catégories de réseaux:

- Les réseaux structurés comme une ontologie, par exemple WordNet, qui organisent l'information lexicale hiérarchiquement sous forme de classes et sous-classes. Ce qui donne une organisation plutôt pyramidale (figure 1.4);
- Les réseaux à modélisation relationnelle se distinguent des modèles ontologiques à organisation hiérarchique des concepts. Le modèle adopté par ce type de réseaux est plutôt un grand graphe de relations sémantiques, syntaxiques et lexicales reliant les nœuds qui représentent les entrées : mots, locutions, concepts, expressions, etc. (figure 1.5 et 1.6).

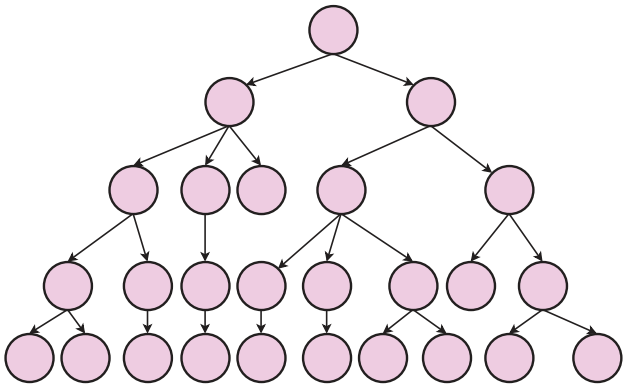

Figure 1.4 – Forme hiérarchique de structuration de réseaux lexicaux.

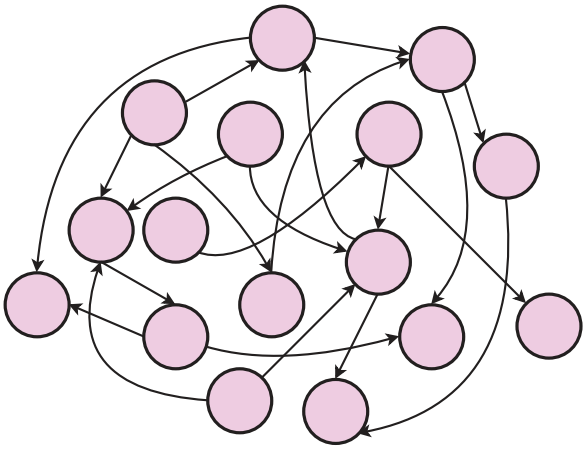

Figure 1.5 – Forme *plat de spaghetti* de graphes de réseaux lexicaux.

Ces réseaux peuvent dépendre fortement de la langue pour laquelle ils sont conçus, comme WordNet (Miller et al., 1990), ou être interlingues comme les réseaux avec le formalisme UNL (Universal Networking Language) (Uchida et al., 2005).

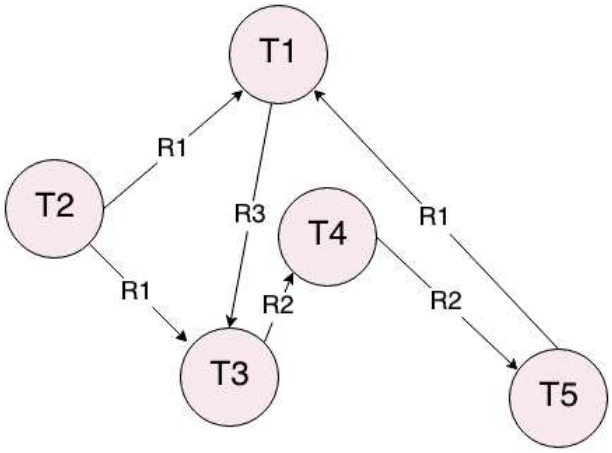

Figure 1.6 – Exemple de structure lexicale sous forme de graphe.

Parmi ces ressources et projets de construction de ces derniers nous pouvons citer Wordnet (Miller, 1995; Miller and Fellbaum, 2007) qui est un réseau lexical basé sur des synsets pouvant être globalement considérés comme des concepts. Quelques dérivations de WordNet sont EuroWordnet de Vossen (1998), une version multi-langues de Wordnet et WOLF de Sagot and Fiser (2008), une version française de Wordnet. Navigli and Ponzetto (2012) a construit BabelNet, un grand réseau lexical multilingue à partir de l'encyclopédie Wikipédia mais en se basant sur les co-occurences entre termes. Dans le domaine de l'intelligence artificielle, Cyc (Lenat, 1995) est un exemple de base de connaissances très redondante. Hownet (Dong et al., 2010) est un autre exemple d'une grande base de connaissances bilingue (anglais et chinois) contenant des relations sémantiques entre les formes de mots, les concepts et les attributs. FrameNet (Baker et al., 1998) est un projet de l'Institut International d'Informatique à Berkley (Californie) qui a fourni une ressource électronique se basant sur la théorie du sens appelée la sémantique des cadres (semantic frames). MindNet (Richardson et al., 1998) est un projet de représentation de connaissance qui utilise l'analyseur de large couverture de Microsoft Research pour construire des réseaux sémantiques à partir des dictionnaires, encyclopédies et textes. ConceptNet (Havasi et al., 2009) est un réseau sémantique basé sur les informations de la base de donnée OMCS (Open Mind Common Sense). Il est présenté comme un graphe orienté où les nœuds sont les concepts et les arêtes sont les relations entre les concepts. Ce réseau représente un en-

semble de phrases de langue naturelle étroitement reliées et qui peuvent être des noms, des verbes, des adjectives ou des expressions. JeuxDeMots (Lafourcade, 2007) est un projet se composant de multiples jeux en ligne (GWAP) et un outil contributif permettant d'acquérir avec externalisation ouverte une base de connaissance sous forme de réseau lexico-sémantique avec des relations sémantiques orientées, pondérées et pouvant être sémantiquement annotées. Les nœuds dans ce réseau peuvent être des entrées lexicales, des mots, des expressions, des phrases, etc.

Dans les parties suivantes nous allons présenter en détail quelques unes de ces ressources.

WordNet

WORDNET, (Miller et al., 1990) développé par le Cognitive Science Laboratory (Princeton University) sous la direction de George A. Miller et Christiane Fellbaum est un réseau lexical qui fournit des informations sémantiques riches sur les mots de la langue anglaise.

Ce projet a débuté en 1985, quand un groupe de psychologues et linguistes de l'Université Princeton a décidé de développer une base de données lexicales. L'idée principale était de fournir un outil qui aide les machines à explorer les dictionnaires en se basant sur des concepts à la place de l'alphabet. Au fil des travaux, les objectifs ont évolué pour finalement donner naissance à WordNet.

Ce réseau couvre le domaine de la langue générale anglaise en présentant le sens des mots dans différents domaines. La dernière version contient 155 287 mots. La construction de WordNet a été faite dans un contexte psycho-lexicographique et est influencée par des travaux sur le processus cognitif d'accès au lexique (Miller, 1995, 1985).

WordNet est un réseau lexical où les nœuds sont les *synsets* et les arcs représentent les relations entre ces synsets (Miller and Fellbaum (2007)). Un *synset* dans WordNet est *un ensemble de synonymes; ensemble de mots que l'on peut interchanger dans un contexte donné sans altérer la valeur de vérité de la proposition dont ils font partie*.

Ces synsets comportent un code formé à partir de leur position dans la base de données et d'une lettre indiquant leur catégorie grammati-

cale (adjectif, nom, adverbe ou verbe), l'ensemble de synonymes qui définit le synset, une définition (glose), et éventuellement un exemple d'utilisation. Ci-dessous, quelques exemples sont présentés :

```
02383458-n : car auto automobile machine motor-
car | 4-wheeled motor vehicle; usually propelled
by an internal combustion engine; "he needs a car
to get to work"
```

```
00136205-a : mouth-watering savory savoury tasty
| pleasing to the sense of taste
```

```
00191458-r : bewilderedly | in a bewildered man-
ner
```

```
01118553-v : compile | use a computer program to
translate source code written in a particular pro-
gramming language into computer-readable machine
code that can be executed
```

L'architecture prend en compte le fait que le même mot peut avoir différents sens exprimant différents concepts. Ainsi, l'organisation des informations est faite par sens et non par "formes" de mots.

WordNet utilise une matrice lexicale pouvant être illustrée par un tableau où les lignes présentent les sens et les colonnes les mots. Une cellule remplie indique que cette forme de mot peut être utilisée pour exprimer ce sens. Par exemple, le mot *livre* peut avoir comme entrée le sens correspondant à l'*argent* ou le sens correspondant à la *lecture* dans sa matrice. S'il existe plusieurs entrées pour la même colonne alors ce mot est polysémique et si plusieurs entrées existent pour une même ligne alors il existe des synonymes qui expriment ce sens.

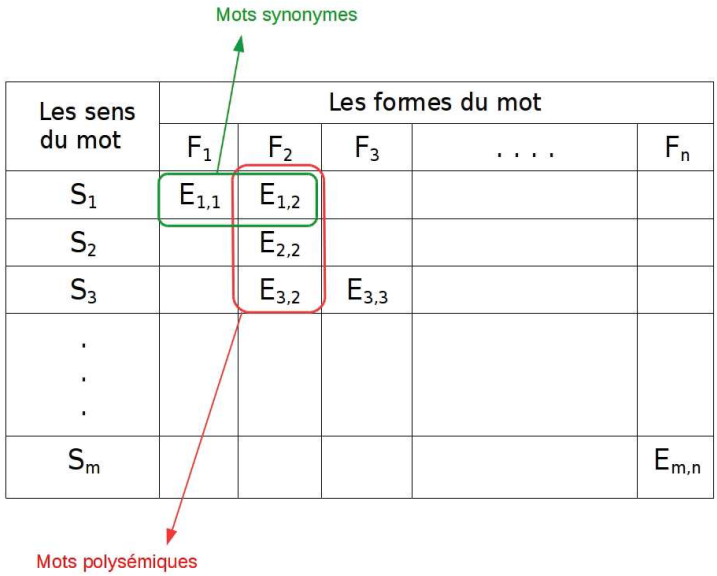

Figure 1.7 – Matrice lexicale les S_i sont les sens du mot et les F_j les formes du mot. Les entrées $E_{1,2}$, $E_{2,2}$ *et* $E_{3,2}$ ayant une forme F_2 et trois sens distincts S_1, S_2 et S_3 représentent les sens d'un terme polysémique. Les entrées $E_{1,1}$ et $E_{1,2}$ ayant un sens S_1 et deux formes F_1 et F_2 représentent des synonymes.

La conception de WordNet a été inspirée par les théories de la mémoire linguistique humaine. Il est composé d'ensembles de synonymes (quasi-syno-nymes) appelés *synsets* (synonym set), où chaque terme est regroupé en classes d'équivalence sémantique (synonymes, quasi-synonyme). Chaque ensemble de synonymes représente un concept (sens) particulier. Chaque terme appartient de plus à une ou plusieurs catégories lexicales données (nom, verbe, adverbe, adjectif) organisées par WordNet dans des sections séparées. Un terme peut appartenir à plusieurs synsets et à plusieurs catégories lexicales. Chaque synset est accompagné par une description du sens qu'il représente ("glose") et des exemples d'usage (figure 1.8).

Nom

- `S: (n) rabbit, coney, cony (any of various burrowing animals of the family Leporidae having long ears and short tails; some domesticated and raised for pets or food)`
- `S: (n) lapin, rabbit (the fur of a rabbit)`
- `S: (n) rabbit, hare (flesh of any of various rabbits or hares (wild or domesticated) eaten as food)`

Verbe

- `S: (v) rabbit (hunt rabbits)`

Figure 1.8 – Exemple d'entrée *rabbit* (lapin) qui appartient à deux différentes catégories lexicales (nom et verbe) avec des gloses pour chaque sens.

Les noms et les verbes sont organisés hiérarchiquement par des relations d'hyperonymie dites *is-a*. Ainsi l'un des sens du terme *rabbit* (lapin) est organisé comme illustré ci-dessous (figure 1.9). Notons que les termes sur le même niveau représentent des membres du synset.

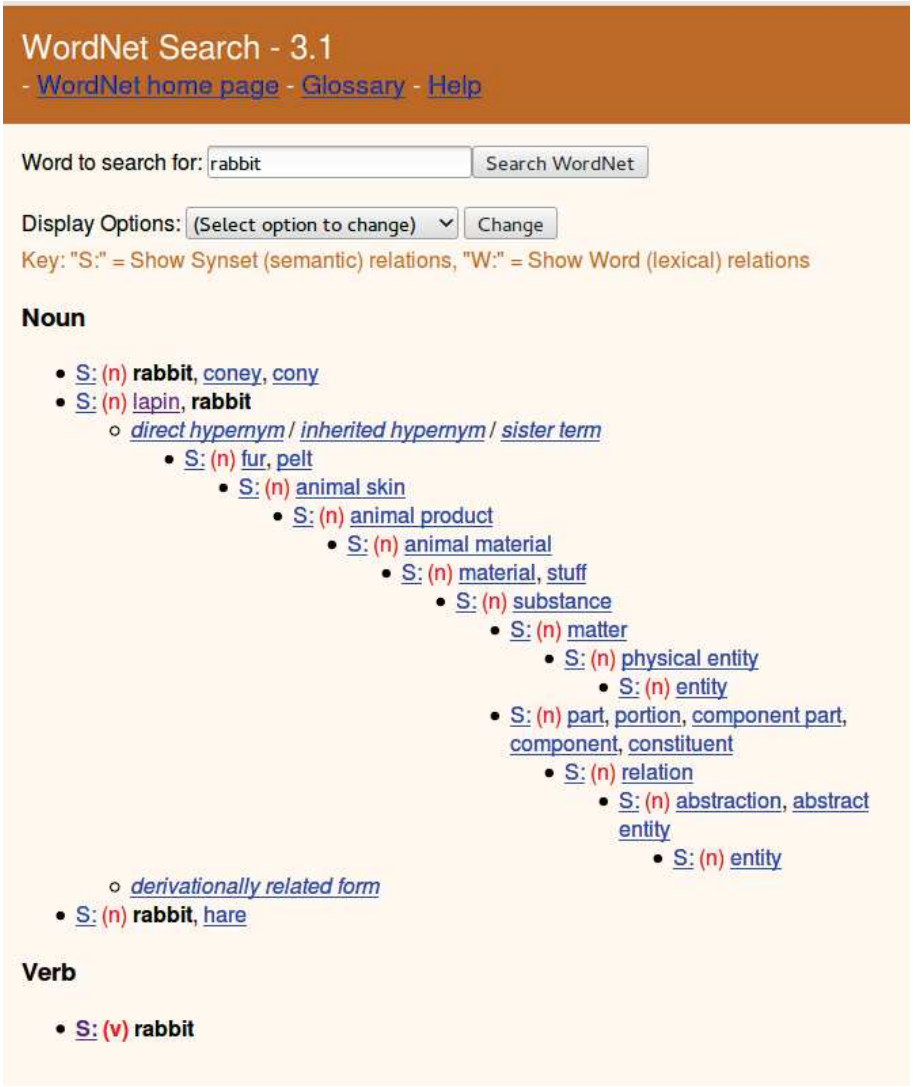

Figure 1.9 – Exemple d'organisation hiérarchique d'un des sens de l'entrée *rabbit* (lapin) dans WordNet en ligne.

Les ensembles de synonymes sont reliés par des relations sémantiques : généricité, spécificité, synonymie, antonymie, méronymie et holonymie. La sémantique des relations peut varier d'une catégorie lexicale à une autre comme expliqué ci-dessous.

Pour les noms

- hyperonymes: *A* est un hyperonyme de *B* si tout *B* est une sorte de *A* (lapin est un hyperonyme d'Angora);
- hyponymes: A est un hyponyme de B si tout A est une sorte de B (*Angora* est un hyponyme de *lapin*);
- méronyme: A est un méronyme de B si A est une partie de B (*queue* est un méronyme de *lapin*);
- holonyme: A est un holonyme de B si B est une partie de A (*forêt* est un holonyme de *lapin*);
- termes coordonnés: A est un terme coordonné de B si B et A partagent un hyperonyme (*rat* est un terme coordonné de *lapin*, et *lapin* est un terme coordonné de *rat*).

Pour les verbes

- hyperonyme: le verbe A est un hyperonyme du verbe B si l'activité B est une sorte de A (*chasser* est un hyperonyme d'*attraper*);
- troponyme: le verbe A est un troponyme du verbe B si l'activité A fait B en quelque manière. Plus clairement, le verbe A précise l'action décrite par le verbe B (*décapiter* et *égorger* sont des manières de *tuer* alors troponymes de *tuer*);
- implication: le verbe A est impliqué par B si en faisant B, A est obligatoirement fait (*dormir* est impliqué par *ronfler*);
- termes coordonnés: les verbes partagent un hyperonyme (*chuchoter grommeler*, *bavarder*, *susurrer* et *crier* sont des termes coordonnés avec comme hyperonyme *parler*).

Puisque le terme *mot* est généralement utilisé pour désigner et la suite finie d'unités de la chaîne parlée, et le concept qui lui est associé, le problème d'une association lexicale sensible à une confusion terminologique se pose. Dans WordNet, pour réduire cette ambiguïté, ils ont utilisé *forme* pour désigner la forme physique ou l'inscription, et *usage de terme* pour parler du concept lexicalisé qui peut être décrit

par une forme. D'où la nécessité de présenter la correspondance entre les formes et les sens (Miller, 1985) et d'évoquer l'idée que différentes catégories syntaxiques des mots peuvent avoir différents types de correspondance.

La construction manuelle de ce réseau lexical (coût de 3 millions de dollars) a commencé en 1985 et a abouti à une première version en 1993. La version la plus récente pour WINDOWS est la 2.1 sortie en Mars 2005, pour les systèmes UNIX la version 3.0 est sortie en Décembre 2006. Quant à la version 3.1, elle est disponible exclusivement en-ligne.

Hownet

HOWNET (Dong and Dong, 2006) est un projet de recherche lancé en Chine qui avait comme but de créer un réseau sémantique se basant simultanément sur des entrées lexicales et conceptuelles des lexiques chinois et leur équivalent en anglais. Ce réseau s'appuie sur une base de connaissances conceptuelle et a été construit manuellement avec une structure bien définie ce qui le rend parfaitement utilisable par des machines satisfaisant ainsi l'objectif principal du projet .

HowNet se base sur deux idées sous-jacentes:

- La composition: les concepts peuvent être composés pour former d'autres concepts. Les relations *inter-conceptuelles* peuvent exprimer cette notion de "ensemble/partie" (figure 1.10);
- L'évolution: les entités ont des propriétés qui peuvent ne pas être partagées par spécialisation mais exprimées en utilisant les relations *inter-attributives* (figure 1.11).

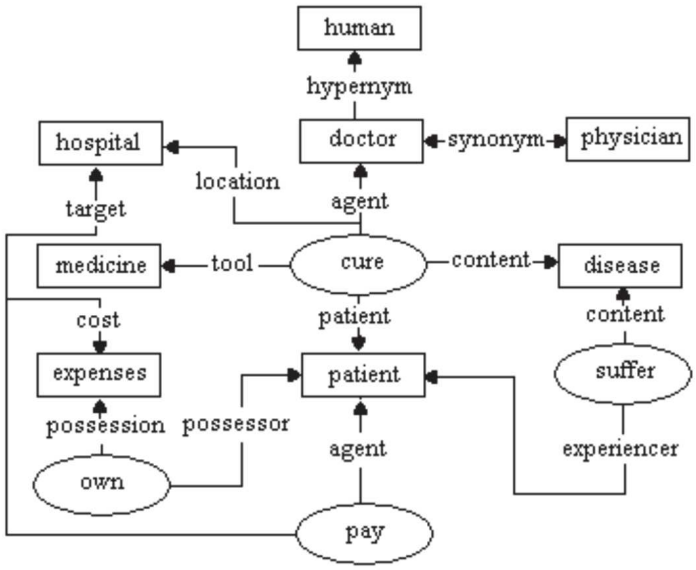

Figure 1.10 – Exemple de relations inter-conceptuelles dans le thème de la médecine (Dong et al. (2010)).

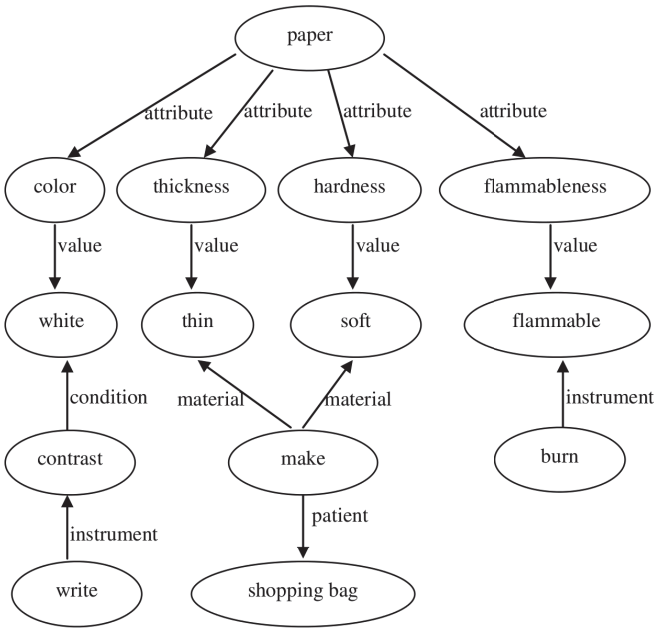

Figure 1.11 – Exemple de relations inter-attributives pour *paper* (papier) Dong et al. (2010).

HowNet a été conçu selon une méthode *constructive*. Un ensemble de concepts les plus fondamentaux a été identifié au début, d'autres entités de haut-niveau sont composées après en combinant ces concepts de base et d'autres entités du même niveau préalablement disponibles. Des relations appropriées sont rajoutées par la suite entre ces concepts.

Cette méthode ascendante est opposée au modèle WordNet où les mots sont organisés d'une manière descendante.

Par exemple le concept lexical *surgeon*|医生 correspondant à *chirurgien* a la définition sémantique suivante dans HowNet:

surgeon|医生≡{human|人: HostOf={Occupation|职位} domain={medical|医}},{doctor|医治:agent={~}}

* *human* correspond à *humain*, *HostOf* à *Hôte*, *Occupation* à *Occupation/métier*, *domain* à *domaine*, *medical* à *médical*, *doctor* à *docteur* et *agent* à *agent* *

Ce concept peut être défini comme suit : "un chirurgien est un humain avec un métier dans le domaine médical et qui agit selon son activité de médecine". HowNet utilise la définition ci-dessus non seulement pour *chirurgien* mais aussi pour tout le corps médical en général, allant des aide-soignants, aux infirmières, internistes et neurologues.

La base de données de HowNet est une base de *sémèmes* consciencieusement construite à la main. Un sémème est une unité de sens atomique. Exemples de sémèmes : *humain* ou *boisson*. Les concepts comme *architecte* ou *chercheur* comme l'exemple ci-dessus peuvent être renseignés comme une combinaison de sémèmes comme *humain*.

L'hypothèse de HowNet est qu'un ensemble fermé de sémèmes doit être suffisant pour décrire un ensemble ouvert de concepts. Une hypothèse originale et évolutionniste mais qui incite à poser la question "comment mettre en place de nouveaux concepts d'une manière efficace? De quelle manière la méthode *constructive* va-t-elle prendre en considération ces ajouts?".

HowNet a utilisé l'ensemble de caractères de la langue chinoise comme un point de départ d'identification de sémèmes. La plupart des caractères chinois représentent plutôt un concept de base qu'une lettre alphabétique vide de sens comme dans d'autres langues. Prenons comme exemple le concept *Enseignant*. Dans HowNet, ce concept

est présenté comme la combinaison des sémèmes *humain* (entité), *enseigner* (action) et *éducation* (entité). En conséquence la fiche du concept "enseignant" va contenir:

- Hyperonyme : *humain*
- Attribut : *éducation*
- Relation : *Agent* vers *enseigner*

La question se pose ici autour de la possibilité d'étendre HowNet vers d'autres langues sachant qu'il est basé sur le système des alphabets chinois.

BabelNet

BABELNET (Navigli and Ponzetto, 2012) est un RLS multilingue contenant plus de 6 millions de concepts. Cette ressource a été construite par fusion de la grande encyclopédie multilingue du web *Wikipedia* et du réseau lexical populaire *WordNet*. Par la suite, le comblement des lacunes lexicales pour les langues faibles en ressource se fait par traduction automatique des concepts et l'extraction automatique de sens et de relations de sens inter-langues (figure 1.12). Ce qui donne un dictionnaire encyclopédique qui fournit plus de 13 millions de *Babel synsets* sous forme de 6 millions de concepts et 7 millions d'entités nommées, lexicalisées en différentes langues et liées par plus de 262 millions de relations lexico-sémantiques multilingues (figure 1.13).

Ce réseau est structuré de façon similaire à WordNet, chaque *Babel synset* représente un certain sens et contient tous les synonymes, appelés *Babel sens*, qui expriment ce sens en différentes langues.

Le réseau fournit, par exemple, de la connaissance lexicale pour le concept *piqûre* comme *injection*, avec sa partie de discours, sa définition et son ensemble de synonymes dans différentes langues, ainsi que de la connaissance encyclopédique, entre autres *piqûre* dans le sens de *piqûre d'insecte* et *piqûre de couture* et leurs définitions dans plusieurs langues. Grâce aux relations sémantiques, il est possible de savoir que *piqûre* dans le sens d'injection (médecine) est *une méthode instrumentale utilisée pour introduire dans l'organisme une substance à*

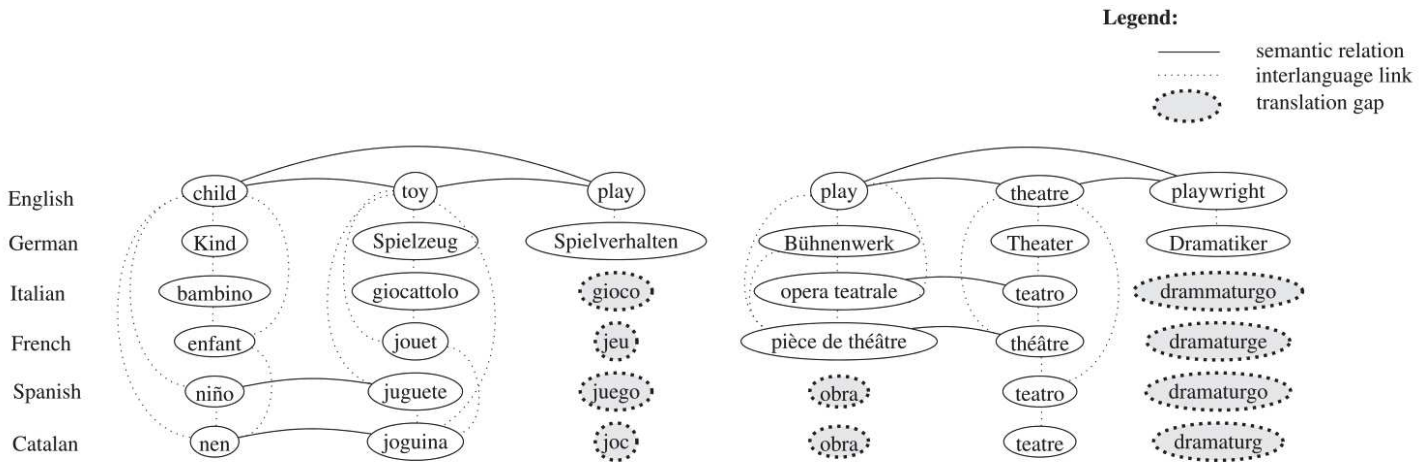

Figure 1.12 – La traduction automatique des synsets dans BabelNet dans le but de remplir les lacunes lexicales comme les silences de traductions spécialement pour les langues avec ressources limitées (Navigli and Ponzetto, 2012).

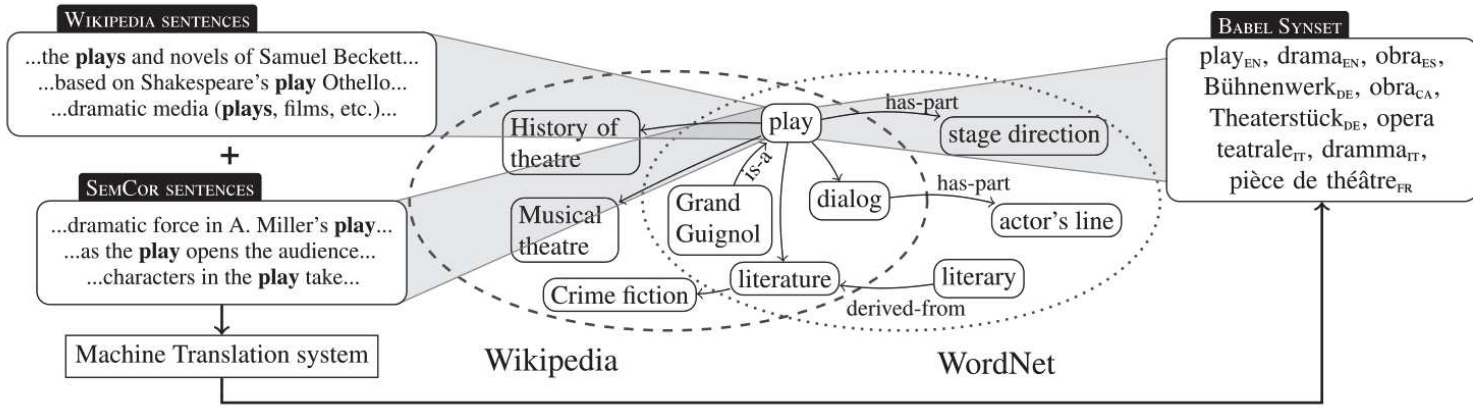

Figure 1.13 – Vue globale du terme *play* (pièce de théâtre). Les arcs sans notation sont obtenus à partir des liaisons des pages Wiki (exemple *Play* est relié à *Musical theatre*), alors que les arcs avec notation sont ceux obtenus à partir de WordNet (exemple *Dialog has-part actor's line* (Dialogue a comme partie réplique)) (Navigli and Ponzetto, 2012).

visée thérapeutique ou diagnostique alors que piqûre (insecte) est relié à *Piqûre d'insecte - piqûre - piquer - mordre* et défini comme suit: *Une piqûre d'insecte peut être causée lorsqu'un insecte est agité et qu'il cherche à se défendre, grâce à ses mécanismes de défense, ou lorsqu'il cherche à collecter le sang de l'individu qu'il a mordu*.

BabelNet, comme mentionné ci-dessus, est le résultat du croisement de WordNet et Wikipédia. Lors de la construction, les auteurs ont pris de WordNet tous les sens de mots (en guise de concepts) et toutes les liaisons sémantiques et lexicales entre les synsets (en guise de relations). En revanche, ils ont pris de Wikipédia toutes les entrées encyclopédiques comme les pages Wiki (en guise de concepts) et des relations non-typées à partir des liens hypertextes.

FrameNet

FRAMENET (Baker et al., 1998) est un projet développé à l'Institut International de Berkeley qui a produit une ressource lexicale électronique se basant sur l'anglais, en grande partie sur le Corpus National Britannique (BNC). Cette ressource est fondée sur la théorie des *cadres sémantiques* (frame semantics (Fillmore, 1976)).

Ce projet produit les définitions de ces cadres pour quelques milliers d'éléments lexicaux annotés sémantiquement des corpus d'anglais. Ces définitions sont générées à partir des annotations sémantiques manuellement étiquetées des phrases-exemples extraites à partir des corpus de texte par des lexicographes et des linguistes.

L'objectif principal de FrameNet est d'encoder de la connaissance sémantique sous une forme exploitable par les machines, ce qui a été effectué essentiellement en utilisant une méthode manuelle soutenue par quelques outils de TALN.

Les cadres sont bien connus dans le domaine du TALN. Un cadre est considéré comme un *ensemble d'attributs, de valeurs associées et de contraintes* (Rich and Knight, 1999).

Dans FrameNet, un cadre est une structure conceptuelle qui décrit un type particulier de situation, objet ou évènement et les participants. Un cadre est constitué d'*éléments de cadre* qui décrivent les sous-parties de ce cadre. Il y a aussi les *unités lexicales* qui sont des mots évoquant

un concept et sont représentés par des frames.

Exemple: le frame *Revenge* (vengeance)

Unités lexicales: "revenge" "avenge" "retaliation" etc (vengeance, venger, représailles etc)
Élément du frame:
Agent – the avenging person
Offender – original offender and victim of the retaliation
Injured Party – victim of the original offense
Injury – the original offense
Punishment – the measure taken by the avenger
Exemples de phrase:
[Sam's brothers$_{\text{Agent}}$] AVENGED [him$_{\text{Injured Party}}$].
The team sought REVENGE for their 4-1 defeat last month.
Jo had the affair as a kind of REVENGE against Pat.
The team took REVENGE with a resounding victory.
The military decapitated several prisoners to AVENGE the blood of the fallen soldiers.

La base de données de FrameNet est constituée de trois éléments:

1. *Lexique:* consiste en un ensemble d'entrées. Une entrée pour chaque unité lexicale dans FrameNet. Une entrée est composée de:
   (a) Une définition similaire à celle d'un dictionnaire;
   (b) Des formules pour la réalisation syntaxique utiles pour générer les variations de "une personne joue au ballon" comme "une personne joue au ballon dans le jardin" ou " une personne joue au ballon avec professionnalisme" ou même "une personne joue au ballon dans le jardin avec professionnalisme". L'entrée de "jouer" par exemple va pointer vers les formules qui l'incluent;
   (c) Des liens vers les frames en vigueur impliquant cette unité lexicale;
   (d) Des liens vers des phrases d'exemples dans le corpus pour les différentes réalisations.

2. *Bases de frames*: contient les différents frames. Chaque frame est décrit par le nom, la description de ses éléments de frame, les unités lexicales évoquant ce frame et les attributs;

3. *Phrases-exemples annotées*: ensemble d'exemples de phrases annotées avec des informations syntaxiques et sémantiques.

Pour produire la base FrameNet de représentation en cadre sémantique, il faut passer par quatre étapes:

- La préparation qui consiste à la génération des descriptions initiales des patrons syntaxiques et sémantiques à utiliser lors de l'annotation et des requêtes de corpus;
- L'extraction de bonnes phrases exemples pour former le corpus;
- L'annotation manuelle;
- L'écriture de l'entrée par la construction d'une base de représentation sémantique et lexicale basée sur l'annotation et les autres données.

Ces étapes, évidemment, rendent le processus susceptible d'être très lent et très coûteux. Et comme c'est prévisible avec ce type de construction et de maintenance, la qualité de la ressource est relativement assurée mais le rythme de la construction est très lent et celui de l'évolution est considérablement figé.

MindNet

MINDNET (Richardson et al., 1998) est une initiative du groupe de recherche en TALN de Microsoft dans le domaine des structures lexicales. C'est un ensemble de relations sémantiques automatiquement extraites des données textuelles par un analyseur à large couverture. C'est le même analyseur dans les applications Microsoft Office. Il est appliqué sur les données des dictionnaires informatisés qui sont principalement constitués par des mots et des définitions. Cette extraction est un processus totalement automatisé dans le sens où les constructeurs de MindNet n'ont pas vu la nécessité de mettre en place un processus

d'inspection des informations pour assurer la pertinence et la qualité de la ressource.

Cette ressource contient 24 relations sémantiques comme hyperonyme, partie, temps, taille, lieu, etc. La construction de ce réseau implique l'extraction des *semrels* (pour dire relation sémantique) des phases. Ces *semrels* sont très structurées. Le processus d'extraction automatique les extrait à partir des définitions ou des phrases exemples et en produit une structure hiérarchique, représentant la définition complète ou la phrase de laquelle elles sont issues (figure 1.14). Ces structures sont stockées en l'intégralité dans MindNet afin d'être utilisées plus tard (Syed et al., 2010).

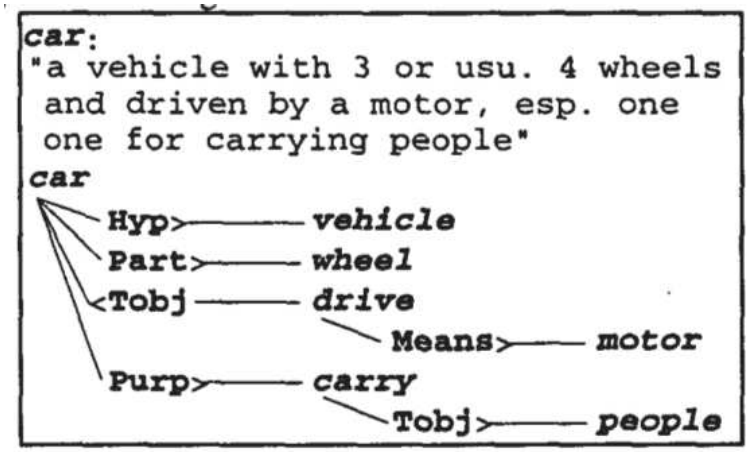

Figure 1.14 – La structure des semrels pour la définition de *car* (voiture) Richardson et al. (1998).

En plus de cette extraction, d'autres méthodes d'inférences sont appliquées à cette structure afin d'extraire des propositions potentiellement utiles. La plus importante est *l'inversion*. En utilisant l'exemple de *car*, une structure où l'entrée *drive* est liée à *car* sera extraite. De cette manière, le réseau est amélioré en ajoutant des associations pertinentes entre les mots qui se trouvent dans ces semrels. La construction de MindNet contient aussi une méthode pour pondérer[5] les chemins entre les entrées des semrels (figure 1.15).

[5]Les chemins semrels sont automatiquement affectés de poids qui reflètent leur importance. Les pondérations dans MindNet sont basées sur le calcul de la probabilité moyenne de vertex, ce qui donne la préférence aux relations sémantiques survenant

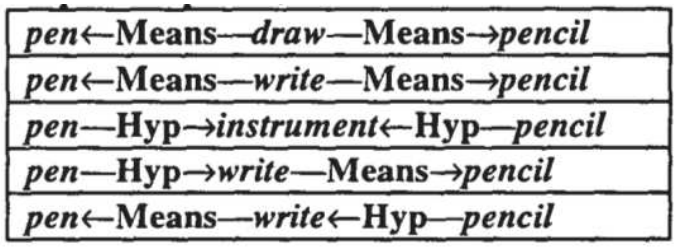

Figure 1.15 – Exemple de chemins *semrels* les plus fortement pondérés entre *pen* (stylo) et *pencil* (crayon) (Richardson et al. (1998)).

MindNet est une ressource lexicale qui se distingue par son processus totalement automatisé d'extraction de relations sémantiques principalement à partir des dictionnaires informatisés, et quelques idées originales d'inférence sur les données collectées pour améliorer la structure de connaissances lexicales (Richardson, 1997).

VerbNet

VERBNET est un lexique générique hiérarchisé de verbes avec des informations sémantiques et syntaxiques des verbes anglais, utilisant les classes de verbes de Levin (1993) pour construire les entrées lexicales systématiquement Karin et al. (2006). Ce lexique est mis en correspondance avec différentes ressources populaires comme WordNet, Xtag (Group, 2006) et FrameNet. Le premier niveau de la hiérarchie est constitué par les classes originales de Levin, avec chaque classe raffinée par la suite pour représenter les différences sémantiques et syntaxiques au sein d'une classe. Chaque nœud de la hiérarchie est caractérisé en extension par son ensemble de verbes et en intention par une liste d'arguments de ces verbes et des informations syntaxiques et sémantiques les concernant.

Chaque verbe dans ce lexique est décrit par un ensemble de membres, de rôles thématiques pour la structure prédicat-argument de ces membres (figure 1.16), de restrictions sélectives sur les arguments (figure 1.17), et de cadres consistant en une description syntaxique et de prédicats sémantiques temporisés.

avec fréquence moyenne. Ces approches sont décrites en détail dans (Richardson, 1997).

Actor, Agent, Theme, Patient, Asset, Attribute, Beneficiary, Cause, Destination, Experiencer, Instrument, Location, Material, Patient, Product, Recipient, Source, Stimulus, Time, Topic

Figure 1.16 – Désignations de rôles thématiques utilisés dans VerbNet (Karin et al., 2002).

concrete, abstract, location, organization, currency, communication, phys_obj, human, animate, body_part, int_control, ...

Figure 1.17 – Exemples de restriction sélective utilisée dans VerbNet (Karin et al., 2002).

Le Réseau lexico-sémantique de JeuxDeMots

LANCÉ en septembre 2007, JeuxDeMots[6] (Lafourcade, 2007) est un projet comportant plusieurs GWAP[7] et un outil contributif, visant à construire un grand réseau lexico-sémantique.

Le RLS construit est composé de termes (nœuds) et de relations typées et pondérées (arcs) selon un modèle initialement présenté par Collins and M.R. (1969) et davantage explicité par Polguère (2008). Il y a plus de 50 types de relations et chaque occurrence de relation est pondérée indiquant une force d'association entre les termes reliés (Lafourcade and Joubert, 2009). La pondération d'une relation est faite en fonction du nombre de paires de joueurs qui l'ont proposée comme indiqué dans Lafourcade and Joubert (2009). Une occurrence de relation peut éventuellement avoir un poids négatif indiquant dans ce cas que la relation est fausse bien que pertinente (ex: une autruche ne vole pas). Une relation peut donc être considérée comme un quadruplet

[6] http://www.jeuxdemots.org/

[7] En TAL, d'autres jeux accessibles via le web existent, comme Open Mind Word Expert Mihalcea and Chklovski (2003) qui vise a créer un grand corpus étiqueté sémantiquement avec l'aide des utilisateurs du Web ou comme SemKey Marchetti et al. (2007) qui utilise WordNet et Wikipédia pour désambiguïser les formes lexicales se référant à des concepts, ainsi identifiant les mots-clés sémantiques.

⟨terme source, terme cible, type et poids de la relation⟩. Entre deux mêmes termes, plusieurs relations de types (et de poids) différents peuvent exister.

Ci-dessous un exemple d'entrée obtenu par l'interface de Diko[8] le dictionnaire contributif de JDM.

[8] http://www.jeuxdemots.org/diko.php

Figure 1.18 – Exemple partiel de l'entrée du terme *lapin* pour l'usage d'animal de Diko. Le terme *lapin* pouvant aussi désigner la *fourrure*, la *viande* ou un *rendez-vous manqué*.

En août 2015, le réseau de JDM contient environ **561 048** termes et plus de **21 813 253** relations. Plus de **10 532** termes ont quelques raffinements[9] (usages variés) pour un total d'environ **32 668** usages. Bien que JDM ait des relations ontologiques, il ne constitue pas une ontologie proprement dite avec des concepts ou des termes soigneusement hiérarchiques. Un terme donné peut avoir une collection substantielle d'hyperonymes qui couvre une large partie de la chaîne ontologique jusqu'aux concepts supérieurs. Par exemple:
`hyperonyme(chat)=`
`{félin,mammifère,être vivant,animal de compagnie,`
`vertébré,...}.`
Dans la liste d'hyperonymes précédente, nous avons omis les poids pour simplifier, mais en toute généralité, les termes les plus *lourds* sont ceux considérés par les utilisateurs comme les plus pertinents.

Graphes conceptuels (ConceptNet)

LES graphes conceptuels ont été proposés par John F.Sowa dans (Sowa, 1984) poursuivi par ses autres travaux (Sowa, 1988) et (Sowa, 1992). Ils représentent un système logique basé sur les graphes existentiels de Charles Sanders Peirce (Peirce, 1931) et les réseaux sémantiques de l'IA.

Un graphe conceptuel est un graphe fini connecté et bipartite qui se compose de concepts et relations conceptuelles. Chaque relation conceptuelle a un ou plusieurs arcs, dont chacun est lié à un concept.

Gaines (1993) a mentionné que les graphes conceptuels peuvent servir de base à un système opérationnel de représentation de connaissances pouvant manier aussi bien la langue naturelle, la logique et les langages formels que les raisonnements qui leur sont associés. Le modèle des graphes conceptuels est assez expressif pour représenter et raisonner sur des connaissances complexes et structurées. Ainsi, il est possible

[9]Un terme marqué comme polysémique par les utilisateurs a plusieurs *usages de mot* qui peuvent se différencier fondamentalement des sens de mots classiquement définis. Un usage donné peut avoir lui même plusieurs *raffinements* et l'ensemble d'usage va prendre la forme d'un arbre (figure 2.9) Par exemple, *voler* peut être avoir comme raffinements malhonnêteté, déplacement aérien, etc. *voler>malhonnêteté* peut être raffiné en *voler>malhonnêteté>plagier*,
voler>malhonnêteté>léser, *voler>malhonnêteté>s'approprier*.

d'utiliser ce formalisme pour une large gamme de domaines d'application. De plus, ce formalisme permet une représentation précise des connaissances.

Un graphe conceptuel (GC) est un graphe biparti orienté. Il contient donc deux sortes de nœuds : concept et relation. Les nœuds sont reliés par des arcs orientés qui sont contraints de relier exclusivement un concept à une relation, mais l'existence des nœuds concepts isolés est possible.

Exemple `" la clé est sous le paillasson"`
- représentation graphique : 2 concepts (clé, paillasson), 1 relation (sous)
- représentation linéaire:
`[concept]/(relation) : [clé]->(sous)->[paillasson]`

ConceptNet est une base de connaissance lexicale de l'équipe NLP de l'Institut de Technologie de Massachusetts qui vise à capturer de l'information de *sens commun* et les relations entre les éléments de cette information. Cette base de connaissance libre intègre un ensemble d'outils de traitement automatique des langues naturelles fonctionnelles sur plusieurs tâches de raisonnement textuel. *Sens commun* fait référence à l'information sémantique qui permet aux humains de comprendre les événements triviaux du quotidien (Liu and Singh, 2004).

Exemple : pour comprendre une phrase comme "j'ai emprunté *Harry Potter* pour un mois" il faut posséder les informations de sens commun suivantes:

- *Harry Potter* est le nom d'un livre;
- les gens empruntent les livres pour les lire;
- le livre peut être très probablement emprunté à la bibliothèque ou à une personne;
- le livre doit être retourné dans la période d'un mois.

Les fondateurs de ConceptNet sont partis sur l'idée que, même si les approches statistiques et basées sur les mots clés ont obtenu un certain succès en assistant des applications telles que l'extraction d'information, la fouille de données et le TALN, elles restent faibles en compréhension.

Et pour faire des progrès sur ce point, il est exigé d'avoir des quantités importantes de connaissances qui donneront aux systèmes la capacité d'avoir une compréhension plus approfondie des données textuelles.

Contrairement à WordNet qui se veut optimisé pour la catégorisation lexicale et la détermination des similarités inter-mots, et à CycNet pour le raisonnement de logique formalisée, ConceptNet est optimisé pour faire des inférences pratiques et basées sur le contexte.

1.2 Acquisition des ressources

SACHANT que le TALN est une discipline frontière entre la linguistique, l'informatique et l'intelligence artificielle et que les ressources lexicales ont une importance indéniable dans ce domaine, énormément de recherches ont été menées dans le but de construire et d'améliorer ce type de ressources.

La qualité des ressources lexicales existantes n'est pas pleinement satisfaisante lors de leur utilisation par les applications du TAL dans le sens où elles ne sont pas sans erreurs et ont toujours besoin d'une validation humaine, et ceci malgré les différentes méthodes automatiques qui ont été conçues afin de les construire.

Il existe de multiples méthodes pour construire un réseau lexical mais elles ne présentent pas toutes les mêmes caractéristiques de qualité des données, de coût et de délai de réalisation.

1.2.1 Acquisition manuelle

La plupart des ressources existantes ont été construites à la main par des experts ou des linguistes, comme dans le cas de WordNet. Bien entendu, quelques outils sont généralement utilisés pour la vérification de la cohérence. Cyc est un exemple de base de connaissances très redondante ayant demandé un effort manuel particulièrement important. Il en est de même de HowNet.

Certes la construction manuelle de telles ressources minimise le taux d'erreur et augmente la pertinence. Cependant, cette approche reste trop coûteuse en temps en prix et en effort humain. De plus, des critères ou

des règles doivent être clairement définis pour éviter les collisions entre les différents points de vue.

1.2.2 Acquisition (semi-)automatique

Les approches automatisées d'acquisition de données sont des approches non supervisées qui extraient de la connaissance à partir de corpus, texte brut, listes, etc. Celles semi-automatisées sont semi-supervisées car les données sont habituellement annotées. EuroWordnet, une version multi-langues de Wordnet et WOLF, une version française de Wordnet, ont utilisé des croisements automatiques de Wordnet avec d'autres ressources lexicales suivis d'une vérification manuelle partielle. Quant à BabelNet, c'est un grand réseau lexical multilingue construit à partir de l'encyclopédie Wikipédia mais en se basant sur les co-occurences entre termes.

Les approches entièrement automatisées sont généralement limitées à la co-occurrence des termes car l'extraction des relations sémantiques précises entre termes à partir d'un texte reste difficile.

Les ressources acquises à partir de corpus ne sont pas aussi complètes et précises que les ressources construites à la main mais peuvent apporter des informations qui ne sont souvent pas possibles à construire par des moyens humains.

1.2.3 Acquisition par externalisation ouverte

Parallèlement aux stratégies automatiques ou manuelles, les approches contributives dites par externalisation ouverte[10] sont de plus en plus populaires, par le fait qu'elles sont à la fois peu coûteuses à mettre en

[10]Externalisation ouverte ou Crowdsourcing signifie l'externalisation par une organisation, via un site web, d'une activité auprès d'un grand nombre d'individus dont l'identité est le plus souvent anonyme (Lebraty, 2007). Selon Wikipédia *Le terme crowdsourcing est un néologisme sémantiquement calqué sur l'outsourcing (externalisation). La traduction littérale de crowdsourcing est « approvisionnement par la foule, ou par un grand nombre [de personnes] », mais l'expression ne reflète pas vraiment le sens anglo-saxon du terme ; « Impartition à grande échelle » ou encore « externalisation distribuée à grande échelle » ou « externalisation ouverte » sont d'autres traductions plus précises.*

œuvre et qu'elles conduisent à des résultats de qualité. Plus spécifiquement, nous constatons une tendance croissante à l'utilisation de jeux en lignes (Thaler et al., 2011).

Beaucoup de chercheurs (Singh et al., 2002; Lenat, 1995; von Ahn et al., 2006) prétendent que c'est irréalisable de collecter des connaissances de sens commun (Liu and Singh, 2004) à partir des textes ou de n'importe quelle autre ressource existante. Ils pensent que seuls des humains, experts ou pas, sont capables de fournir des faits du sens commun.

La contribution dans ces approches peut être faite d'une manière libre avec l'existence d'un modérateur comme dans le projet papillon (Mangeot et al., 2003), rémunérée dans le cas d'Amazon Mechanical Turk (Adda and Mariani, 2010) ou aussi par des GWAP *Game With A Purpose* (von Ahn and Dabbish, 2008) le cas de JeuxDeMots[11] et Phrase Detectives[12].

La construction collaborative d'un réseau lexical peut être catégorisée selon deux stratégies. Premièrement, comme un système contributif du type Wikipédia où des volontaires complètent les entrées (Wikitionnaire, Diko[13]...). Dans un second cas, les contributions sont faites indirectement par l'entremise de GWAP.

Malgré l'efficacité de ces approches et leur capacité à capturer de la connaissance (contribuée ou jouée) pertinente, nous avons remarqué que les ressources construites ne sont pas exemptes de *silence* et de *bruit*.

Le *silence* se manifeste par l'absence d'un grand nombre de relations malgré leur importance. Ces relations sont primordiales pour avoir une ressource de qualité suffisante pour être utilisée dans diverses applications en TALN, en particulier en analyse sémantique.

Quant au *bruit*, il est issu principalement des jeux et ceci est dû généralement aux termes ou consignes difficiles ainsi qu'à la contrainte de temps.

Le manque de relations s'explique par le fait que les contributeurs

[11] http://www.jeuxdemots.org/
[12] http://anawiki.essex.ac.uk/phrasedetectives/
[13] http://www.jeuxdemots.org/diko.php

n'indiquent que très rarement que tel type d'oiseau peut voler, car ça représente pour eux une généralité manifeste et triviale. Seuls les faits notables qui ne sont pas aisément déductibles sont indiqués par les contributeurs comme les exceptions par exemple.

Dans ce mémoire, nous avons pris comme cas d'étude le réseau lexico-sémantique JeuxDeMots [14] construit par le biais d'un outil contributif (Diko) et de multiples GWAP en ligne. Nous cherchons, dans les prochains chapitres, à détecter les silences d'information et les combler, détecter le bruit et l'information fausse et les corriger ou les supprimer ainsi que déceler la polysémie et enrichir les termes en question afin d'éviter dans le mesure du possible l'ambiguïté.

[14] http://jeuxdemots.org

Chapter 2

INFÉRENCE DE RELATIONS SÉMANTIQUES

Sommaire

2.1 Introduction

LES travaux sur l'inférence dans les réseaux lexicaux sont curieusement assez peu répandus, en particulier si on les compare aux approches concernant diverses formes de déduction logique dans les textes (pour la détermination du référent dans les groupes circonstanciels, ou la résolution d'anaphores, par exemple). Dans Blanco et al. (2011), des relations sémantiques entre concepts présents dans des textes sont inférées. Les termes du texte sont considérés de facto comme des concepts en ignorant de façon quelque peu artificielle tout problème lié à la polysémie lexicale. Dans Harabagiu (1998) du raisonnement par inférence est effectué sur le WordNet étendu où non seulement les synsets mais également les termes des définitions associées sont des concepts délexicalisés.

Quelques approches spécifiques d'inférence sont menées sur le texte à l'aide des ontologies. Comme Besnard et al. (2008) qui a capturé de l'explication causale avec de l'inférence basée sur une ontologie. Ontolearn (Velardi et al., 2006) est un systéme qui construit automatiquement des ontologies de domaine à partir du texte en utilisant entre autres des inférences. D'autres recherches ont été menées autour de l'induction des taxonomies avec WordNet (Snow et al., 2006).

Malgré les travaux considérables menés sur les inférences à partir des textes ou à partir de ressources manuellement construites, ceux sur les inférences endogènes dans les réseaux lexicaux construits par externalisation ouverte sont (quasi-)inexistants. La raison principale de cette situation, à notre avis, est le manque de telles ressources spécifiques.

Pour augmenter le nombre de relations dans le réseau lexical JDM, nous avons conçu un **système d'élicitation** [1] ayant deux principales composantes : (1) un moteur d'inférences et (2) un réconciliateur. Le moteur d'inférences propose des relations, tout comme le ferait un contributeur, qui vont être évaluées ensuite par un contributeur humain. Dans le cas de l'invalidation d'une relation inférée, le processus de réconciliation est lancé pour essayer de cerner la cause de l'erreur. La

[1] L'élicitation en Gestion des Connaissances est l'action d'aider un expert à formaliser ses connaissances pour permettre de les sauvegarder et/ou les partager. Celui ou celle qui élicite va donc inviter l'expert à rendre ses connaissances tacites en connaissances aussi explicites que possible.

consolidation dans ce cadre est la transformation d'une certaine connaissance implicite des utilisateurs en une connaissance explicite sous forme de relation(s) dans le réseau lexical.

Dans ce chapitre nous présentons les différents schémas d'inférence de relations sémantiques qui servent de patrons de comportement du système d'élicitation. Ce dernier faisant partie de notre *Système de Consolidation Endogène de RÉseaux Lexico-sémantiques* (SCEREL).

Les idées principales des inférences dans notre système sont les suivantes:

- inférer consiste à trouver de nouvelles prémisses (sous forme de relations entre les termes) à partir de prémisses connues (les relations déjà existantes);
- les inférences candidates peuvent être bloquées logiquement selon la présence ou l'absence de quelques conditions ;
- les inférences candidates peuvent être filtrées selon une évaluation de la *force* de la relation.

2.2 La déduction et l'induction

2.2.1 La déduction

Le premier type d'inférence que nous présentons est la *déduction* qui est un schéma *descendant* se basant sur la transitivité de la relation ontologique *is-a* (hyperonyme) [2]. L'hypothèse est que si un terme A a comme hyperonyme un terme B et que B est relié par une relation R à C, nous conclurons que le terme A a cette même relation avec C (Zarrouk et al., 2013c,b). Le schéma peut être formulé comme suit:

$$\exists\, A \xrightarrow{is-a} B \quad \wedge \quad \exists\, B \xrightarrow{R} C \quad \Rightarrow \quad A \xrightarrow{R} C$$

Par exemple si nous avons *Amour blanc* $\xrightarrow{is-a}$ *poisson* et *poisson* $\xrightarrow{has-part}$ *nageoire* nous allons obtenir *Amour blanc* $\xrightarrow{has-part}$ *nageoire*.

[2] Voir le tableau D.1 pour les types de relations

Principe

Considérant un terme T ayant un ensemble d'hyperonymes. Le moteur d'inférence génère à partir de chaque hyperonyme un ensemble d'inférences. Le poids d'une inférence proposée à partir de plusieurs occurrences de schémas est la moyenne géométrique du poids de chaque occurrence.

Par exemple, nous avons l'ensemble d'hyperonymes suivant pour le terme chat: {félin, être vivant, mammifère, animal de compagnie, vertébré, ...}. L'inférence *chat* $\xrightarrow{has-part}$ *squelette* peut parvenir de plusieurs hyperonymes mais plus fortement de l'hyperonyme *vertébré*, alors que la relation *chat* $\xrightarrow{lieu}$ *canapé* ne peut être conclue que de l'hyperonyme *animal de compagnie*.

Blocage logique

Le schéma proposé jusque là est certainement très naïf, notamment en considérant la ressource sur laquelle nous travaillons. En effet, B peut être polysémique et il est possible de bloquer les inférences certainement fausses se basant sur ce terme. S'il existe deux sens[3] distincts du terme B qui ont respectivement une relation avec la première et la deuxième prémisses, comme représenté dans la figure 2.1, alors très probablement l'inférence susceptible d'être proposée est fausse et par la suite rejetée.

Dans ce cas, la relation R qui peut être inférée doit satisfaire la contrainte formulée ci-dessous :

$$\begin{aligned} & A \xrightarrow{is-a} B \quad \wedge \quad B \xrightarrow{R} C \\ \bigwedge \quad & (\exists\, (B_i \xrightarrow{refinement} B) \quad \wedge \quad \exists\, (B_j \xrightarrow{refinement} B)) \\ \bigwedge \quad & (\nexists\, (A \xrightarrow{is-a} B_i) \quad \vee \quad \nexists\, (B_j \xrightarrow{R} C)) \\ & \Rightarrow \quad A \xrightarrow{R} C \end{aligned}$$

Notons que si l'une des prémisse est annotée comme non pertinente, l'inférence est directement bloquée. C'est un aspect sur lequel nous

[3]Nous utilisons indistinctement les termes *usage* et *sens*

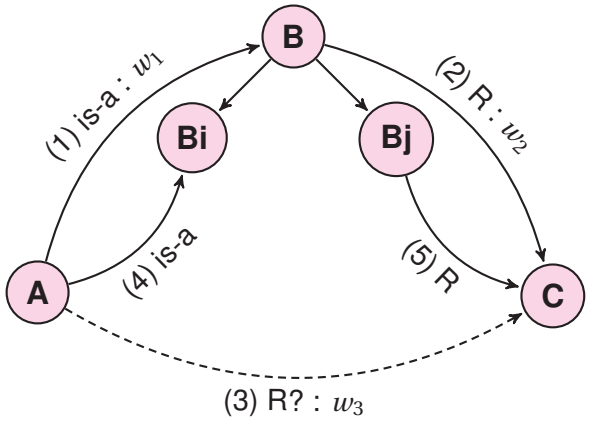

Figure 2.1 – Schéma triangulaire d'inférence représentant les cas de blocage logique causé par la polysémie du terme central B qui possède deux sens distincts B_i et B_j. Les deux flèches sans étiquette sont celles représentant les différents sens.

reviendrons dans la partie détaillant les annotations (chapitre 3).

Filtrage statistique

Il est possible d'évaluer le niveau de confiance de chaque inférence produite de manière à filtrer les inférences douteuses. Le poids w d'une relation inférée est la moyenne géométrique des poids des prémisses (relation (1) et (2) dans la figure 2.1). Si la deuxième prémisse a un poids de valeur négative, le poids de l'inférence n'est pas un nombre et cette dernière est rejetée.

Comme la moyenne géométrique est plus sensible et par conséquence moins tolérante aux petites valeurs que la moyenne arithmétique, les inférences n'étant pas basées sur des prémisses fortement valables ne passeront pas ce filtre.

$$w(A \xrightarrow{R} C) = (w(A \xrightarrow{is-a} B) * w(B \xrightarrow{R} C))^{1/2}$$
$$\Rightarrow \quad w3 = (w1 * w2)^{1/2}$$

2.2.2 L'induction

COMME pour l'inférence par déduction, l'*induction* exploite la transitivité de la relation ontologique *is-a*. Si un terme A est un B et que A est relié à C avec une relation R, alors nous pouvons supposer que B peut avoir le même type de relation avec C. Ce schéma peut être considéré comme une inférence de généralisation (Zarrouk et al., 2013a, 2014b). Nous pouvons écrire plus formellement que :

$$\exists A \xrightarrow{is-a} B \quad \wedge \quad \exists A \xrightarrow{R} C \quad \Rightarrow \quad B \xrightarrow{R} C$$

Principe

Le principe de l'induction ainsi que son filtrage statistique et logique sont similaires à ceux de la déduction. Le terme central (ici le terme A) peut être polysémique. S'il existe deux sens distincts du terme A qui ont respectivement une relation avec la première et la deuxième prémisses, comme illustré dans la figure 2.2, alors l'inférence potentielle se basant sur ce terme est très probablement fausse.

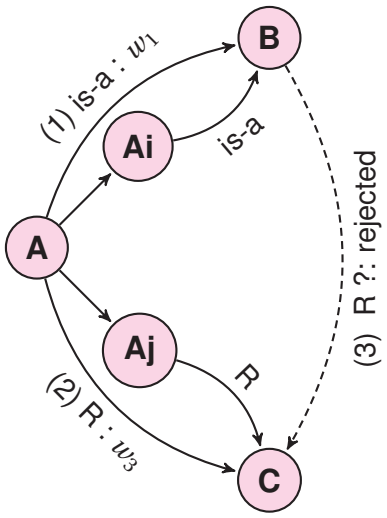

Figure 2.2 – (1) et (2) sont les prémisses, et (3) est l'induction logique proposée pour validation/invalidation. Le terme A peut être polysémique avec des sens présentant des prémisses. Dans ce cas, l'inférence susceptible d'être faite est fausse.

Blocage logique

Ce filtrage est appliqué en soumettant les inférences potentielles à la contrainte suivante :

$$\begin{aligned} & A \xrightarrow{is-a} B \quad \wedge \quad A \xrightarrow{R} C \\ \bigwedge \quad & (\exists\,(A_i \xrightarrow[refinement]{} A) \quad \wedge \quad \exists\,(A_j \xrightarrow{refinement} A)) \\ \bigwedge \quad & (\nexists\,(A_i \xrightarrow{is-a} B) \quad \vee \quad \nexists\,(A_j \xrightarrow{R} C)) \\ & \Rightarrow \quad B \xrightarrow{R} C \end{aligned}$$

Filtrage statistique

Il est possible d'évaluer un niveau de confiance au dessous duquel rejeter les relations inférées. En considérant le poids de confiance de la déduction, le poids estimé dans le cas de l'induction est :

$$\begin{aligned} w(B \xrightarrow{R} C) &= (w(A \xrightarrow{R} C))^2 \,/\, w(A \xrightarrow{is-a} B) \\ \Rightarrow \quad w2 &= \frac{(w_3)^2}{w_1} \end{aligned}$$

2.2.3 La réconciliation

Le raisonneur propose des inférences déduites et ces relations inférées sont présentées au validateur pour décider si elles sont "plutôt vraies", "plutôt fausses", "possibles" ou "vraies mais pas pertinentes".

Dans le cas d'invalidation, le réconciliateur essaie de diagnostiquer les raisons : erreur (une des relations déjà existantes sur lesquelles le moteur s'est basé pour inférer est fausse), exception (un cas rare), polysémie (l'inférence est faite en se basant sur un terme de milieu polysémique) ou un "cas général" qui va être rencontré principalement lors des premières validations. Cela est réalisé par un dialogue avec l'utilisateur dont le but ici est de découvrir les raisons de l'invalidation et d'essayer de réconcilier le réseau avec des informations issues de l'utilisateur en rapport avec une relation inférée invalidée. Ce dialogue doit être aussi minimal que possible (le moins possible de questions/le plus possible d'informations déduites). Pour savoir dans quel ordre

procéder, le réconciliateur détermine si les poids des relations (1) et (2) (figure 2.2) sont relativement forts ou faibles et ce en comparant chaque poids au seuil de confiance correspondant à son terme (Figure 2.3).

Prenons comme exemple le terme A : *écolier* ; B : *humain* ; C : *visage* ; relation 1: *écolier* $\xrightarrow{is-a}$ *humain*. La figure 2.3 est la courbe de poids/distributions pour toutes les relations sortantes ayant comme terme source A (ici *écolier*) avec un pic séparant en deux parts égales la surface sous la courbe. Le seuil de confiance est la valeur de l'intersection entre la courbe de distribution et ce pic (valant ici $\simeq 70$).

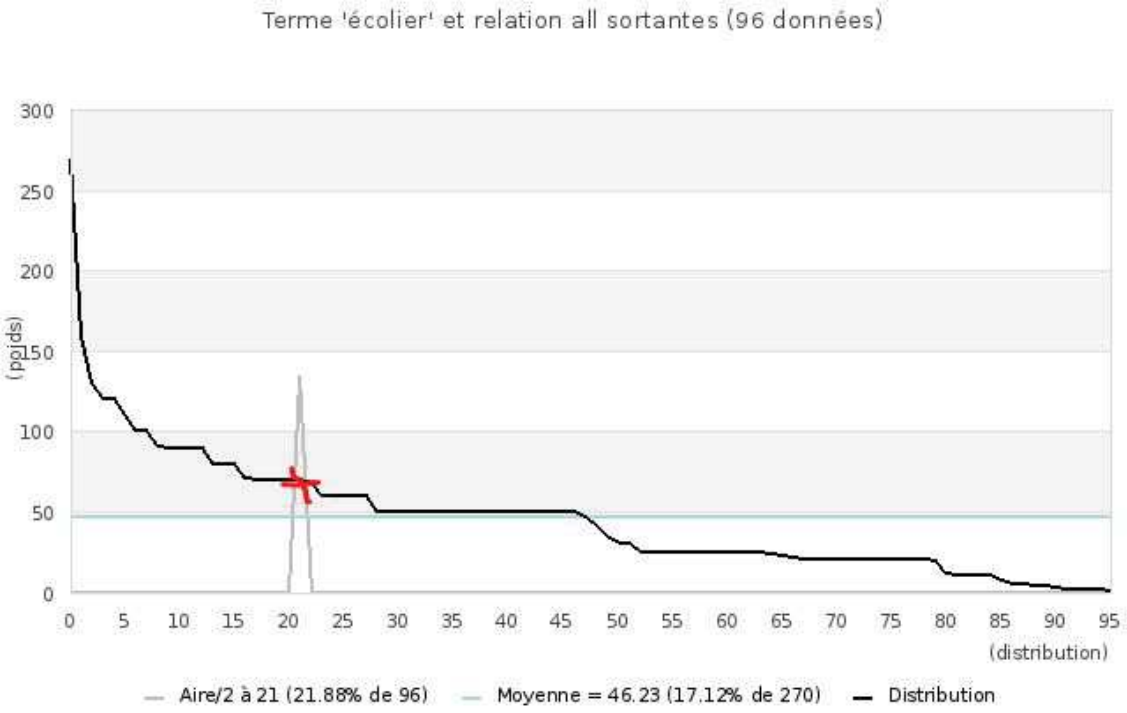

Figure 2.3 – Courbe poids/distribution des relations sortantes du terme *écolier*

- Si P($A \xrightarrow{is-a} B$) $\geqslant$ seuil-confiance(A)
  $\Rightarrow$ $A \xrightarrow{is-a} B$ est une relation vraisemblable.

- Si P($A \xrightarrow{is-a} B$) $<$ seuil-confiance(A)
  $\Rightarrow$ $A \xrightarrow{is-a} B$ est une relation douteuse.

Exception → Polysémie → Erreur Dans le cas où les deux relations de base (1) et (2) sont des relations vraisemblables, le réconciliateur essaie en premier lieu, en entamant un dialogue avec les validateurs/utilisateurs, de vérifier si la relation R inférée est une exception. Si ce n'est pas le cas, il vérifie si le terme B (terme central) est polysémique. Finalement si aucun des deux cas précédents ne s'avère vrai, il demande si c'est un cas d'erreur. Ce cas d'erreur est vérifié en dernier lors de cette démarche car les niveaux de confiance des deux relations les rendent plutôt vraisemblables.

Soit l'exemple suivant :

$$\text{A : } \textit{autruche} \text{ ; B : } \textit{oiseau} \text{ ; C : } \textit{voler} \text{ ; R : } \textit{agent-1}$$
$$\text{(1): } \textit{autruche} \xrightarrow{is-a} \textit{oiseau} \quad \text{(2): } \textit{oiseau} \xrightarrow{agent-1} \textit{voler}$$
$$\Rightarrow \quad \text{(3): } \textit{autruche} \xrightarrow{agent-1} \textit{voler}$$

Dans cet exemple, il est plutôt vrai que (1) *l'autruche est un oiseau* et que (2) *un oiseau peut voler* d'où les deux relations ont probablement des niveaux de confiance dépassant le seuil. Or la relation inférée (3) *l'autruche peut voler* est fausse et constitue une exception.

Erreur → Exception → Polysémie Dans le cas où l'une des relations (1) et (2) est douteuse, le réconciliateur suppose que c'est un cas d'erreur et que cette relation avec le faible niveau de confiance est l'origine de l'invalidation de la relation inférée. Alors il demande à l'utilisateur de la confirmer ou de l'invalider.

Dans le cas d'une invalidation pour l'une des relations, un cas d'erreur se présente. Sinon, nous procédons à la vérification des deux autres cas : cas d'exception ou de polysémie.

Soit l'exemple suivant :

$$\text{A : } \textit{enfant} \text{ ; B : } \textit{humain} \text{ ; C : } \textit{aile} \text{ ; R : } \textit{has-part}$$
$$\text{(1): } \textit{enfant} \xrightarrow{is-a} \textit{humain} \quad \text{(2): } \textit{humain} \xrightarrow{has-part} \textit{aile>plume}$$
$$\Rightarrow \quad \text{(3): } \textit{enfant} \xrightarrow{has-part} \textit{aile>plume}$$

Évidemment, la relation *enfant* $\xrightarrow{has-part}$ *aile>plume* est fausse et la relation erronée *humain* $\xrightarrow{has-part}$ *aile>plume* est la cause de cette erreur.

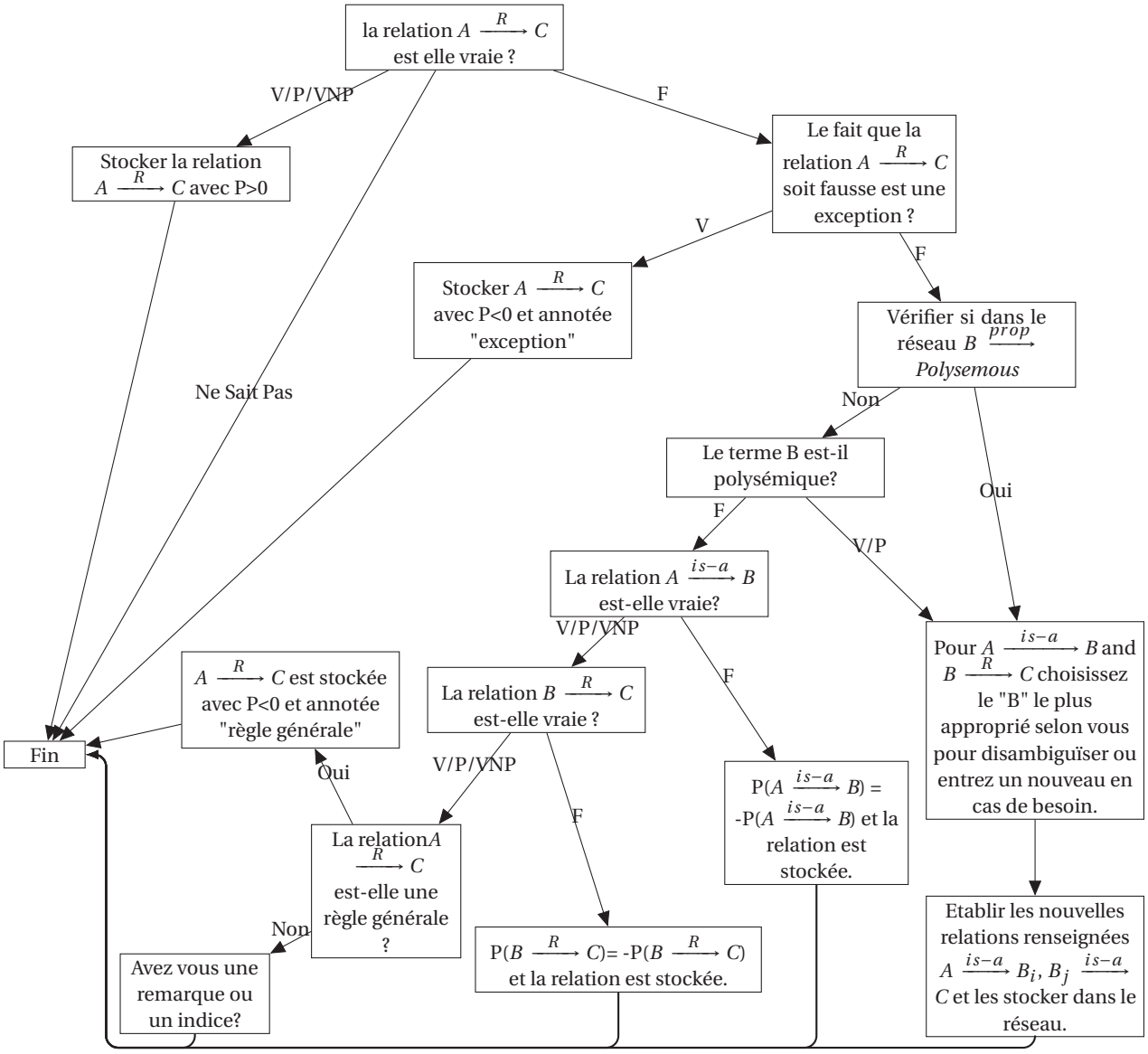

Figure 2.4 – Diagramme de flux du processus de réconciliation.
Légende : plutôt **V**rai, **P**ossible, **V**rai **N**on **P**ertinent, plutôt **F**aux.

Cas d'erreur dans les prémisses

Supposons que la relation (1) (figure 2.1 et 2.2) ait un poids faible (plus petit que le seuil de confiance). Alors le réconciliateur[4] demande au validateur si la relation (1) est vraie. Si la réponse est négative, l'opposé du poids actuel de la relation (1) lui est attribué (soit P = -1*(P)) et la réconciliation se termine. Si la réponse est positive, le réconciliateur demande si la relation (2) est vraie et il procède comme précédemment en cas de réponse négative. Sinon, enfin, il vérifie les autres cas (exception, polysémie).

Cas d'exception

Dans le cas de deux relations vraisemblables (ayant un poids $\geqslant$ seuil de confiance), le réconciliateur demande au validateur si la relation inférée $A \xrightarrow{R} C$ constituerait une exception. Si c'est le cas, la relation est stockée dans le réseau avec un poids négatif mais annotée avec une méta-information qui indique qu'il s'agit d'une exception.

Cas de polysémie

Dans le cas d'une polysémie, le terme B est soit déjà marqué dans le réseau comme polysémique ou indiqué comme tel par le validateur. Il s'agit alors de lister dans le dialogue les raffinements B_1, B_2, ...,B_n en les ordonnant selon une fonction de similarité et ainsi de permettre au validateur de choisir le plus approprié selon lui pour les deux relations $A \xrightarrow{is-a} B_i$ et $B_j \xrightarrow{R} C$. Il est possible à l'utilisateur de préciser un nouveau raffinement en cas d'insatisfaction vis-à-vis de ceux présentés.

Après cette procédure, le réseau sera réconcilié par deux nouvelles relations $A \xrightarrow{is-a} B_i$ et $B_j \xrightarrow{R} C$ qui pourront être utilisées ultérieurement par le moteur d'inférences.

[4]Le réconciliateur est la machine alors que le validateur est l'utilisateur humain.

2.2.4 Expérimentations

NOUS avons lancé le moteur d'inférence avec la déduction et l'induction dans le but de mesurer la production de ce moteur avec le blocage et le filtrage. Nous avons sélectionné, de l'ensemble des relations inférées par déduction et induction, un sous-ensemble aléatoire de **400** propositions pour chaque type de relation sur lesquelles nous avons appliqué le processus de validation/réconciliation. A noter que ces expérimentations sont menées dans le seul but de l'évaluation et qu'actuellement tout notre système de consolidation fonctionne itérativement en conjonction avec les contributions et les jeux.

Enclencher la déduction et l'induction

Nous avons appliqué le moteur d'inférence avec les schémas de déduction et d'induction sur approximativement **23 000** termes sélectionnés aléatoirement et ayant au moins un hyperonyme ou hyponyme (a contrario, les schémas ne peuvent pas s'appliquer). Ce qui a produit par déduction **1 484 209** inférences (**77 089** autres ont été bloquées). Le seuil de confiance étant fixé à un poids de **25**. Cette valeur de seuil est pertinente dans le sens où quand un contributeur humain propose une relation validée ensuite par un expert, elle est introduite avec un poids par défaut de **25**. Pour l'induction, le moteur d'inférence a produit **353 371** propositions. Le tableau 2.1 présente le nombre de relations proposées par notre système avec la déduction. Les différents types de relation de la deuxième prémisse (la relation générique *R* dans les schémas) sont différemment productifs. Cela est dû principalement au nombre des relations existantes et la distribution de leur type dans le réseau. La productivité d'un type de relation est le ratio entre le nombre proposé d'inférences et le nombre d'occurrences de ce type dans le réseau.

La relation transitive *is-a* est la moins productive ce qui peut paraître surprenant à première vue (tableau 2.2). En fait, ce type de relation est déjà très bien renseigné dans le réseau, ce qui fait que seulement un petit nombre d'inférences est produit. Les chiffres sont inversés pour d'autres type de relations qui sont pas bien renseignées. La relation *agent-1* est de loin la plus productive avec 30 fois plus de propositions que ce qui existait dans le réseau.

Type de relation	**proposées**	**%**	**bloquées**	**%**	**filtrées**	**%**
is-a	91 037	6,13	4 034	5,23	53 586	26,32
has-parts	372 688	25,11	31 421	40,76	100 297	49,26
holo	108 191	7,28	17 944	23,27	26 818	13,17
location	271 717	18,30	11 502	14,92	14 174	6,96
charac	203 095	13,68	2 647	3,43	6 576	3,23
agent-1	198 359	13,36	9 052	11,74	1 122	0,55
instr-1	24 957	1,68	127	0,16	391	0,19
patient-1	14 658	0,98	7	0,01	13	
location-1	145 159	9,78	129	0,17	206	0,10
location-action	50 035	3,379	91	0,12	132	0,06
object-mater	4 313	0,29	135	0,17	262	0,12
Total	**1 484 209**	100	**77 089**	100	**203 577**	100

Tableau 2.1 – Nombres et pourcentages des inférences proposées, bloquées et filtrées par déduction.

Type de relation	**proposées**	**existantes**	**Productivité**
is-a	91 037	91 799	99,16%
has-part	372 688	21 886	1702,86%
holo	108 191	13 124	824,37%
location	271 717	26 346	1031,34%
charac	203 095	24 180	839,92%
agent-1	198 359	6 820	2908,48%
instr-1	24 957	4 797	520,26%
patient-1	14 658	3 930	372,97%
location-1	145 159	8 835	1642,99%
location-action	50 035	4 559	1097,49%
object-mater	4 313	3 097	139,26%

Tableau 2.2 – Productivité des types de relations utilisés par le moteur d'inférences.

Dans la figure 2.5, la distribution semble suivre une loi de puissance. Intuitivement, on peut s'attendre à un tel résultat étant donné que la distribution des relations dans le réseau est elle-même régie par une telle loi.

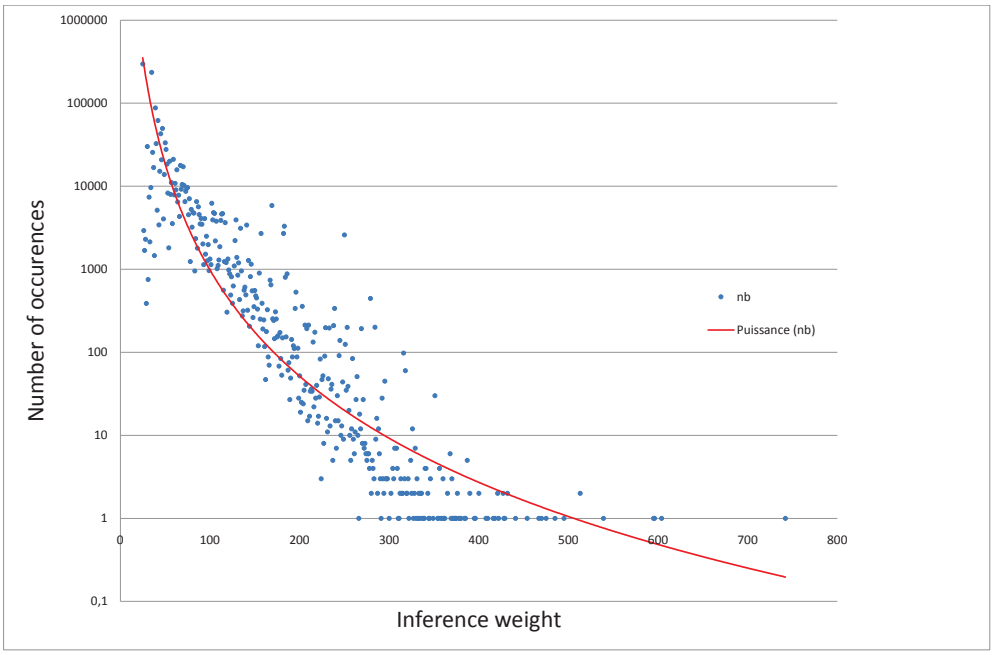

Figure 2.5 – Distribution des inférences proposées selon leur poids. La distribution semble une loi de puissance contravariante au poids. En pratique, les relations candidates de très fort poids sont validées les premières.

Évaluation de la réconciliation

La table 2.3 présente une évaluation de l'état des inférences proposées par le moteur d'inférences déductives. Les inférences sont valides globalement à 80-90% avec environ 10% de relations valides mais pas pertinentes (par exemple $mammouth \xrightarrow{has-part} atome$).

Nous remarquons que le taux d'erreur dans les prémisses est assez bas en notant que les erreurs peuvent être corrigées à fur et à mesure. La réconciliation aide à identifier les 5% représentant les quelques cas de polysémie. Globalement les faux négatifs (inférence votées fausses alors qu'elles sont vraies) et les faux positifs (inférences votées vraies alors qu'elles sont fausses) sont évalués à moins de 0.5%. Pour l'induction

(tableau 2.4), la relation *is-a* est écartée car un réseau lexical n'est pas réductible à une ontologie et l'héritage multiple est possible. Les résultats semblent être de 5% meilleurs que ceux de la déduction, les inférences sont valides à un taux de 80-95%. La différence majeure avec les résultats de la déduction est sur les erreurs causées par la polysémie qui sont plus basses.

Déduction	**% valides**		**% erreur**		
Type de relation	**pert**	**¬ pert**	**prém**	**excep**	**pol**
is-a	76%	13%	2%	0%	9%
has-part	65%	8%	4%	13%	10%
holo	57%	16%	2%	20%	5%
location	78%	12%	1%	4%	5%
charac	82%	4%	2%	8%	4%
agent-1	81%	11%	1%	4%	3%
instr-1	62%	21%	1%	10%	6%
patient-1	47%	32%	3%	7%	11%
location-1	72%	12%	2%	10%	6%
location-action	67%	25%	1%	4%	3%
object-mater	60%	3%	7%	18%	12%

Tableau 2.3 – Résultats de la validation/réconciliation selon les types de relations pour la déduction. Les relations valides peuvent être **pert**inentes ou non, et les erreurs peuvent être à cause des **prém**isses erronées, des **excep**tions ou de la **pol**ysémie.

Induction	**% valides**		**% erreur**		
Type de relation	**pert**	**¬ pert**	**prém**	**excep**	**pol**
has-part	78%	10%	3%	2%	7%
holo	68%	17%	2%	8%	5%
location	81%	13%	1%	2%	3%
charac	87%	6%	2%	2%	3%
agent-1	84%	12%	1%	2%	1%
instr-1	68%	24%	1%	4%	3%
patient-1	57%	36%	3%	2%	2%
location-1	75%	16%	2%	5%	2%
location-action	67%	28%	1%	3%	1%
object-mater	75%	10%	7%	5%	3%

Tableau 2.4 – Résultats de la validation/réconciliation selon les types de relations pour l'induction. La relation *is-a* est inappropriée pour l'induction.

2.3 L'abduction

Le schéma d'inférence par abduction suit une stratégie basée sur des exemples. En somme, l'abduction se base sur la *similarité* entre les termes. Dans notre contexte, cela peut être formalisée comme *le partage de certaines relations sortantes entre les termes.* Le principe de l'inférence par abduction est de supposer que les relations détenues par un terme peuvent être proposées comme éventuelles relations aux termes similaires (Zarrouk et al., 2014a).

En fait, l'abduction sélectionne d'abord un ensemble de termes similaires au terme cible *A* (le terme à enrichir), ces termes sont considérés comme exemples corrects. Les relations sortantes des termes exemples et qui ne sont pas déjà communes avec *A* sont proposées comme relations potentielles à ce dernier et ensuite présentées aux utilisateurs pour le processus validation/réconciliation.

À la différence de la déduction et de l'induction, l'abduction peut être appliquée sur des termes n'ayant pas de relations ontologiques (ou ayant des relations non pertinentes bien que vraies) et peut générer des relations de ce type pouvant être utilisées ensuite par le reste de

la boucle de schémas d'inférences.

Ce schéma d'inférence est basé sur l'abduction et peut être vu comme une stratégie d'exemples. Si un terme donné A est similaire à un certain ensemble de termes, alors potentiellement les propriétés de ces termes peuvent être valides pour A. Plus précisément, si A possède quelques relations en commun avec les autres termes, il peut potentiellement être un candidat pour détenir les autres relations sortantes de ces exemples.

Une relation sortante d'un terme est définie par ce même terme en tant que nœud source *nc*, un type de relation *t*, un poids *w* et un nœud cible (nœud d'arrivée) $nc : R_i = \langle\ ns_i,\ t_i\ ,\ w_i\ ,\ nc_i\ \rangle$.

2.3.1 Principe de fonctionnement

COMME précédemment indiqué, une relations sortante est un quadruplet de nœud source *ns*, type *t*, poids *w* et de nœud cible *nc*.

Par exemple, considérons le terme A ayant n relations sortantes. Parmi ces relations nous avons :

$$\bullet\ bec \overset{has-part}{\longleftarrow} A \quad \& \quad \bullet\ nid \overset{location}{\longleftarrow} A.$$

Nous avons trouvé 3 exemples partageant ces deux relations avec le terme A :

$$\bullet\ bec \overset{has-part}{\longleftarrow} \{ex_1, ex_2, ex_3\}$$

$$\bullet\ nid \overset{location}{\longleftarrow} \{ex_1, ex_2, ex_3\}$$

Nous considérons ces termes comme un ensemble d'exemples valables similaires à A. Ces exemples détiennent d'autres relations sortantes que le terme A n'a pas, ces relations sont proposées comme relations potentielles pour A. Par exemple :

$$\bullet\ \{ex_1, ex_2\} \overset{agent-1}{\longrightarrow} voler \qquad \bullet\ \{ex_2\} \overset{charac}{\longrightarrow} coloré$$

- $\{ex_1, ex_2, ex_3\} \overset{has-part}{\longrightarrow} plumes$
- $\{ex_3\} \overset{agent-1}{\longrightarrow} chanter$

Nous inférons que *A* est un potentiel candidat pour détenir ces relations et nous les proposons pour validation.

- $A \overset{agent-1}{\longrightarrow} voler$?
- $A \overset{has-part}{\longrightarrow} plumes$?
- $A \overset{charac}{\longrightarrow} coloré$?
- $A \overset{agent-1}{\longrightarrow} chanter$?

2.3.2 Filtrage et paramétrage

L'APPLICATION sommaire du schéma de l'abduction sur les termes génère énormément de bruit sous forme de beaucoup d'inférences erronées. En conséquence nous avons mis en place une stratégie de filtrage permettant à l'utilisateur de paramétrer le système et éviter d'avoir une quantité considérable de relations candidates douteuses.

Dans ce but, nous définissons deux paires de seuils différentes. Le premier doublet (δ_1, ω_1) est utilisé pour sélectionner l'ensemble valide d'exemples x_1,x_2...x_n qui est défini comme suit :

$$\delta_1 = \max(3, \text{nbogr}(A) \times 0.1) \tag{2.1}$$

où *nbogr(A)* est le nombre de relations sortantes du terme *A*.

$$\omega_1 = \max(25, \text{mwogr}(A) \times 0.5) \tag{2.2}$$

où *mwogr(A)* est la moyenne des poids des relations sortantes de *A*.
Le deuxième doublet (δ_2, ω_2) est utilisé afin de sélectionner les bonnes relations candidates à proposer au terme *A* parmi l'ensemble de relations sortantes des exemples R'_1,R'_2...R'_q.

$$\delta_2 = \max(3, \overline{\{x_i\}} \times 0.1) \tag{2.3}$$

où $\overline{\{x_i\}}$ est le cardinal de l'ensemble $\{x_i\}$.

$$\omega_2 = \max(25, \text{mwogr}(\{x_i\}) \times 0.5) \tag{2.4}$$

où $mwogr(\{x_i\})$ est la moyenne des poids de relations sortantes de l'ensemble d'exemples x_i.

Si un terme A partage, avec un ensemble d'exemples $x_1, x_2, \ldots x_n$, au minimum δ_1 relations sortantes, ayant un poids supérieur à ω_1, parmi le totale des relations $R_1, R_2, \ldots R_p$, vers les termes $T_1, T_2, \ldots T_p$, on admet que ce terme a un degré de similarité valable avec ces exemples. Après la construction d'un ensemble d'exemples sur lequel nous pouvons appliquer notre moteur d'inférence par abduction, nous procédons avec la deuxième partie de notre stratégie.

Si nous trouvons au moins δ_2 exemples x_i détenant une relation donnée R'_k ayant un poids supérieur à ω_2 vers un terme B_k (plus formellement $R'_k = \langle\ t,\ w \geq \omega_2,\ B_k\ \rangle$), nous supposons que le terme A peut avoir cette même relation R'_k avec le même terme cible B_k (figure 2.6).

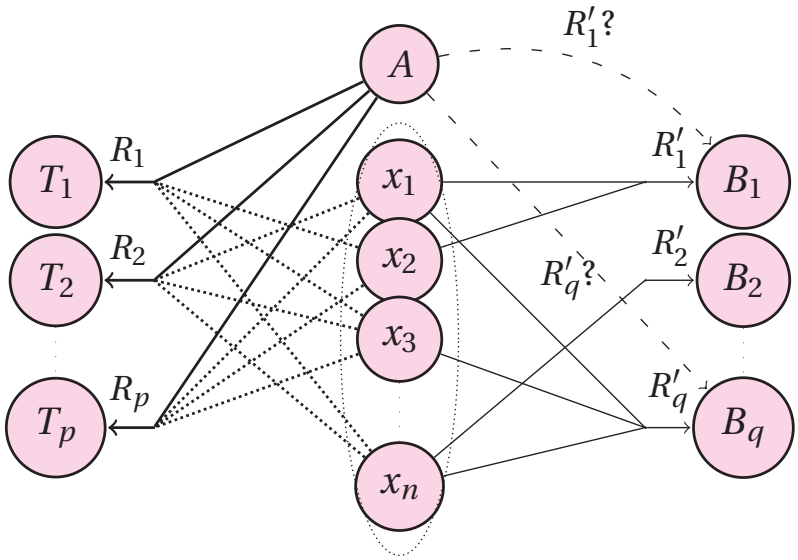

Figure 2.6 – Exemple d'un schéma d'abduction avec les exemples x_i ayant des relations en commun avec le terme A. De nouvelles relations sont proposées au terme A.

Dans la figure 2.6 nous avons simplifié les seuils à 2 dans un but il-

lustratif. Alors les exemples ne sont sélectionnés que s'ils ont au moins 2 relations communes avec A. Une relation $R'_{1\rightarrow q}$ doit être détenue par un minimum de 2 exemples pour qu'elle soit proposée comme relation potentielle pour A. Plus formellement :

$$\blacktriangleright\ x_1 \xrightarrow{R'_1} B_1\ \&\ x_2 \xrightarrow{R'_1} B_1 \Rightarrow R'_1 : 2$$
$$\Longrightarrow \text{propose } A \xrightarrow{R'_1?} B_1$$
$$\blacktriangleright\ x_n \xrightarrow{R'_2} B_2 \qquad\qquad \Rightarrow R'_2 : 1$$
$$\Longrightarrow \text{relation rejetée.}$$
$$\blacktriangleright\ x_1 \xrightarrow{R'_q} B_q\ \&\ x_3 \xrightarrow{R'_q} B_q\ \&\ x_n \xrightarrow{R'_q} B_q \Rightarrow R'_q : 3$$
$$\Longrightarrow \text{propose } A \xrightarrow{R'_q?} B_q$$

Pour un filtrage statistique nous pouvons agir sur le doublet de seuils (δ_2, ω_2) comme le nombre minimum d'exemples x_i ayant une relation R' avec le terme B_k. Il est aussi possible d'évaluer le poids d'une relation inférée par ce schéma comme suit :

$$w(A \xrightarrow{R'} B) = \left(\frac{1}{nb_{R'_{ins}}}\right) \sum_{i=1,j=1}^{n',p} \sqrt[3]{w_1 \times w_2 \times w_3} \tag{2.5}$$

Où :
$nb_{R'_{ins}} = n' \times p$: le nombre d'instances de la relation R' candidate proposée (pour chaque exemple participant à l'inférence de R' et pour chaque relation partagée avec A)
n' : nombre d'exemples x participant à l'inférence
p : nombre de relations partagées entre A et son ensemble d'exemples
$w_1 = w(A \xrightarrow{R_j} T_j)$
$w_2 = w(x_i \xrightarrow{R_j} T_j)$
$w_3 = w(x_i \xrightarrow{R'} B)$

Ces paramètres de filtrage sont ajustables selon les besoins et les critères de l'utilisateur afin de satisfaire les diverses attentes. Les constantes dans les formules des seuils ont été définies empiriquement.

2.3.3 Quelle réconciliation?

LA réconciliation pour l'abduction est plus simple que celle pour la déduction ou l'induction. L'effet potentiel de la polysémie est contrebalancé par l'approche statistique implémentée par le nombre d'exemples (quand valides). La réconciliation, dans ce cas d'abduction, est de déterminer si les fausses inférences ont été produites logiquement ou pas en considérant l'ensemble des exemples.

Dans 97% des cas, les inférences erronées étant qualifiées comme *fausses mais logiques* par les utilisateurs.

Par exemple : *Boeing 747* $\xrightarrow{has-part}$ *hélice*, *baleine* $\xrightarrow{location}$ *lac* et *pélican* $\xrightarrow{agent-1}$ *chanter* sont des fausses inférences qui *auraient pu être correctes*. En examinant les exemples utilisés pour produire ces inférences, il n'y a aucune manière possible pour le système de prévoir que ces relations sont fausses.

2.3.4 Expérimentations

NOUS avons appliqué le moteur d'abduction sur une sous-partie du RLS JDM ($\simeq$ 1‰) ce qui a produit **629 987** inférences parmi lesquelles **137 416** nouvelles (n'existent pas déjà dans le réseau). Ces **137 416** relations concernent **10 889** entrées lexicales distinctes, ce qui fait une moyenne de **12** nouvelles relations par entrée. La distribution de ces relations proposées suit une loi de puissance, ce qui est pas surprenant en soi car la distribution des relations dans le réseau lexical suit cette même loi. Ces chiffres indiquent que l'abduction reste productive malgré son indépendance des relations ontologiques existantes.

Le tableau 2.5 présente le nombre de relations proposées par le moteur d'inférences par abduction. Les différents types de relations sont diversement productifs et ce principalement à cause du nombre des relations existantes et de la distribution de leurs types. Le type de relation le plus productif étant le *has-part* (un terme *A* a comme partie un terme *B*) et le moins productif est le *holo* (holonyme). Les relations valides représentent **80%** des relations évaluées. Ce qui est surprenant c'est que ce taux d'exactitude (**80%**) paraît globalement constant en dépit du type de la relation. Le type de relation ayant le plus petit taux d'exactitude

de **77%** est *instr-1* (ce qu'on peut faire avec *X* en tant qu'instrument), celui ayant le plus haut taux d'exactitude (**85%**) est *action-location* (la relation associant à une action une location typique où elle peut se produire).

La stabilité apparente de l'exactitude des inférences peut être une conséquence positive du fait de se baser sur un ensemble d'exemples irréductible à **20%** de fausses inférences.

Abduction	**#prop**	**#eval**	(%)	**Correcte**	(%)	**fausse**	(%)
is-a	7 141	421	5.9	343	81.5	78	18.5
has-part	26 517	720	2.7	578	80.3	142	19.7
holo	1 592	153	9.6	124	81	29	18.9
agent	7 739	298	3.9	236	79.2	62	20.8
location	17 148	304	1.8	253	83.2	51	16.8
instr	10 790	431	4	356	82.6	75	17.4
charac	7 443	319	4.3	251	78.7	68	21.3
agent-1	18 147	955	5.3	780	81.7	175	18.3
instr-1	11 867	886	7.5	682	77	204	23
location-1	14 787	1 106	7.5	896	81	210	19
location-action	8 268	270	3.3	214	79.3	56	20.7
action-location	5 976	170	2.8	145	85.3	25	14.7
Total	137 416	6 033	4.3	4 858	81	1 175	19

Tableau 2.5 – Nombres de propositions produites par abduction et taux de relations valides et fausses.

La figure 2.7 présente deux types de données : (1) le pourcentage d'inférences valides selon le nombre d'exemples utilisés pour les proposer, et (2) la proportion entre les relations proposées et le total de **107 416** de relations selon le même critère. Ce que nous remarquons clairement est que quand le nombre d'exemples augmente, le taux d'inférences valides augmente de même mais le nombre total de relations proposées chute. Le nombre d'inférences suit une loi de puissance inversée de celui des exemples requis.

Avec 3 exemples, seulement 40% des propositions sont correctes, et avec un minimum de 6 exemples plus de 3/4 des propositions sont

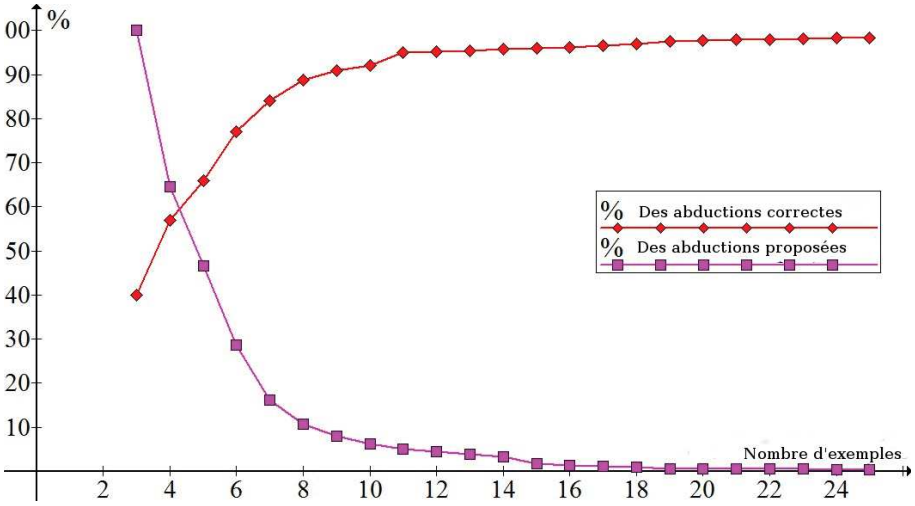

Figure 2.7 – Production d'inférence par abduction et le pourcentage d'exactitude par rapport au nombre d'exemples.

votées exactes. Le F-score optimal à l'intersection des deux courbes est aux alentours de 4 exemples.

La figure 2.8 montre la moyenne du nombre de nouvelles inférences pendant une itération du moteur d'inférence par abduction. Entre deux itérations, les utilisateurs/validateurs évaluent les relations proposées. Ce processus est appliqué à discrétion avec la possibilité de laisser des relations non évaluées. Les expérimentations montrent que les utilisateurs ont tendance à valider les relations clairement exactes et à invalider celles clairement fausses. Les propositions ayant un statut flou sont souvent mises à part par rapport à celles jugées plus faciles.

La troisième itération est la plus productive avec une moyenne d'approximativement 20 nouvelles relations. Après cela, le moteur d'inférence a tendance à être moins productif en proposant moins de nouvelles relations. Notant que le processus d'abduction peut, d'une itération à une autre, supprimer certaines propositions précédemment faites.

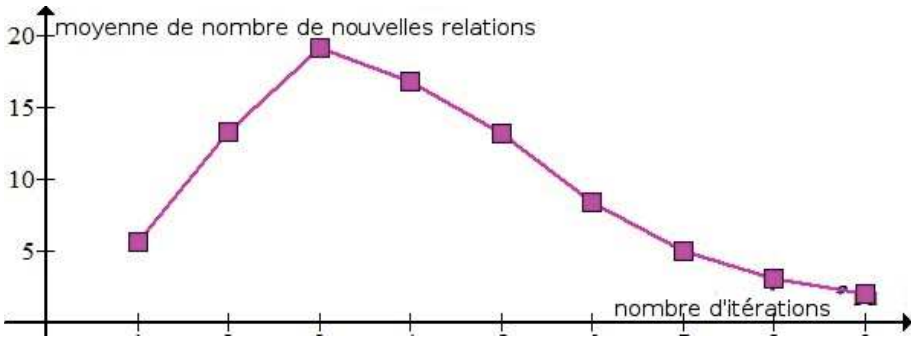

Figure 2.8 – Moyenne des nombres de nouvelles relations proposées tout au long des itérations.

2.4 L'inférence par raffinement

Qu'est-ce qu'un raffinement?

UN terme marqué comme polysémique par les utilisateurs possède plusieurs *usages de mot* qui peuvent se différencier fondamentalement des sens de mots classiquement définis. Un usage donné peut avoir lui même plusieurs *raffinements* et l'ensemble d'usage va prendre la forme d'un arbre.

Par exemple, sur la figure 2.9, un *journal* peut être avoir comme raffinements (publication, émission, etc.). *Journal>publication* peut être raffiné en (*journal>publication>entreprise de presse*, *journal>publication>exemplaire*, *journal>publication>contenu*).

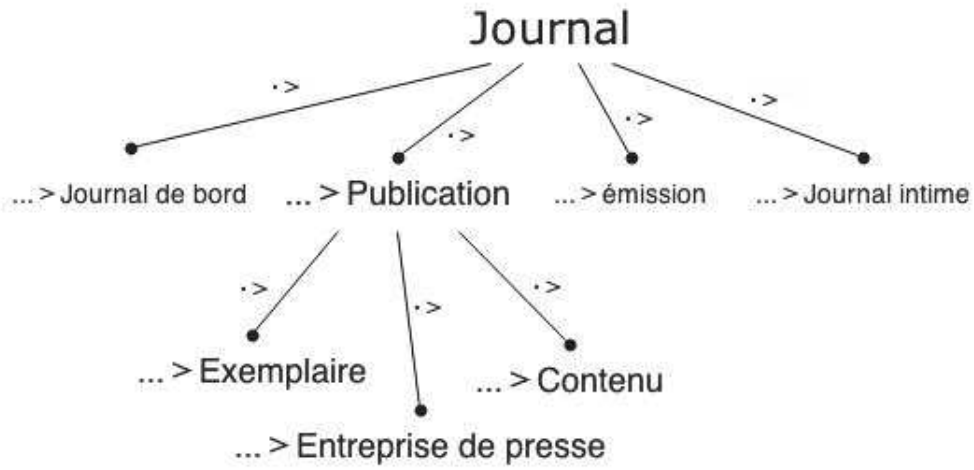

Figure 2.9 – Arbre de raffinements (notés ⩾) du terme *journal*. Le premier niveau se compose de *Journal>journal de bord*, *Journal>publication*, *Journal>émission*, etc. *Journal>publication* se raffine en un deuxième niveau avec *journal>publication>entreprise de presse*, *journal>publication>exemplaire* ou encore *journal>publication>contenu*. Cet arbre fait partie du réseau lexical et utilise une relation spécifique de raffinement. Chaque raffinement est un nœud relié aux autres raffinements par la relation *refinement* ainsi qu'au reste du réseau.

2.4.1 Principe de fonctionnement

Ce schéma, comme son nom l'indique, exige que le terme *A* sur lequel il va s'appliquer ait un raffinement *A'*. Ce schéma d'inférence de relations par raffinement (SIR_R) va essayer d'échanger les relations sortantes pas communes entre le raffinement A' et chaque synonyme, hyperonyme ou hyponyme du terme A (figure 2.10). Les relations échangées représentent les inférences à valider/invalider (Zarrouk and Lafourcade, 2014a,b).

Pour augmenter la pertinence des relations proposées, nous nous assurons de l'existence de relation(s) entre le terme de raffinement *A'* et le terme *B*.

Par exemple, supposant que nous avons un terme *A* : *rose*, qui possède au minimum deux raffinements *A'* : *rose>fleur* et *rose>prénom* et

un hyperonyme *B* : *plante*. Dans cet exemple, le terme *A'* : *rose>fleur* et *B* : *plante* sont reliés par une relation quelconque contrairement au terme *A'* : *rose>prénom* et *B*. Cette stratégie évite au schéma de proposer par exemple *plante* $\xrightarrow{has-part}$ *lettres* (relation provenant de *rose>prénom*) (figure 2.10).

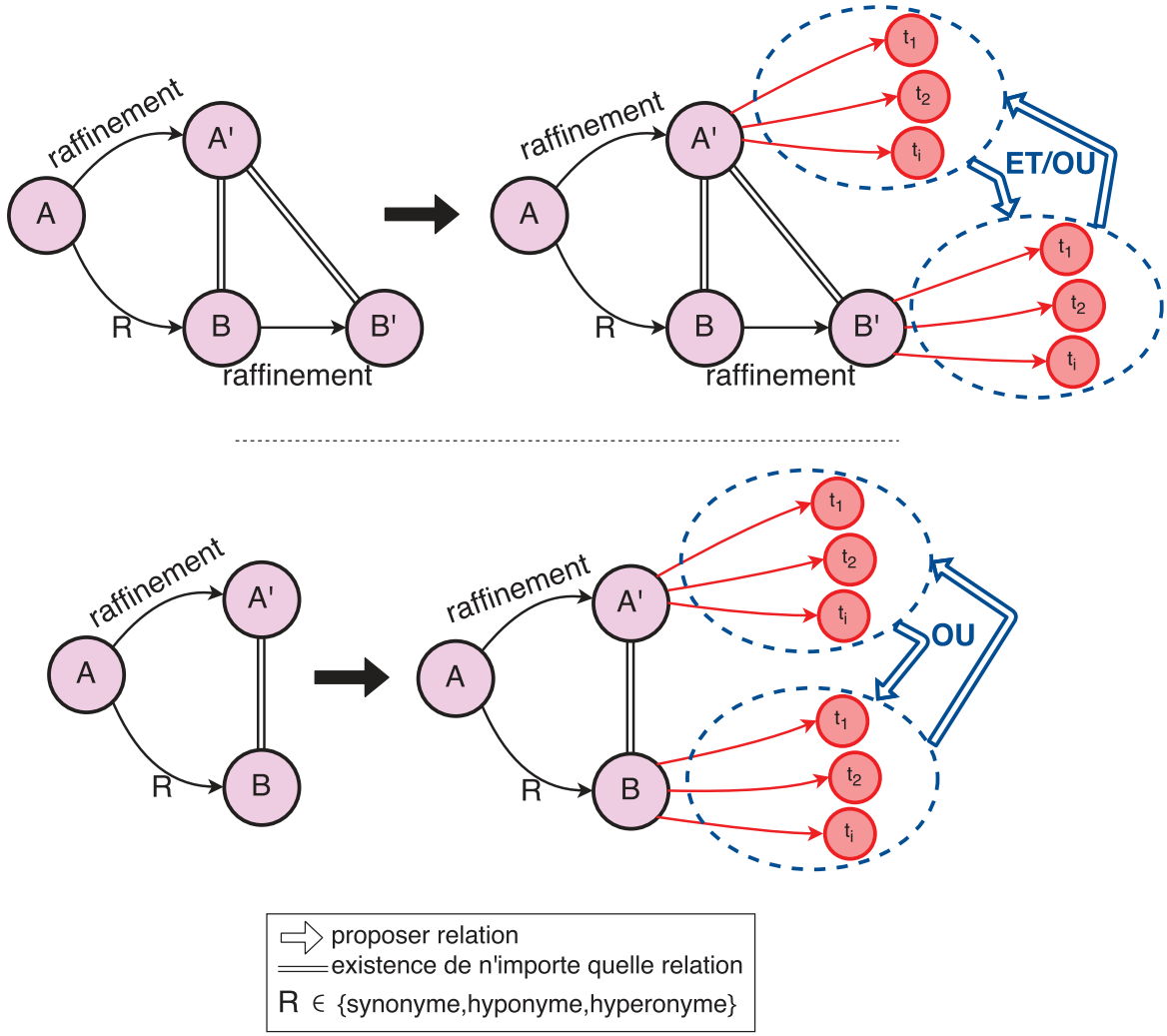

Figure 2.10 – Schéma explicatif du fonctionnement du (SIR_R) avec les deux cas possibles (existence ou pas d'un raffinement du terme B). La direction de la proposition des relations potentielles (entre A' et B' ou A' et B) dépend de la nature de la relation R.

Une stratégie complémentaire consiste à ne pas proposer de relations d'un hyperonyme à ces hyponymes mais plutôt le contraire. La direction du transfert de relations sera toujours du plus spécifique (hyponyme) au plus générique (hyperonyme) (sauf pour les synonymes où l'échange se fait dans les deux sens) et ce parce qu'en général les relations sortantes d'un hyperonyme ne sont pas forcément valides pour tous ses hyponymes.

Comme par exemple le terme *A* : animal ayant comme raffinement un *A'* : *animal>metazoa* qui a comme relations sortantes *animal>metazoa* $\xrightarrow{has-part}$ *nageoire, écailles, croc*. Ces relations ne sont pas valides pour l'hyponyme *vache* de *animal* par exemple.

Ce schéma a un comportement subtilement différent selon la nature du terme *B* (synonyme, hyponyme ou hyperonyme) qui a été illustré dans le pseudo-algorithme 1.

foreach $(A$ ***such as*** $\exists A \xrightarrow{raff} A' \wedge \exists A \xrightarrow{R \in (syn/hyper/hypo)} B \wedge \exists B \overset{X}{\rightleftharpoons} A')$ **do**
  if $(\exists B \xrightarrow{raff} B' \wedge \exists A' \overset{X}{\rightleftharpoons} B')$ **then**
    if $A \xrightarrow{R=syn} B$ **then**
      $out(B') \overset{\mathbb{P}}{\hookrightarrow} out(A')$;
      $out(A') \overset{\mathbb{P}}{\hookrightarrow} out(B')$;
    end
    if $A \xrightarrow{R=hyper} B$ **then**
      $out(A') \overset{\mathbb{P}}{\hookrightarrow} out(B')$;
    end
    if $A \xrightarrow{R=hypo} B$ **then**
      $out(B') \overset{\mathbb{P}}{\hookrightarrow} out(A')$;
    end
  end
  if $(\nexists B \xrightarrow{raff} B' \vee \nexists A' \overset{X}{\rightleftharpoons} B')$ **then**
    if $A \xrightarrow{R=syn} B$ **then**
      $out(A') \overset{\mathbb{P}}{\hookrightarrow} out(B)$;
    end
    if $A \xrightarrow{R=hyper} B$ **then**
      $out(A') \overset{\mathbb{P}}{\hookrightarrow} out(B)$;
    end
    if $A \xrightarrow{R=hypo} B$ **then**
      $out(B) \overset{\mathbb{P}}{\hookrightarrow} out(A')$;
    end
  end
end

Pseudo-algorithme 1 : Le comportement du SIR_R l'existence du terme B' et du type de la relation R.
• ***out****(X)* fait référence aux relations sortantes du terme X;
• ***out****(X)* $\overset{\mathbb{P}}{\hookrightarrow}$ ***out****(Y)*: propose toutes les relations sortantes de X comme relation sortantes de Y;
• *raff* dans ce pseudo-algorithme représente la relation appelée *refinement* dans le reste du mémoire;
• $A \overset{X}{\rightleftharpoons} B$: une relation entre A et B dans n'importe quelle direction.

2.4.2 Expérimentation

NOUS avons appliqué séparément en considérant le type de la relation sur laquelle se fonder (pseudo-algorithme 1) :

- SIR_R (synonyme) : le schéma appliqué avec $R = syn$ (spécification : le terme *A'* et *B* (*B'* s'il existe) s'échangent[5] les relations non communes entre eux);
- SIR_R (hyponyme) : le schéma s'applique avec $R = hypo$ (spécification : les relations sont copiées de *B* (ou *B'*) à *A*);
- SIR_R (hyperonyme) : le schéma s'applique avec $R = hyper$ (spécification : les relations sont copiées de *A'* à *B* (ou *B'* s'il existe).

Productivité

Le schéma se base sur une condition pour être appliqué comme expliqué précédemment. Un sous-ensemble du réseau contenant seulement **6 349** termes satisfait cette condition. L'ensemble du processus a produit **308 532** nouvelles relations ce qui fait une moyenne de **49** nouvelles relations par entrée.

Le SIR_R (syn) a produit **2,7** fois les relations déjà existantes ce qui fait de lui la variante la plus productive du schéma, suivi par le SIR_R (hypo) produisant **2,6** fois et SIR_R (hyper) avec une productivité de **0,73** fois (tableau 2.6).

	# existantes	# proposées	productivité
SIR_R (syn)	38 792	105 288	271,41%
SIR_R (hyper)	139 490	101 908	73,05%
SIR_R (hypo)	38 756	101 336	261,47%

Tableau 2.6 – Productivité du SIR_R.

La distribution des relations proposées selon le type de relation est illustrée par le tableau 2.7. Les différents types de relations sont dif-

[5] S'échanger dans le sens copier les relations et non pas les transférer

féremment productifs et ceci est principalement dû au nombre de relations déjà existantes et la distribution de leur type. L'*assoc (idée associée)* est le type le plus proposé des trois variantes du schéma et ceci s'explique par le spectre sémantique très large de cette relation puisqu'elle fait référence à tous les termes associés au terme cible. Les chiffres sont inversés pour d'autres types de relations qui ne sont pas pas assez renseignés dans le réseau mais qui restent néanmoins valides.

Type de relation	RIS_R(syn)	RIS_R(hypo)	RIS_R(hyper)
assoc	50 946	39 325	51 960
has-part	13 362	13 120	8 049
is-a	3 711	5 114	5 707
hypo	6 463	10 186	6 326
holo	1 927	1 407	3 757
charac	10 378	10 063	7 614
location	5 921	9 251	5 529
agent-1	6 887	9 366	3 024
autres	5 693	4 076	9 370

Tableau 2.7 – Relations proposées selon la variante du schéma et du type de relation.

Précision

La procédure de validation été appliquée manuellement sur un ensemble de **1 000** propositions choisies aléatoirement pour chaque variante de schéma. Le SIR_R (syn) a la précision la plus élevée avec **90,76%** de propositions valides, le SIR_R (hyper) avec **72,69%** et **66,24%** pour le SIR_R (hypo) (tableau 2.8).

Type de relation	RIS_R(syn)	RIS_R(hypo)	RIS_R(hyper)
assoc	92.4%	65%	60.8%
has-part	93.2%	46.9%	80.8%
is-a	86.2%	56.4%	46%
hypo	69.7%	60%	65%
holon	74.4%	60.7%	64.2%
charac	91.5%	73.9%	90.5%
location	91.1%	81%	79.5%
agent-1	92.1%	78.9%	90.9%

Tableau 2.8 – Pourcentage de relations valides par variante de schéma et par type de relation. Le SIR_R (syn) est la variante de schéma ayant systématiquement la meilleure précision.

La variante de schéma se basant sur les synonymes a systématiquement la meilleure précision pour tous les types de relations. Les autres variantes ont des pourcentages plus faibles pour les raisons qui suivent.

Dans certains cas, quelques relations sortantes d'un hyponyme ne sont pas valables pour les hyperonymes.
Prenons l'exemple suivant :

• A : *animal* • A' : *animal>metazoa* • B(hypo) : chat

▷ Le schéma va proposer la relation sortante de $chat$ (*chat* $\xrightarrow{is-a}$ *animal de compagnie*) à *animal>metazoa* (*animal>metazoa* $\xrightarrow{is-a}$ *animal de compagnie*) ce qui est faux et de là la précision baisse (*is-a* (**56,4%** par SIR_R(hypo) et **46%** par SIR_R(hyper))) et *has-part* (**46,9%** par SIR_R(hypo)).

Une autre raison est que dans le réseau quelques termes ne sont pas complétement raffinés ce qui peut induire des fausses inférences, comme par exemple:

• A : *fromage* • A' : *fromage>produit laitier* • B(hypo) : *chèvre*

▷ Le schéma va proposer la relation *fromage>produit laitier* $\xrightarrow{has-part}$ *mamelle* ce qui est faux mais causé par le fait que le terme *chèvre* n'est pas encore

raffiné en *chèvre>produit laitier* et *chèvre>animal*.

Avec les chiffres obtenus, les résultats montrent globalement que les inférences produites sont fortement valides pour la variante synonyme du schéma. Par contre ces résultats sont un peu plus faibles mais restent pertinents pour les deux autres variantes (tableau 2.8). Ceci est plus ou moins évident en sachant que pour la variante synonyme, les termes s'échangeant les relations sont presque sur le même niveau taxonomique. Ce qui n'est pas le cas quand ils sont hyponymes ou hyperonymes.

Rappel

En effet, nous sommes dans l'impossibilité de définir le rappel dans notre cas, vu que nous ne pouvons pas savoir s'il existe des relations correctes mais non inférées par notre système. Nous aurions déjà la ressource *parfaite* et *idéale* si c'était le cas.

2.5 Conclusion et perspectives

NOUS avons présenté dans ce chapitre l'un des composants du SCEREL (figure 5.1) qui est le système d'élicitation. Ce système se compose d'un moteur d'inférence et d'un réconciliateur. Nous avons détaillé les différents schémas d'inférences que nous avons mis en place pour définir le comportement du moteur d'inférences, ainsi que le processus de réconciliation. Le moteur d'inférence, à l'instar d'un contributeur humain, propose des relations potentielles. Ces dernières sont, par la suite, présentées pour validation / invalidation de la part des utilisateurs humains. Les inférences validées sont intégrées au RLS. Celles invalidées sont présentées au réconciliateur qui va, par le biais d'un dialogue avec l'utilisateur, essayer de trouver la cause de l'invalidation et y remédier (attribution d'un poids négatif pour les relations fausses, annoter les exceptions, raffiner des termes, etc.).

Ce système d'élicitation est totalement ouvert à diverses améliorations. Parmi lesquelles on peut citer la définition de la portée d'application de l'inférence spécialement dans les schémas de déduction et d'induction (exemple : *comment savoir à quel niveau de la hiérarchie s'arrêter pour*

éviter d'inférer des relations invalides. Comme dans le cas de proposer la relation du terme *animal* : *animal* $\xrightarrow{has-part}$ *aile* pour son hyponyme *chat* : *chat* $\xrightarrow{has-part}$ *aile* ou *poisson* : *chat* $\xrightarrow{has-part}$ *aile*) (figure 2.11).

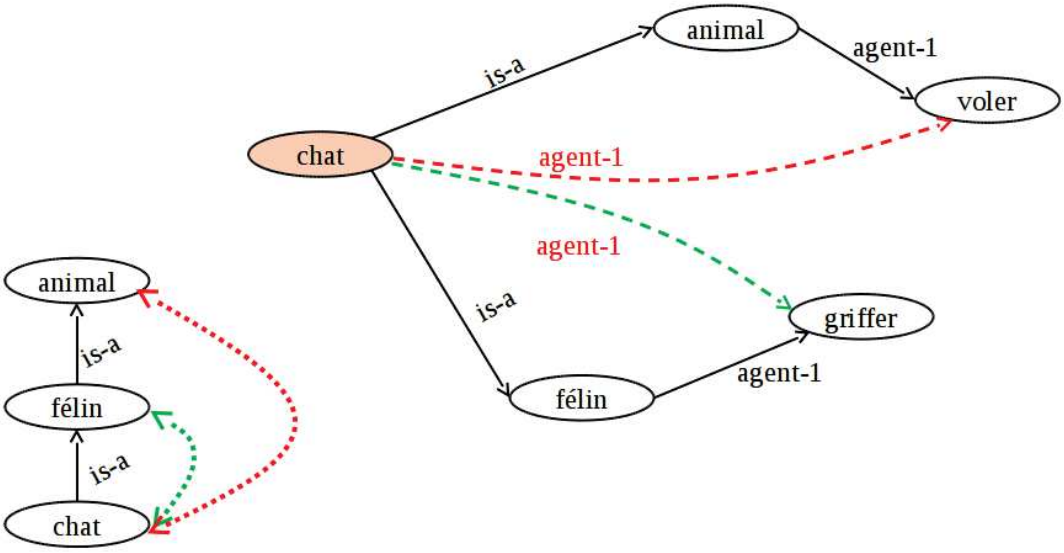

Figure 2.11 – Pour le terme *chat*, les inférences sont généralement valables en se basant sur le terme *félin*. Par contre, en se basant sur le terme *animal*, les inférences contiennent beaucoup de bruit. D'où l'intérêt de pouvoir indiquer la portée des schémas d'inférence pour réduire le taux de relations erronées.

Une autre piste possible serait d'améliorer le schéma d'inférence d'abduction pour qu'il puisse fournir des propositions de paramètres idéaux selon le terme sur lequel il va être appliqué. Par exemple pour un terme ayant **10** relations sortantes, définir un seuil de **6** pour la sélection des termes similaires peut nous fournir des inférences assez pertinentes, mais si nous définissons ce même paramètre (6) pour un terme ayant **100** relations sortantes, nous aurons en retour des inférences avec un taux d'erreur assez conséquent. *Dans ce cas comment déterminer un seuil idéal qui augmenter un maximum possible la pertinence et la véracité des inférences proposées?*

Une autre question se pose naturellement dans ce contexte. *Comment peut-on augmenter l'autonomie de ce système pour pouvoir lui donner la capacité de valider automatiquement les relations inférées ?*

Peut on définir un seuil de confiance au delà duquel les relations peuvent être considérées comme valides?

Chapter 3

ANNOTATION DE RELATIONS

Sommaire

3.1 Annotation de relation : définition, motivation et représentation

Qu'est-ce que l'annotation?

L'annotation est le processus consistant à attacher des informations complémentaires (généralement appelés méta-informations ou métadonnées) à des informations existantes. Un des processus d'annotation les plus connus est l'annotation sémantique de texte ou de corpus (Lefeuvre et al., 2014; Stenetorp et al., 2012) où les mots de ces derniers sont étiquetés avec des liens qui réfèrent à une description sémantique. En effet, lors du processus de l'annotation sémantique, des ressources lexicales sont utilisées pour étiqueter les textes et les corpus (Lefeuvre et al., 1998). Dans notre cas, nous allons annoter les relations associant les termes avec d'autres termes du réseau.

La plupart des ressources manquent de méta-informations d'ordre symbolique sur la force d'association (ce qu'on appelle poids dans le RLS JDM) et d'annotations (information sur la fréquence (fréquent, rare...) ou d'information sur la pertinence (pertinent ou pas). Par exemple, la relation entre *carcinome hépatocellulaire* et *wash-out* a un poids faible dans le réseau puisqu'elle est spécifique à la radiologie mais une information cruciale dans ce domaine. En conséquence, il est intéressant d'introduire les *annotations* pour certaines relations dans le réseau car elles vont être déterminantes pour améliorer la qualité des inférences et de l'analyse sémantique (Ramadier et al., 2014a,b,c).

Par exemple, dans la figure 3.1 parmi les annotations attribuées nous trouvons *fréquent* pour la relation *rougeole* $\xrightarrow{cible}$ *enfant* et *toujours vraie* à *rougeole* $\xrightarrow{charac}$ *contagieux*. Ces annotations apportent une méta-connaissance pertinente sur les relations du réseau, généralement absente des réseaux ou des ontologies, à la relation concernée et à l'ensemble du réseau lexical.

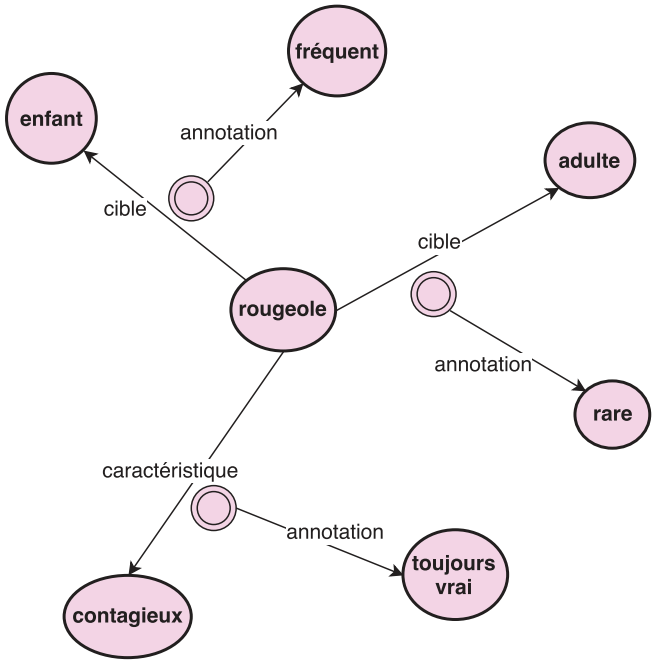

Figure 3.1 – Principe de représentation des annotation dans JDM : l'ajout d'une annotation se fait par la réification de la relation à annoter dans le réseau. Le nœud de relation créé peut être relié à d'autres nœuds (terme). La relation *annotation* est un type de relation tout comme le reste des types. La valeur de l'annotation *réification de la relation à annoter* $\xrightarrow{annotation}$ *valeur de l'annotation* est un terme du réseau.

Dans le réseau lexical JDM une relation est représentée par un triplet[1]
⟨nœud-début, type de relation/annotation, nœud-arrivée⟩

écrite *nœud-début* $\xrightarrow{type-de-relation/annotation}$ *nœud-arrivée*.

Les types d'annotation sont de différentes natures (information fréquentielle d'usage ou de pertinence). Ci-dessous, nous présentons les principaux types d'annotation :

- annotations fréquentielles : *très rare*, *rare*, *possible*, *fréquent*, *toujours vrai* ;

[1]Nous avons omis le poids pour simplifier.

- annotations d'usage : *souvent cru vrai*, *abus de langage* ;
- annotations quantitatives : *un nombre (1, 2, 4, etc.)*, *beaucoup*, *peu*, etc;
- annotations d'exception: *exception*;
- annotations qualitatives : *pertinent*, *non pertinent*, *inférable*.

Un médecin peut utiliser le terme *grippe* au lieu de *virus de la grippe*: c'est un **abus de langage**, le praticien fait simplement un raccourci de langage sans pour autant faire de confusion dans son esprit. Il semble évident que pour lui ces deux expressions sont sémantiquement différentes.

L'annotation **souvent cru vrai** s'applique pour une relation fausse (avec un poids négatif) qui est souvent considérée comme vraie, par exemple *araignée* (*is-a/souvent cru vrai*) *insecte*. Les exceptions sont également renseignées et prennent la forme d'une relation ayant un poids négatif. Ce type d'annotation est utilisé pour bloquer le schéma d'inférence.

L'annotation de nature **qualitative** est liée au statut inférable de la relation, particulièrement avec les méthodes d'inférence. L'annotation **pertinente** se rapporte à un niveau ontologique adéquat pour une relation donnée. Par exemple, *être vivant (charac / pertinent) vivant* ou *être vivant (charac / pertinent) mort*. L'annotation **inférable** est supposée être ajoutée quand une relation est inférable (ou a été inférée) à partir d'une relation existante, par exemple: *chien (charac / inférable) vivant* car *chien (is-a) être vivant*. L'annotation *non pertinent* est ajoutée aux relations vraies mais qui sont très éloignées du niveau ontologique pertinent, par exemple *animal (possède/non pertinent) atomes*.

L'annotation **quantitative** représente le nombre de parties d'un objet. Un être humain a généralement deux poumons alors nous avons

$$humain \xrightarrow{has-part/2} poumon$$

Ce type d'annotation n'est pas nécessairement numérique mais il peut aussi être symbolique avec *beaucoup* ou *peu*, etc.

L'annotation de **fréquence** peut être de plusieurs types (toujours vraie, fréquente, possible, rare et exceptionnel). La **qualitative** ayant

deux types pour le moment (pertinente et non pertinente). Nous avons attribué des valeurs d'encodage empiriques aux libellés d'annotation comme 4 pour toujours vraie, 3 pour fréquente, 2 pour possible, 1 pour rare et 0 pour le reste. Ces valeurs indiquent la priorité ou la force de l'annotation dans son genre et va nous permettre de sélectionner quelques annotations pour faciliter ou bloquer les schémas d'inférence.

3.2 Utilisation des annotations dans le moteur d'inférence

LES premières annotations ont été attribuées à la main mais sont ensuite automatiquement propagées dans le réseau à l'aide du moteur d'inférence. Pour améliorer la qualité du réseau lexical et éviter quelques inférences incohérentes, quelques types d'annotation peuvent bloquer l'inférence de certaines relations.

Principe de fonctionnement

Dans nos expérimentations présentées dans ce mémoire, nous utilisons uniquement le schéma d'inférence par déduction pour propager les annotations (rappel figure 3.2) afin de tester l'efficacité de l'idée des annotations dans le moteur d'inférence.

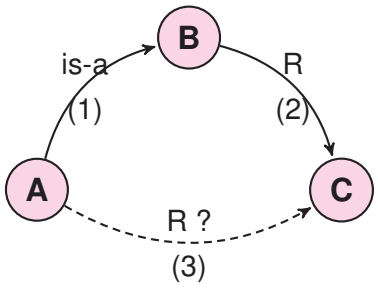

Figure 3.2 – Rappel du schéma simplifié de l'inférence par déduction.

Une relation inférée peut être issue de différentes prémisses. Supposons le terme *angora* (*nœud A* dans 3.2) qui a comme ensemble

d'hyperonymes *(nœud B)* {chat, félin, carnivore, animal, animal de compagnie ...}.
La relation inférée

$$angora \xrightarrow{has-part} griffe \text{ (nœud C)}$$

peut être issue de

$$chat \xrightarrow{has-part/toujours-vraie} griffe$$

$$félin \xrightarrow{has-part/toujours-vraie} griffe$$

$$carnivore \xrightarrow{has-part/fréquent} griffe$$

$$animal \xrightarrow{has-part/possible} griffe$$

et

$$animal\ de\ compagnie \xrightarrow{has-part/possible} griffe$$

Pour obtenir l'annotation la plus précise pour la relation

$$angora \xrightarrow{has-part} griffe$$

nous avons besoin d'ordonner les termes centraux (hyperonymes) du plus spécifique au moins spécifique. C'est-à-dire reconstituer les différents niveaux taxonomiques avec les hyperonymes (figure 3.3). Ce qui va nous donner pour notre exemple :

animal de compagnie<animal & chat<félin<carnivore<animal

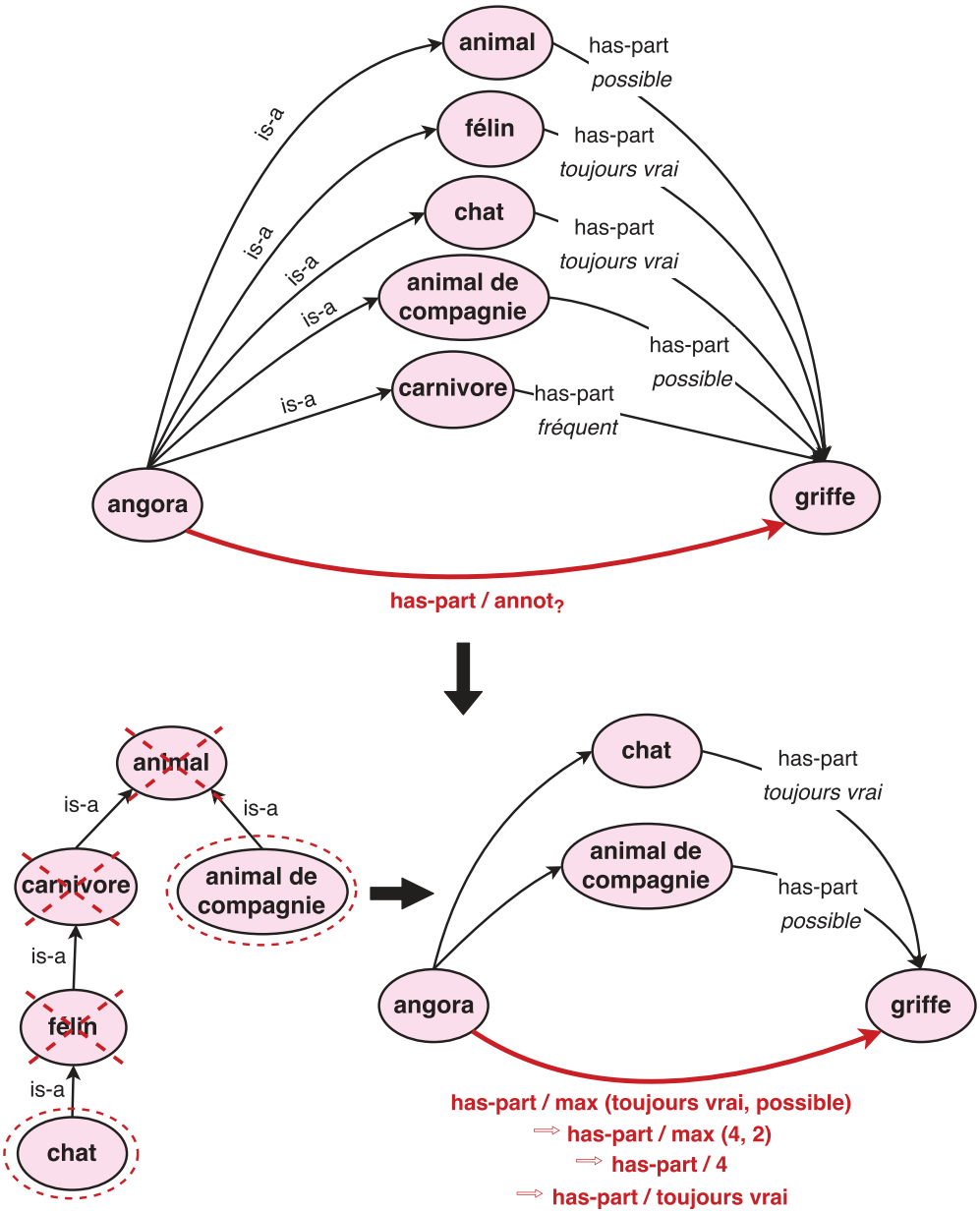

Figure 3.3 – Un exemple d'attribution d'annotation à la relation $angora \xrightarrow{has-part} griffe$.

L'annotation va être différente selon les prémisses utilisées pour inférer la relation. Pour choisir la meilleure annotation pour la rela-

tion inférée, l'ordre taxonomique joue un rôle important. C'est-à-dire, l'annotation de la prémisse basée sur l'hyperonyme le plus spécifique est potentiellement la plus valide pour la relation proposée.

Il est possible qu'il existe plusieurs niveaux taxonomiques, d'où plusieurs hyperonymes de même niveau. De façon similaire à l'exemple précédent où après élimination des hyperonymes les moins spécifiques (félin, carnivore, animal), deux hyperonymes du même niveau taxonomique (chat, animal de compagnie) restent. Ceci donne deux prémisses ayant des annotations différentes. Dans ce cas, nous appliquons la règle du maximum sur les valeurs respectives de chaque annotation (figure 3.4).

3.3 Expérimentations

DANS les expérimentations précédentes, nous avons utilisé soit le réseau dans son intégralité, soit un sous-réseau aléatoirement choisi. Dans cette partie nous avons fait fonctionner le système d'inférence par déduction et propagation d'annotation sur une sous-partie du réseau qui contient tous les hyperonymes et toutes les annotations manuellement attribuées, afin de réduire l'espace de recherche.

Première étape : inférer des relations

Pour augmenter la pertinence des résultats et éviter d'inférer des relations incorrectes (réduction de bruit), nous avons bloqué les inférences se basant sur des prémisses annotées comme *non pertinentes* ou *exceptions*. Le moteur d'inférence par déduction ici est appliqué sur **146 934** relations et a produit **1 825 933** relations dont **573 613** distinctes pour une moyenne de **3** occurrences par relation (tableau 3.1).

Relations existantes	146 934
Relations inférées	1 825 933
Relations distinctes inférées	57 361

Tableau 3.1 – Nombre de relations inférées par rapport à celles existantes.

Deuxième étape : propagation d'annotations de relations

Le moteur de propagation d'annotation est appliqué, en second lieu, sur la base de relations déjà enrichie par le moteur de déduction. Ce système d'annotation ne s'applique bien évidemment que sur les relations inférées et il prend en considération les annotations des prémisses utilisées pour inférer ces relations. S'il n'y a qu'une seule prémisse, l'annotation de cette prémisse, si existante, est attribuée à la relation inférée. S'il existe plusieurs prémisses, le système va reconstruire la hiérarchie entre ces dernières et prendre en compte le fait que l'annotation de la plus proche prémisse est la plus spécifique (figure 3.4). La procédure est décrite précédemment (3.2).

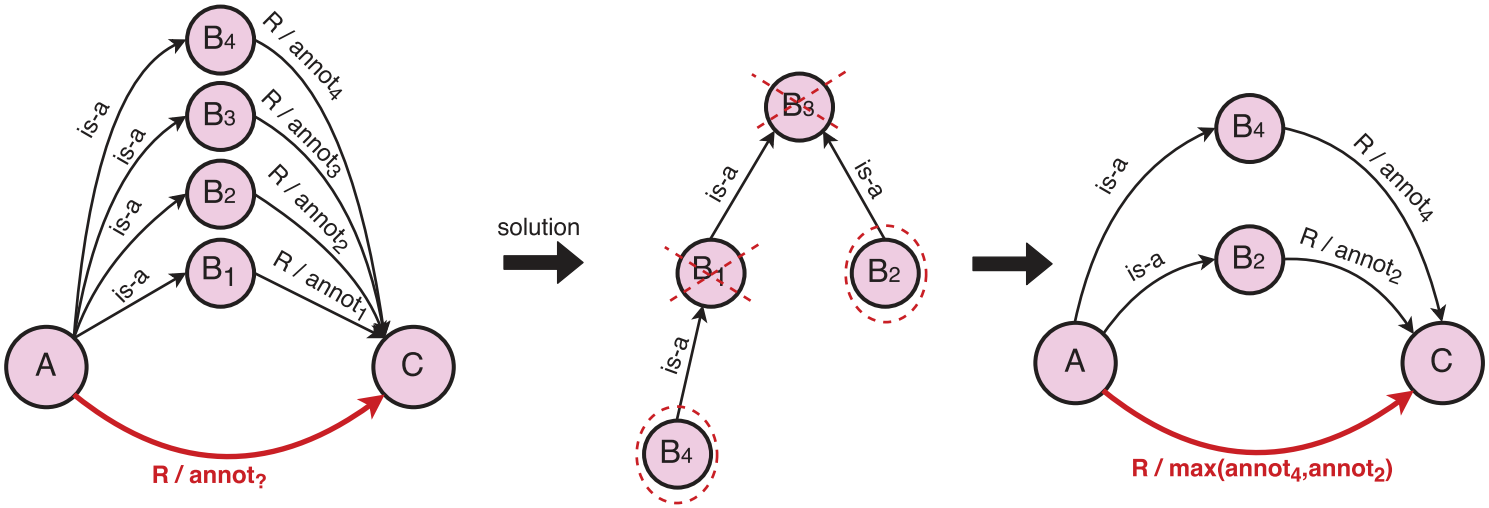

Figure 3.4 – Une stratégie basée sur la hiérarchie des hyperonymes pour choisir l'annotation la plus appropriée à donner à une relation inférée via plusieurs occurrences du schéma.

Comme nous pouvons le remarquer dans ces expérimentations, nous avons autorisé les redondances contrairement à l'application originale du schéma. En effet, cela augmente la pertinence du système de propagation d'annotations en fournissant quand c'est possible plusieurs occurrences différemment annotées pour la même instance de relation inférée.

Le moteur de propagation d'inférences, en une seule itération, a annoté **10 085** relations à partir de seulement **72** annotations (tableau 3.2).

Type d'annotation	**existantes**	**inférées**
Fréquentiel: toujours vrai	20	8 092
Fréquentiel: fréquent	18	1
Fréquentiel : possible	16	150
Fréquentiel : rare et très rare	7	35
Qualitatif : souvent cru vrai	1	7
Qualitatif : non pertinent	5	1 604
Quantificateur:	5	178
Total	72	10 085

Tableau 3.2 – Nombre d'annotations inférées après application du système d'annotation des relations sur celles existantes.

Comme l'indique le tableau, le nombre de relations annotées par libellé d'annotation ne dépend pas du nombre initialement existant de ces annotations 3.2, mais de certains faits expliqués ci-dessous :

1) Le schéma de déduction basique est le suivant:

$$A \xrightarrow{is-a} B \;\&\; B \xrightarrow{R/annotation} C \Rightarrow A \xrightarrow{R/annotation} C$$

Prenons par exemple les prémisses suivantes :

$$\textit{carcinome pulmonaire} \xrightarrow{is-a} \textbf{\textit{tumeur maligne}}$$

$$\textit{carcinome hépatocellulaire} \xrightarrow{is-a} \textbf{\textit{tumeur maligne}}$$

$$\textit{glioblastome} \xrightarrow{is-a} \textbf{\textit{tumeur maligne}}$$

et

$$\textbf{\textit{tumeur maligne}} \xrightarrow{charac/fréq} \textit{pauvre pronostic}$$

⇒ À partir des prémisses présentées ci-dessus 3 relations seront annotées comme fréquentes et cela est grâce aux multiples hyperonymes du terme *tumeur maligne* :

$$\textit{carcinome pulmonaire} \xrightarrow{charac/fréquent} \textit{pauvre pronostic}$$

$$\textit{carcinome hépatocellulaire} \xrightarrow{charac/fréquent} \textit{pauvre pronostic}$$

$$\textit{glioblastome} \xrightarrow{charac/fréquent} \textit{pauvre pronostic}$$

.

Plus nous avons des hyperonymes du B (*tumeur maligne*), qui lui possède une relation sortante annotée
(*tumeur maligne* $\xrightarrow{charac/fréquent}$ *pauvre pronostic*), plus le nombre de relations annotées sera grand.

2) L'existence des annotations qui ne génèrent pas d'autres annotation avec le système de propagation (comme pour l'annotation *fréquent* dans le tableau 3.2) peut s'expliquer par le fait qu'elles sont attribuées à des relations inéligibles pour le schéma déductif. Par exemple

$$\textit{carcinome hépatocellulaire} \xrightarrow{charac/fréquent} \textit{hypervascularisé}$$

Le terme *carcinome hépatocellulaire* n'a pas d'hyperonymes
($X \xrightarrow{is-a}$ *carcinome hépatocellulaire*), alors dans ce cas-là l'annotation *fréquent* ne va générer aucune autre annotation.

Nous avons évalué statistiquement les annotations produites et **87%** d'entre elles ont été évaluées comme *correctes*, **5%** comme *incorrectes* et le reste **(8%)** comme *discutables* (les experts peuvent discuter non pas leur validité mais plutôt si la valeur d'annotation doit être modifiée).

Dans ces expérimentations, nous avons appliqué le système inférence/annotation en une seule itération. Mais évidemment le système est en pratique lancé itérativement avec les contributions et les jeux et utilise les nouvelles relations et annotations rajoutées.

3.4 Conclusion et perspectives

Le système d'annotation décrit dans ce chapitre peut être vu comme un complément du système d'élicitation. En effet il représente un des composants du SCEREL (Système de Consolidation Endogène de Réseaux Lexicaux) qui fonctionne en coopération avec le moteur d'inférence de RS. Ce système propage, grâce à la procédure d'annotation, des informations sémantiques ou d'usages importantes qui peuvent être utilisées lors de l'analyse sémantique pour avoir une compréhension plus riche.

Les annotations mises à part leur informations rajoutées au réseau, servent de filtres aux schémas d'inférence. En effet, il est possible de paramétrer les schémas d'inférence pour ne proposer que les relations annotées comme toujours vraies, possible, fréquent, etc.

De futures pistes peuvent également viser à améliorer la diffusion des annotations de relations à travers le réseau et à l'appliquer si possible sur les autres schémas d'inférence.

Une possibilité est d'utiliser les annotations pour limiter la portée des inférences comme nous l'avons expliqué dans le chapitre 2 (2.5). Par exemple, nous pouvons limiter la portée du schéma d'inférence par les annotations *possible*, d'où nous éviterons d'avoir l'inférence *chat* $\xrightarrow{has-part}$ *aile* grâce à *animal* $\xrightarrow{has-part/possible}$ *aile*.

Chapter 4

EXTRACTION DE RÈGLES D'INFÉRENCE

Sommaire

DANS le cadre de l'enrichissement du RLS que nous avons effectué jusque là (l'inférence de relations à partir des relations préexistantes selon des schémas d'inférence et l'annotation de ces relations), nous avons mis en place un système de génération de règles d'inférence. Il représente une des composantes du SCEREL et il s'agit de produire automatiquement des règles d'inférence par l'inspection du contenu du réseau.

Les règles présentent une forme d'intention et nous permettent donc d'enrichir notre réseau par des connaissances qui vont s'appliquer par la suite pour générer des relations.

Une fois validée par un utilisateur, une règle s'applique sur les termes du réseau afin de produire de nouvelles relations dans le réseau. La polysémie et les silences présents dans le réseau présentent les principaux obstacles à l'identification statistique des règles correctes.

Mis à part le rôle de consolidation des règles d'inférence dans le réseau, nous visons un autre aspect pratique. En effet, en obtenant une base de règles validées pendant la procédure, il est possible d'enlever du réseau toutes les relations inférables par le biais de ces règles pour ne garder que les relations utilisées dans les prémisses. Ainsi, une sorte de réseau *déshydraté* ayant une taille minimale sera obtenu, sur lequel nous pouvons lancer un processus d'application des règles contenues pour avoir comme résultat le RLS dans son état complet. Il ne faut pour autant pas oublier de garder les relations ayant un poids négatif même dans l'état *déshydraté* du réseau parce qu'elles présentent les exceptions et de l'information importante dans leur négativité que nous obtenons des utilisateurs et que nous ne pouvons pas inférer automatiquement au moins dans l'état actuel du système (figure 4.1).

Un autre objectif poursuivi avec la conception de ce système est de fournir un aspect compact au réseau et à l'utilisateur la possibilité d'appliquer les règles sur des termes précis selon la demande.

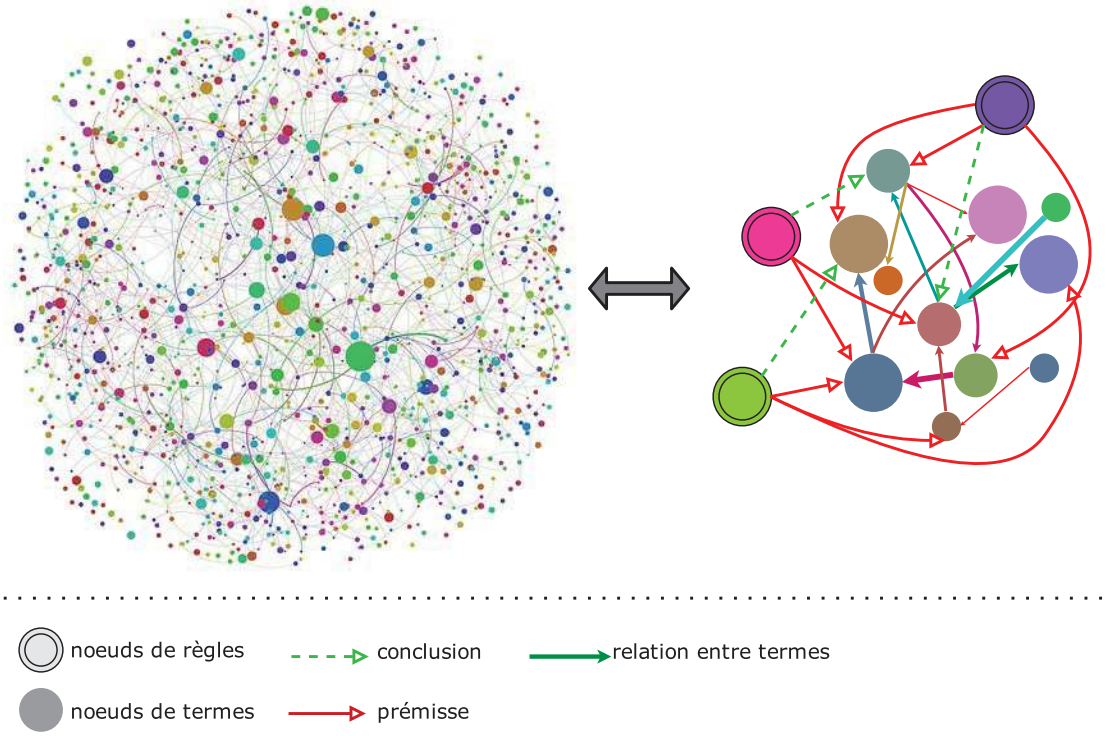

Figure 4.1 – Illustration du principe du réseau *déshydraté* : en ajoutant les règles d'inférence dans le réseau nous pouvons quasiment réduire une grande partie de ce dernier à ces règles, les relations dont elles ont besoin et les relations non-inférables. Le but est de passer d'un réseau de très grande taille à un petit réseau à partir duquel il est possible de régénérer ce dernier.

4.1 Principe de fonctionnement

NOUS avons conçu un *générateur de règles d'inférence* (GRI) qui vise à identifier et construire par inférence des règles potentielles à partir du RLS JDM (Lafourcade et al., 2013). Ces règles ont principalement la forme

"Si $\boldsymbol{x}$ *a une relation* $\boldsymbol{R}_b$ *avec* $\boldsymbol{B}$ *alors* $\boldsymbol{x}$ *a une relation* $\boldsymbol{R}_c$ *avec* $\boldsymbol{C}$*"*

plus formellement notée

$$\forall\, x, R_b(x,B) \Rightarrow R_c(x,C)\ [1].$$

[1] Une relation peut être écrite sous sa forme relationnelle $x \xrightarrow{type-relation} y$ ou sous

Le GRI agit en deux temps. Dans un premier temps, il identifie les règles et les propose. Dans un second temps et après la validation des règles, il applique certaines améliorations que nous allons détailler plus bas.

4.1.1 Génération de règles

Les règles potentielles sont présentées à un utilisateur humain pour validation/invalidation. Une règle valide est une règle vraie pour la plupart des x en omettant les exceptions qui doivent rester en nombre limité pour pouvoir continuer à les considérer comme des exceptions.

Nous allons prendre un terme cible au hasard dans le réseau (*voiture* par exemple).

- le GRI inspecte les relations sortantes ayant un poids assez fort ($\geq$50 : valeur choisie empiriquement) de ce terme : *voiture* $\xrightarrow{has-part}$ *roue*, *voiture* $\xrightarrow{lieu-1}$ *garage*, etc.;
- Pour chaque relation, il va énumérer tous les nœuds ayant des relations sortantes du même type et du même nœud cible : $\forall$ *terme* $\xrightarrow{has-part}$ *roue* où *terme* $\in$ {vélo, avion, charrette ...};
- Pour cet ensemble de nœuds n_i obtenus, il énumérera les occurrences de relations sortantes de tous les termes : *vélo* $\xrightarrow{has-part}$ *guidon*, *{vélo, avion, charrette...}* $\xrightarrow{agent-1}$ *rouler...*);
- Il indiquera :
  - le pourcentage de chaque relation sortante par rapport à tout l'ensemble (*exemples positifs*);
  - l'existence des *silences* (le nombre de termes vérifiant les prémisses mais pas la conclusion : un terme S où $S \xrightarrow{has-part}$ *roue* mais qui ne présente pas une relation sortante $S \xrightarrow{has-part}$ *guidon*);

sa forme logique *type-relation*(x,y)

– l'existence des *contre-exemples* (les termes satisfaisant les prémisses et ayant la conclusion avec un poids négatif : S où $S \xrightarrow{has-part}$ *roue* et $S \xrightarrow{has-part/poids<0}$ *guidon*).

L'idée sous-jacente est d'inspecter l'entourage d'un terme donné et d'y extraire les *patrons fréquents* de relations (nous considérons la relation à partir de 2 occurrences). Ces relations peuvent être considérées comme des règles d'inférence potentielles (figure 4.2).

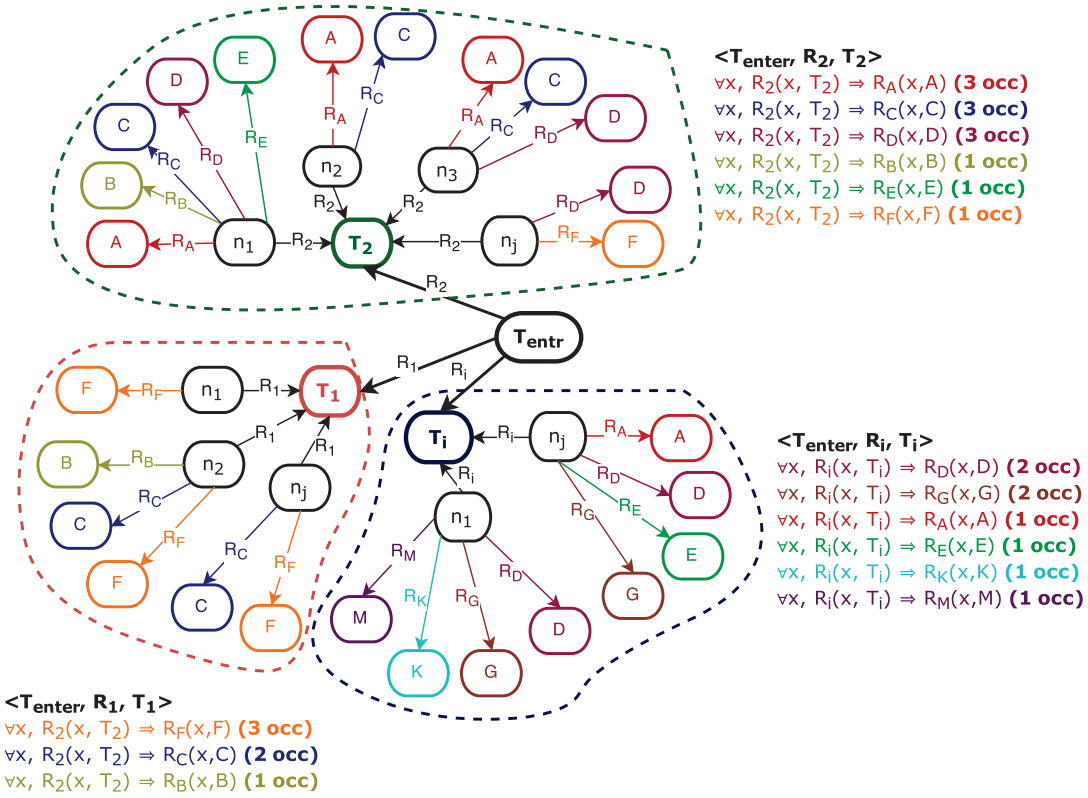

Figure 4.2 – Le GRI explore le voisinage du terme T_{entr}. Ce traitement est fait pour chaque relation sortante du T_{entr}. Pour la relation $T_{entr} \xrightarrow{R_2} T_2$, pour tous les termes ayant cette même relation $n_1, n_2, n_3, n_j \xrightarrow{R_2} T_2$, le système énumère toutes les instances de relations sortantes de ces derniers termes en indiquant leur nombre d'occurrences ($\forall\, x, R_2(x,T_2) \Rightarrow R_A(x,A)$: 3 occurrences)

La figure 4.2 présente le processus d'exploration du voisinage du

terme T_{entr} pour construire et proposer des règles:

- Le système parcourt toutes les relations sortantes de ce terme ($T_{entr} \xrightarrow{R_1} T_1$, $T_{entr} \xrightarrow{R_2} T_2$,...).
- Pour chacune de ces relations (qui va être ici la prémisse), le GRI énumère toutes les relations ayant le même type de relation et le même terme cible. Par exemple pour la relation $T_{entr} \xrightarrow{R_2} T_2$ nous allons avoir $\forall$ *terme* $\xrightarrow{R_2} T_2$.
- À partir de cet ensemble de termes, il inspectera toutes les relations sortantes en les groupant par type de relation et nœud cible d'où nous obtenons un ensemble de relations qui peuvent être les conclusions de nos règles. Les règles sont construites par la relation prémisse et la relation conclusion (**Si** $X \xrightarrow{R_2} T_2$ **alors** $X \xrightarrow{R_A} A$). Le système ne prend en considération que les règles ayant plus que 2 occurrences afin d'assurer un minimum de cohérence : l'existence d'une seule occurrence d'une relation peut provenir du fait que cette relation présente une exception, une relation fausse ou un manque de renseignement dans le réseau.

4.1.2 Traitement post-générateur des règles

Les règles générées sont validées manuellement par les utilisateurs pour être rajoutées dans le réseau, ou éventuellement invalidée et dans ce cas le système les garde dans une base qu'il consultera pour éviter de les proposer une autre fois.

Dans le réseau, nous ne garderons que les règles les plus générales car une règle valide pour un certain hyperonyme (classe) l'est pour tous les hyponymes de cet hyperonyme (sous-classes).

Prenons par exemple l'ensemble des règles suivantes:

- $\forall$ *x,is-a(x,chat)* $\Rightarrow$ *has-part(x,griffes)*
- $\forall$ *x,is-a(x,félin)* $\Rightarrow$ *has-part(x,griffes)*
- $\forall$ *x,is-a(x,tigre)* $\Rightarrow$ *has-part(x,griffes)*

- $\forall$ *x,is-a(x,angora)* $\Rightarrow$ *has-part(x,griffes)*

Sachant que ces règles ayant la même conclusion portent sur un arbre taxonomique (félin<tigre) et (félin<chat<angora), nous ne gardons dans le réseau que la règle la plus générale, dans ce cas

$$\forall \textit{ x,is-a(x,félin)} \Rightarrow \textit{has-part(x,griffes)}.$$

En aucun cas, l'élimination des règles *incluses* dans la règle plus générale ne causera une perte d'information, car l'application de la règle gardée fournira l'information nécessaire pour les schémas d'inférence présentés précédemment pour générer les autres relations.

Par ailleurs, les règles générées par le processus sont des règles simples, c'est-à-dire sous la forme (*prémisse* $\Rightarrow$ *conclusion*). Certaines règles sous cette forme atomique ne seront pas validées. Or, si nous les groupons par deux ou plusieurs prémisses et une conclusion ça les rendra potentiellement valides.

Par exemple, la règle $\forall$ *x,agent-1(x,voler)* $\Rightarrow$ *is-a(x,oiseau)* est invalide, ainsi que $\forall$*x,has-part(x,plumes)* $\Rightarrow$ *is-a(x,oiseau)*. Cependant, tandis que ces deux règles présentent la même conclusion, la construction d'une règle complexe à deux prémisses et une conclusion est possible. La règle complexe aura la forme :

$$\forall \textit{ x,agent-1(x,voler)} \wedge \textit{has-part(x,plumes)} \Rightarrow \textit{is-a(x,oiseau)}$$

et sera de fait une règle valide.

4.2 Expérimentations

NOUS avons lancé le processus d'extraction de règles d'inférences sur l'ensemble du réseau. Il est impossible cependant de parcourir la totalité du réseau : le processus durerait un temps important (plusieurs semaines) car il existe plus de 20 millions de relations. Dés lors, et pour un but démonstratif, nous avons décidé de limiter la portée du processus sur un sous-ensemble du réseau et cela en rajoutant un paramètre indiquant le poids des relations prises en considération lors de l'extraction de règles.

Le poids (Π) des relations dans ce RLS joue un rôle important indiquant la force d'association d'une relation. Nous avons donc considéré la courbe de distribution du nombre de relations selon le poids dans le réseau (figure 4.3) et nous avons remarqué un pic assez important aux alentours de Π=25 qui chute pour avoir un poids plus ou moins constant pour les autres relations.

Nous avons décidé alors de lancer le processus en premier lieu avec les relations ayant un poids $\Pi \geqslant 50$ afin de ne pas alourdir le système et obtenir des statistiques pour une première expérimentation et dans un deuxième lieu avec les relations ayant un poids $25 \leqslant \Pi < 50$ ce qui a pris plus de temps ($\simeq$ 2 semaines). Ces valeurs ont été choisies empiriquement dans un but uniquement démonstratif.

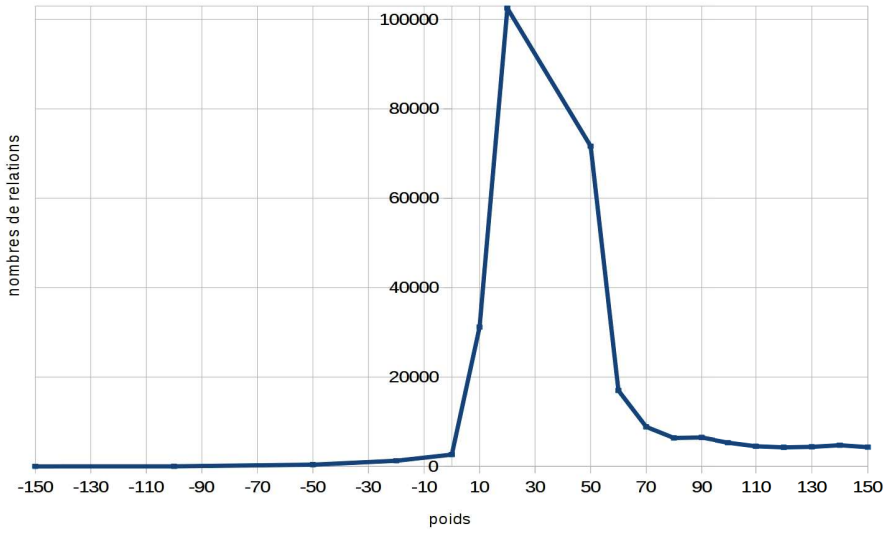

Figure 4.3 – Distribution partielle des relations selon le poids.

4.2.1 Expérimentation 1 ($\Pi \geqslant 50$)

Le processus d'énumération des candidats de règles a été lancé avec deux paramètres :
• nombre minimale d'occurrences = 2
• $\Pi \geqslant 50$.

En conséquence, le processus s'applique sur un sous-ensemble du

réseau de **4 148 585** relations (**18 971** termes) parmi **21 917 570** relations (**1 439 214** termes).

Avec ces paramètres, l'application de ce processus génère **19 482 240** règles. Puisqu'une même règle peut être générée plusieurs fois à partir du voisinage de plusieurs termes, c'est plus significatif de dire que ce processus a généré **322 279** instances de règles à valider (tableau 4.4).

Le taux de validation était de l'ordre de **12.44%**. Lors de l'étape de la validation/invalidation des règles, nous avons remarqué que la présence de la polysémie cause l'invalidation d'un grand nombre de règles. Par exemple pour la règle $\forall$ *x,agent-1(x,voler)* $\Rightarrow$ *has-part(x,ailes)*, la polysémie du terme *voler* (*voler>déplacement aérien*, *voler>appropriation du bien d'autrui*), engendre l'invalidation de la règle alors qu'elle aurait pu être valide si le terme avait été raffiné.

Pour améliorer le rendement en règles valides de notre système, nous avons relancé le processus avec les mêmes paramètres mais en prenant seulement les termes raffinés s'ils existent.

Sans polysémie (dorénavant un paramètre fixé dans notre jeu de paramètres dans ce processus), le processus génère **147 737** instances de règles potentielles avec un taux de validation de **20.88%**. Certes le nombre de règles valides reste exactement le même mais le taux de validité augmente en proportion (tableau 4.4 et courbes 4.5).

Pour mieux comprendre les expérimentations, nous définissons les points suivants:

Base: la cardinalité de l'ensemble de termes vérifiant la prémisse *(Prémisses)*.

Rappel: le ratio entre la cardinalité de l'ensemble de termes vérifiant la prémisse et la conclusion (règles) *Occurrences* et la cardinalité de la *Base*.

Silence: le ratio entre la cardinalité de l'ensemble de termes vérifiant la prémisse mais pas la conclusion *(Silence)* et la cardinalité de la Base.

Précision: le ratio entre la cardinalité de l'ensemble de règles valides et la cardinalité de l'ensemble de termes vérifiant la prémisse et la con-

clusion (règles) *Occurrences.*

Bruit: le ratio entre la cardinalité de l'ensemble de règles invalides et la cardinalité de l'ensemble de termes vérifiant la prémisse et la conclusion (règles) *Occurrences*.

$$\textbf{F-mesure} = \frac{2 * (\text{Précision} * \text{Rappel})}{(\text{Précision} + \text{Rappel})}.$$

Contre exemple: l'ensemble de termes vérifiant la prémisse mais ayant la conclusion avec un poids négatif.

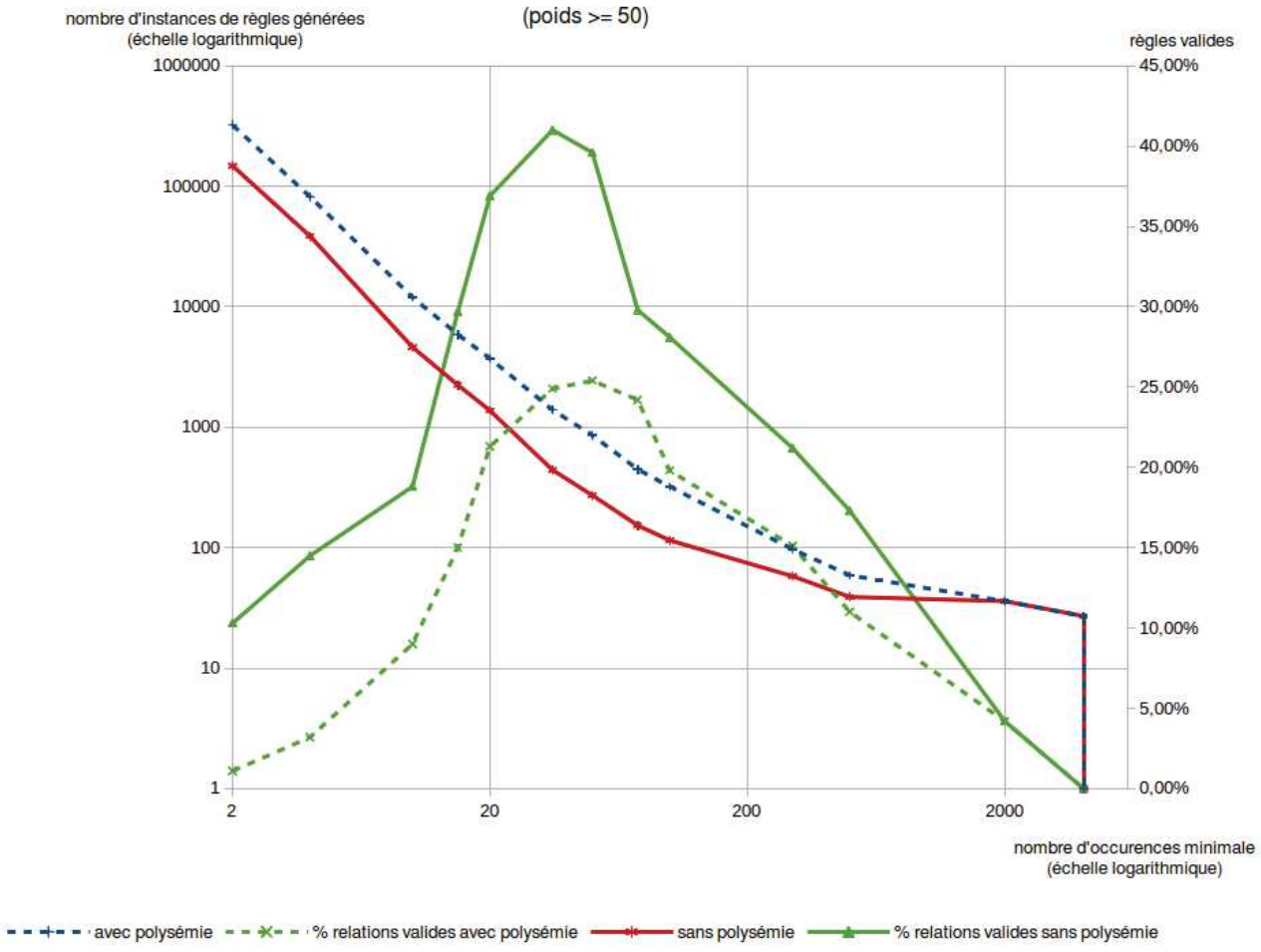

Figure 4.5 – Courbe d'instances de règles selon le nombre minimal d'occurrence et le taux de validité correspondant (avec et sans polysémie).

	avec polysémie			sans polysémie			
occ min	# règles	# instances de règles	taux de valides %	# règles	# instances de règles	taux de valides %	Fmesure
2	19 482 240	322 279	1,10	6 115 689	147 737	10,30	0,103
4	7 191 750	81 894	3,20	2 886 913	38 549	14,50	0,141
10	2 196 923	11 954	9,00	1 123 240	4 623	18,80	0,198
15	1 356 958	5 890	15,00	746 063	2 242	29,70	0,296
20	1 030 870	3 726	21,30	575 351	1 376	36,90	0,357
35	604 619	1 393	24,90	357 129	442	41,00	0,454
50	470 697	858	25,40	288 908	272	39,60	**0,474**
75	371 741	447	24,20	248 567	153	29,80	0,405
100	317 541	321	19,80	223 763	115	28,10	0,405
300	196 761	97	15,10	171 288	58	21,20	0,340
500	174 833	59	11,00	158 808	39	17,30	0,287
2 000	146 592	36	4,20	146 592	36	4,20	0,076
4 050	109 944	27	0	109 944	27	0	0
4 058	4 072	1	0	4 072	1	0	0

Figure 4.4 – Nombre de règles, d'instances de règles et leur taux de validité pour le processus ayant comme paramètre $\Pi \geqslant 50$, avec et sans polysémie.

En calculant la F-mesure pour chaque nombre minimal d'occurrences des règles, nous remarquons que le F-mesure le plus élevé est de **0.474** correspondant au nombre minimal d'occurrences égale à **50**.

Nous remarquons en observant les résultats, que plus on s'approche d'un nombre d'occurrences très bas ou très élevé, plus le taux de règles valides baisse. Plusieurs explications sont possibles.

En effet, pour un nombre minimum d'occurrences = **2**, nous remarquons que le taux de validité des règles est de **10,3%** (sans polysémie) contre **40%** pour **35** occurrences minimum. Ceci peut simplement s'expliquer par le fait que dans plusieurs cas deux occurrences ne suffisent pas pour rendre une règle valide. Par exemple, la règle :

$$\forall\ x, is\text{-}a(x, méduse) \Rightarrow charac(x, létal)$$

va être invalidée pourtant elle a été proposée suite à un minimum de 3 occurrences : *Carukia barnesi*, *Chironex fleckeri* et *Malo kingi* sont effectivement des méduses létales.

La validité des règles baisse aussi quand le nombre minimum d'occurrences tend vers des valeurs assez grandes, comme **4,2%** pour un minimum de **2 000** occurrences. Supposons que nous avons par exemple **3 000** espèces d'oiseaux dans le réseau renseignées comme étant des *vertébrés* et ayant comme lieu *arbre*. Le système en explorant les voisins de chaque terme correspondant à une des espèces de ces oiseaux, proposera la règle fausse $\forall\ x, is\text{-}a(x, vertébré) \Rightarrow lieu(x, arbre)$, ce qui fera **3 000** occurrences.

Ceci amène à se poser la question de la véracité des règles qui n'est pas strictement covariante avec la taille du support.

4.2.2 Expérimentations ($25 \leqslant \Pi < 50$)

Les paramètres lors de ces expérimentations sont :
• nombre minimal d'occurrences = 2
•$25 \leqslant \Pi < 50$.

Ici, le processus s'applique sur **5 878 050** relations (**60 960** termes). L'application de ce processus avec ces paramètres génère **1 463 427 083** règles correspondant à **9 700 581** instances de règles.

Malgré la lenteur du processus avec ces paramètres ($\simeq$ 2 semaines), nous avons remarqué que les règles proposées sont plus ciblées, par exemple

$$\forall\ x, agent\text{-}1(x, regarder) \Rightarrow has\text{-}part(x, oeil{>}vue)$$

contrairement à la majorité des règles proposées lors de la première expérimentation ($\geqslant 50$), par exemple

$$\forall\ x, agent\text{-}1(x, manger{>}se\ nourrir) \Rightarrow is\text{-}a(x, être\ vivant)$$

Le taux de validation des règles proposées par ce processus atteint **34%**.

4.2.3 Estimation de la véracité des règles

Nous avons tenté de mettre en place une approche pour permettre au système de prédire la véracité des règles à valider.

Hypothèse 1: Chaque règle proposée possède un certain nombre de *silences*. Un *silence* est un terme vérifiant la prémisse de la règle mais ne possédant pas la conclusion. Si un silence possède des relations sortantes du même type que celui de la conclusion de la règle, il est fort probable que ce silence soit un *contre-exemple caché*. Ce terme est bien renseigné tout en n'ayant pas la conclusion, ce qui rend probable le fait de pouvoir avoir la conclusion avec un poids négatif.

Exemple :

- $Aigle\ royal \xrightarrow{lieu} arbre{>}végétal$
  $\&\ Aigle\ royal \xrightarrow{is-a} oiseau$
- $Vautour\ moine \xrightarrow{lieu} arbre{>}végétal$
  $\&\ Vautour\ moine \xrightarrow{is-a} oiseau$
- $Alouette\ lulu \xrightarrow{lieu} arbre{>}végétal$
  $\&\ Alouette\ lulu \xrightarrow{is-a} oiseau$
- $nid{>}oiseau \xrightarrow{lieu} arbre{>}végétal$
  $\&\ nid{>}oiseau \xrightarrow{is-a} abri$
  $\&\ nid{>}oiseau \xrightarrow{is-a} endroit$

À partir de cet ensemble de relations, le système va proposer la règle :

$$\forall \text{ x,lieu(x,}\textit{arbre>végétal}\text{)} \Rightarrow \text{is-a(x,oiseau)}$$

avec un seul cas de silence (terme *nid>oiseau* qui vérifie la prémisse mais pas la conclusion).

Ce cas de silence présente d'autres relations du même type que la conclusion (*nid>oiseau* $\xrightarrow{is-a}$ *endroit*, *nid>oiseau* $\xrightarrow{is-a}$ *abri*) mais pas la conclusion ($x \xrightarrow{is-a} oiseau$).

Notre hypothèse est que tant qu'un terme est bien renseigné sur un type de relation, le fait de ne pas avoir une certaine relation de ce type implique qu'elle soit omise volontairement. Dans ce cas il est possible de la rajouter avec un poids négatif.

Cette stratégie permet au système de repérer les termes pouvant probablement être qualifiés de contre-exemples et d'estimer par la suite l'invalidation de la règle.

Hypothèse 2: Un cas de silence qui ne possède pas (ou très peu) de relations sortantes du même type que la conclusion, est vraisemblablement mal renseigné. Il est probable néanmoins qu'il supporte la conclusion en relation sortante potentielle.

Exemple :

- $angora \xrightarrow{is-a} chat>félin$
  $\& \; angora \xrightarrow{has-part} griffes$
- $manul \xrightarrow{is-a} chat>félin$
  $\& \; manul \xrightarrow{has-part} griffes$
- $persan \xrightarrow{is-a} chat>félin$
  $\& \; persan \xrightarrow{has-part} griffes$
- $siamois \xrightarrow{is-a} chat>félin$
  $\& \; siamois \xrightarrow{agent-1} miauler$

Dans ce cas, le système propose la règle $\forall$ *x,is-a(x,chat>félin)* $\Rightarrow$ *has-part(x,griffes)* mais avec un silence *siamois* qui ne possède pas visiblement de relations sortantes du même type que la conclusion. Le système suppose dans ce cas que le terme est mal renseigné et qu'il peut avoir la conclusion comme relation sortante.

Pour vérifier ces hypothèses:

- Nous définissons une variable V représentant le niveau de véracité d'une règle. Cette variable est *la moyenne des relations sortantes des silences ayant le même type que la conclusion*;
- Nous limitons le nombre de silences pris en considération à **1 000** (valeur empiriquement choisie);
- Nous comparons V à **5** (nombre maximal empirique au dessous duquel nous pouvons dire que la règle peut être vraie). Nous autorisons les silences à avoir au maximum 5 relations du même type que la conclusion pour rester dans la marge d'une règle pouvant être juste;
- Si $V < \mathbf{5}$ le système suggère que la règle peut être juste;
- Si $V > \mathbf{5}$, nous la comparons à OCC (75% de la moyenne des relations sortantes des termes occurrences ayant le même type de conclusion).
  - Si $V < OCC$ alors la règle est suggérée juste;
  - Sinon elle est suggérée fausse.

Pour mieux illustrer l'interaction entre les schémas d'inférences et les règles d'inférence ainsi que leur manière d'enrichir le réseau, nous avons sélectionné un sous-graphe ainsi que les règles et les schémas qui peuvent s'appliquer dessus. Une fois ces règles et schémas appliqués, de nouvelles relations sont rajoutées entre les différents nœuds. Notons que les relations rajoutées par les règles vont être utilisées par des schémas et vice versa (figure 4.6).

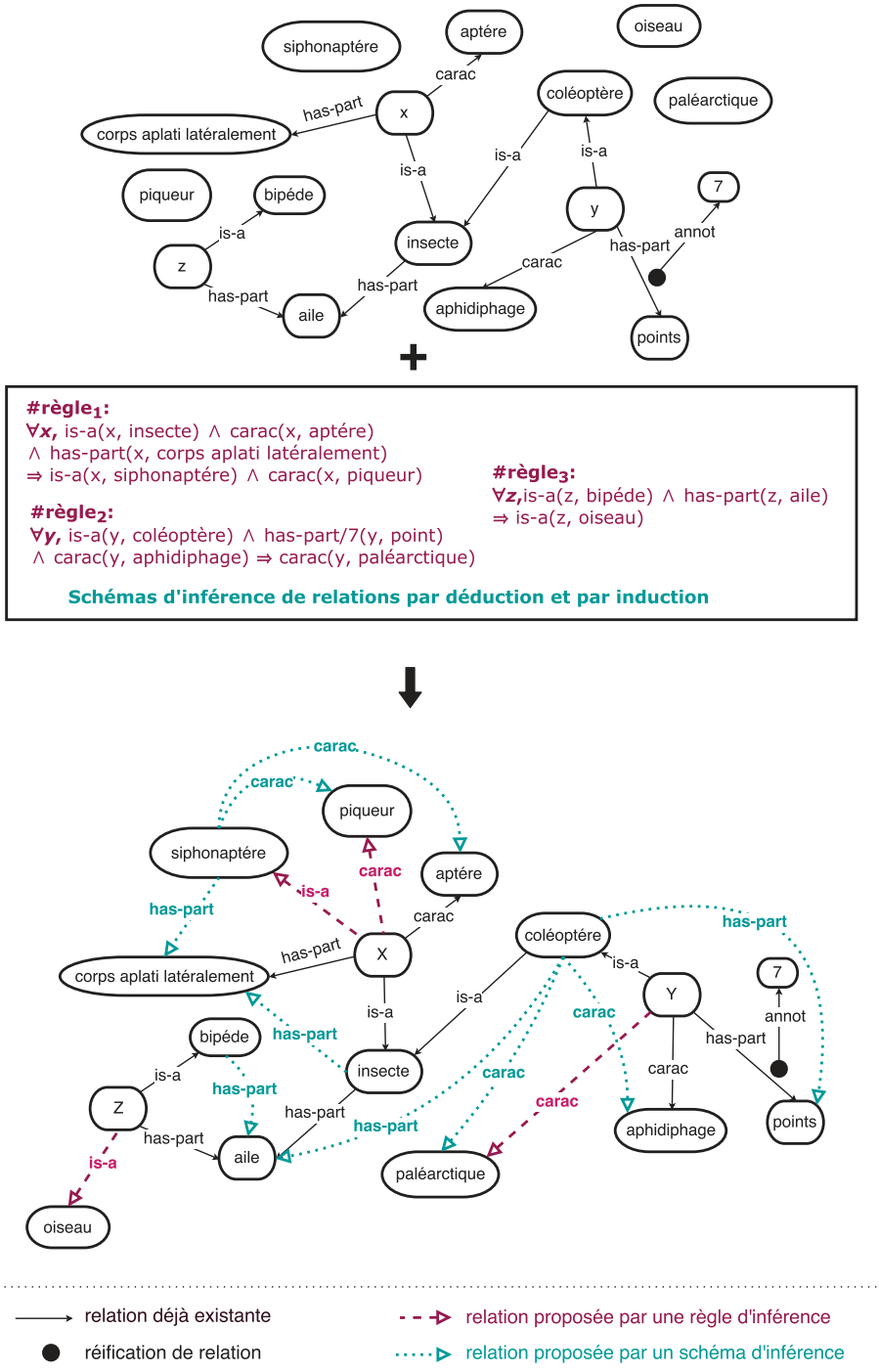

Figure 4.6 – Schéma démonstratif d'un exemple de sous-graphe du réseau et son enrichissement par règles et schémas d'inférences

4.3 Conclusion et perspectives

NOUS avons présenté dans ce chapitre la dernière composante du SCEREL : le système d'identification de règles d'inférence. Ce système propose, suite à l'exploration du RLS, des règles d'inférence potentielles sous la forme

$$\forall\ x, R_b(x,B) \Rightarrow R_c(x,C).$$

Un exemple de sous-graphe de réseau enrichi par les règles et les schémas d'inférence est illustré dans la figure 4.6.

L'une des améliorations à apporter au processus d'extraction de règles d'inférence serait de pouvoir trouver soit en pré-extraction soit en post-extraction, des règles à plusieurs inconnues sous la forme de

$$\begin{gathered}\forall\ x\ \forall\ y,\ carac(x,carnivore)\\ \wedge\ has\text{-}part(x,griffes\ non\ rétractiles)\\ \wedge\ lieu(y,mâchoire)\\ \wedge\ instr(y,déchirer)\\ \wedge\ has\text{-}part(x,y)\\ \Rightarrow\ is\text{-}a(x,canidé) \wedge\ is\text{-}a(y,croc)\end{gathered}$$

.

Un autre aspect important à poursuivre avec les règles validées comme nous avons dit précédemment, est de proposer des règles complexes et de les valider/invalider, ce qui peut augmenter le niveau de validité des règles et réduire le nombre de règles proposées (4.1.2).

Deux propositions de représentation de règles dans le réseau sont présentées ci-dessous (figure 4.7 et 4.8). Aucune d'elles n'étant appliquée actuellement, des réflexions sont portées sur celle qui satisfera au mieux les contraintes de taille, de flexibilité, de clarté, etc.

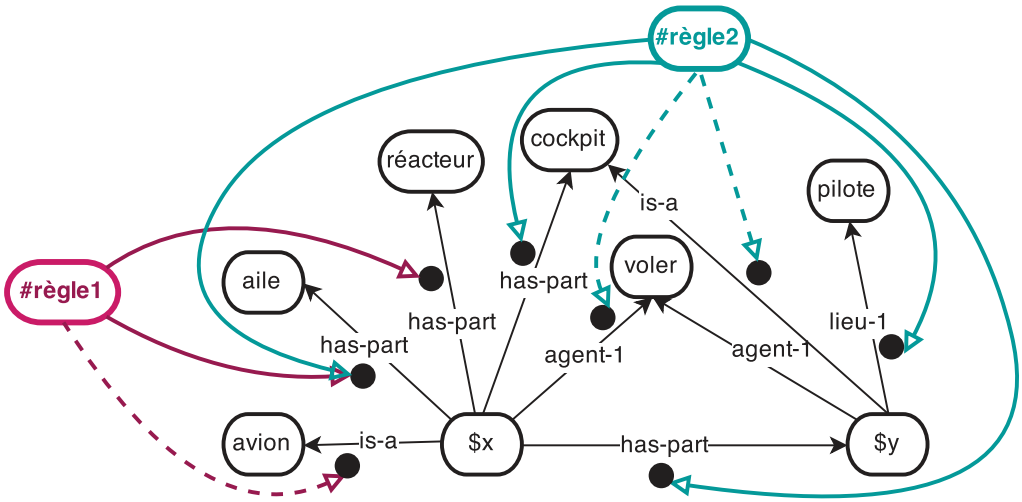

Figure 4.7 – Une modélisation de la représentation des règles d'inférences dans le réseau : des nœuds de variables sont créés et reliés par des relations du réseau à d'autres nœuds selon la règle à représenter. Des nœuds de type *règles* sont créés et reliés par des relations *prémisse* et *conclusion* à la réification des relations correspondantes. Par exemple la *# règle1* est reliée à la réification des relations représentant ses prémisses ($\$x \xrightarrow{has-part}$ *aile*, $\$x \xrightarrow{has-part}$ *réacteur*) et à la relation représentant sa conclusion ($\$x \xrightarrow{is-a}$ *avion*).

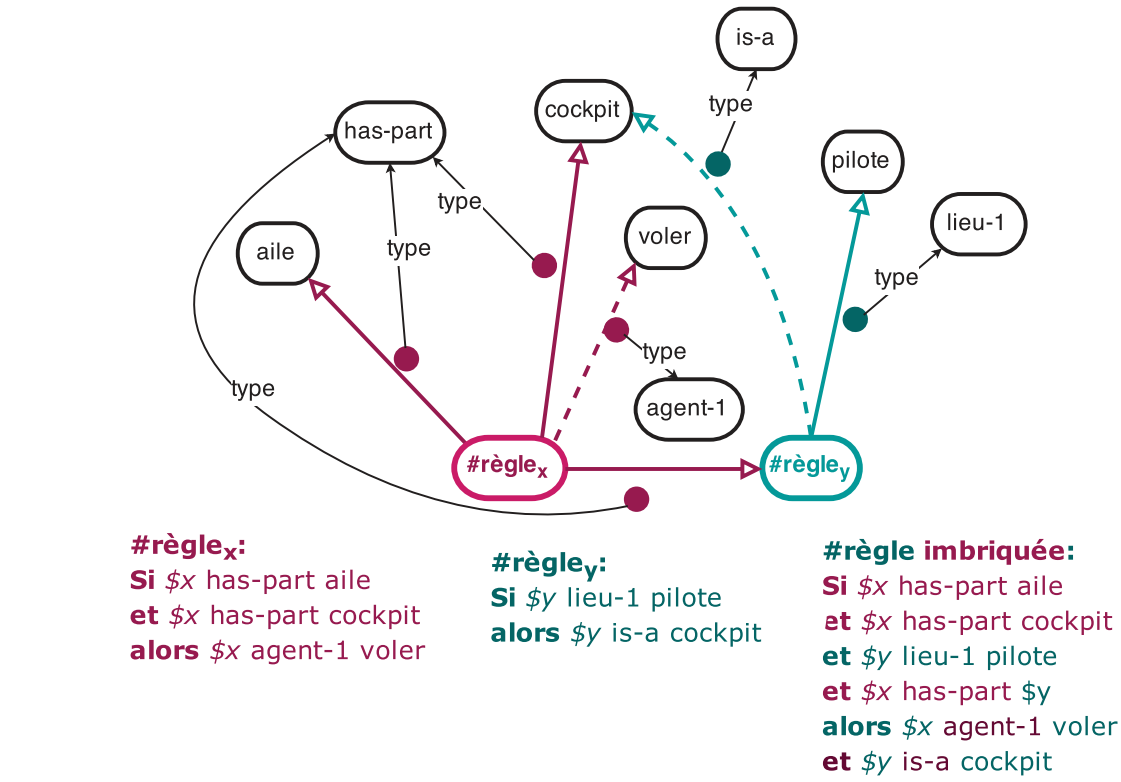

Figure 4.8 – Une manière de représentation des règles d'inférences dans le réseau : des nœuds avec les types de relations utilisés dans la règle sont créés. Des nœuds de type *règles* sont créés et reliés par des relations *prémisse* et *conclusion* aux autres nœuds de termes selon la règle à représenter. Les relations prémisses et conclusions sont ensuite réifiées et reliées par une relation de type *type* aux nœuds représentant les types de relations utilisées dans la règle. Dans cette modélisation les nœuds de type *règle* peuvent être reliés entre eux par des relation prémisse ou conclusion.

Une autre piste à poursuivre vers l'amélioration de ce système consiste à améliorer son approche d'identification de règles et sa capacité à

estimer la véracité d'une règle. D'une autre manière, rajouter de la sémantique à son fonctionnement car jusqu'à présent il suit une méthode totalement statistique.

Chapter 5

MaLLeN : Vers un langage de manipulation du système de consolidation endogène de réseaux lexico-sémantiques

Sommaire

5.1 Langage généraliste et langage dédié

QUEL que soit le programme fourni à une machine, son interprétation est faite sous forme binaire (1 et 0). Les humains sont certes capable de concevoir ces programmes, mais incapable d'interpréter naturellement ou de *parler* le langage binaire.

Cependant, ils peuvent apprendre aux machines de traduire n'importe quel langage en leur propre idiome. C'est la raison pour laquelle les humains, n'ayant pas besoin d'avoir un niveau d'abstraction binaire, essaient de garder un niveau d'échange intuitif avec la machine en élevant le niveau d'abstraction du langage de programmation utilisé. La programmation est certes une tâche orientée machine, mais si la portée de la compréhension et la modification de ces programmes doit inclure les humains, il faut prendre ces derniers en considération et ne pas les exclure lors du choix du niveau d'abstraction du langage de programmation.

Deux types de langages sont principalement utilisés pour programmer : les *Langages Généralistes (LG)* (GPL General Purpose Language) et *les Langages Spécialisés ou Dédiés (LD)* (DSL Domain Specific Language).

Une définition précise d'un LG est difficilement attribuable selon Watt (1990). Ce genre de langage est conçu pour résoudre tout type de problématique sans considération des domaines concernés. Ils représentent en général la préférence des programmeurs. Cette préférence s'explique par la grande communauté d'experts impliqués, l'efficacité et la maturité des langages proposés ainsi que l'existence d'environnements de développement efficaces (Hutchings and Nelson, 2000).

Néanmoins, ces LG ont des inconvénients, comme la difficulté de leur lecture, écriture, apprentissage et compréhension, dans le sens où ils exigent souvent un niveau de compétence et de connaissance élevé. Leurs syntaxe et sémantique ne sont pas assez intuitives à cause de leur généralité et leur propriétés qui s'approchent plus de la manière de fonctionnement de la machine que du raisonnement humain.

Quant aux LD (Visser, 2008), nous citons deux définitions.

Selon Deursen et al. (2000) :

> *Un langage dédié est un petit langage de programmation, généralement déclaratif, offrant à travers des notations et des abstractions appropriées, une expressivité puissante focalisée exclusivement sur un domaine particulier.*

Selon Fowler (2010) :

> *un langage dédié est un langage de programmation d'une expressivité limitée se focalisant sur un domaine particulier.*
>
> Cette définition contient 3 éléments clés :
>
> - *Langage de programmation* : Un LD est utilisé par les humains pour charger un ordinateur de faire quelque chose. Comme pour les langages de programmation modernes, sa structure est conçue pour rendre la compréhension humaine facile tout en restant exécutable par un ordinateur.
> - *Expressivité limitée* : Un LG fournit beaucoup de capacités : le support de diverses données, le contrôle et les structures d'abstraction. Tout cela est utile mais rend l'utilisation et la compréhension encore plus difficiles. Un LD maintient le strict minimum de fonctions nécessaires pour satisfaire les besoins de son domaine.
> - *Focalisation sur un domaine* : Un langage limité n'est utile que quand il a une focalisation claire sur un petit domaine. Cette focalisation est ce qui rend un langage limité profitable.

Les programmes écrits en LD ont l'avantage d'être plus concis et plus faciles à maintenir. En outre, ils sont en général écrits plus rapidement, plus faciles à comprendre et il est plus simple de mener des raisonnements avec (Hudak, 1997).

5.1.1 Caractéristiques des langages dédiés

CHAQUE LD a certaines caractéristiques qui lui sont propres mais la plus grande partie est commune (Oliveira et al., 2009). Généralement, les LD sont **concis** au sens du nombre de fonctions et structures de contrôle prédéfinies, et sont conçus pour traiter des problèmes d'un domaine spécifique. Les concepteurs exploitent les particularités de ce domaine pour créer une notation simple et concise mais puissante pour l'application visée. Cette notation permet de décrire la solution qui permet de résoudre le problème sans avoir à décrire comment cette solution sera faite, à la différence de ce qui se passe avec la majorité des LG. D'où, les LD sont plus **déclaratifs descriptifs** qu'impératifs (Deursen et al., 2000).

En outre, les LD sont **abstraits** (Mernik et al., 2005) et **expressifs** (Hudak, 1996). En effet, lors de la modélisation d'un LD, le concepteur doit être conscient du domaine dans lequel ce langage sera appliqué. La sémantique du domaine doit être claire et implicite dans la notation de ce dernier (Hudak, 1998). Cela veut dire que cette notation doit être abstraite, dans le sens où les notions du bas niveau indiquant la manière de faire les choses doivent être encapsulées avec des notations de haut niveau. Celles-ci doivent exprimer le but précis de l'existence de chaque expression.

Ces langages sont **efficaces** dans le sens où ils peuvent être lus et appris très facilement par des experts du domaine (Consel et al., 2005). Ces experts sont des personnes ayant une large connaissance d'un domaine donné, mais souvent avec peu ou pas d'expertise en informatique. Ainsi, avec l'abstraction et l'expressivité d'un LD, ils peuvent facilement lire les programmes et apprendre le langage afin de réaliser efficacement des programmes, c'est-à-dire en très peu de temps.

5.1.2 Quelques exemples de LD pour la programmation linguistique

Au tout début de leur apparition ces LD ont été appelés métalangages (Vauquois et al., 1967). Les structures de données que ces langages manipulent sont principalement des chaînes de caractères ou des graphes.

COMIT est sans doute le premier LD dans le domaine de linguistique et spécifiquement dans la traduction automatique (TA) pour le TALN. Il a été conçu au MIT (Institut de technologie du Massachusetts) par Yngve (1958). La première version a été mise au point en 1957.

Les systèmes-Q sont un autre exemple de LD pour la programmation linguistique (PL), ils ont été inventés par Colmerauer (1970) à l'Université de Montréal. C'était le premier pas vers la création de Prolog (Colmerauer et al., 1973).

Nous pouvons citer aussi GRADE (Nakamura et al., 1988) de l'Université de Kyoto, ATN (Augmented Transition Network) (Woods, 1970), etc.

5.1.3 Autres exemples dans divers domaines

LD	**Domaine d'application**
LaTeX, TeX, troff	Composition de documents
HTML, SGML	Balisage de documents
Emacs	Édition de texte
SQL, LDL, QUEL	Bases de données
Prolog	Logique
Unix shell scripts	Organisation de données

Tableau 5.1 – Quelques exemples de langages dédiés dans divers domaines.

5.2 MaLLeN : un langage de manipulation pour les réseaux lexicaux

5.2.1 Motivations

NOUS avons modélisé un langage dédié déclaratif qui permet à l'utilisateur mais aussi à la machine de manipuler le réseau lexico-sémantique et d'y appliquer les différents processus de notre système de consolidation selon ses besoins, tout en restant concentré sur sa requête et sans avoir besoin d'être un spécialiste du domaine d'accès aux bases de données ou d'utiliser un LG. Ceci aide l'utilisateur à configurer lui-même

des retours plus adaptés et à surmonter les limitations associées à la conception initiale.

Comme il est clairement dit dans (Lafourcade, 1994), contrairement aux langages généraux qui sont trop puissants et peu contraints pour être appliqués au domaine du TALN, les langages dédiés visent à fournir le minimum nécessaire pour pouvoir exploiter et manipuler les objets afin de créer une application adaptée aux besoins.

Un langage dédié, ayant un niveau d'abstraction plus élevé et des opérations spécifiques, est plus approprié au domaine pour lequel il est défini que les langages impératifs.

Dans notre cas, créer un langage dédié de manipulation de notre système, au lieu de se contenter d'utiliser un langage générique, est désirable dans le sens où cela satisfait et prend plus en considération les besoins de l'utilisateur. Un langage général ne fournit pas des fonctionnalités assez spécifiques pour le TALN, ce qui demande à l'utilisateur une capacité d'abstraction intellectuelle et des efforts de développement importants.

En outre, puisqu'un LD est plus petit et plus facile à comprendre, il permet aux non-programmeurs de voir le code des programmes, de le comprendre et de le modifier, en d'autres termes, de le superviser.

Notre LD que nous avons appelé MaLLeN (Manipulating Language for Lexical Network) se rapproche plus spécifiquement de ce qu'on appelle informellement un mini-langage. En effet, la conception de ce langage est spécifique au SCEREL.

Il permet à l'utilisateur de manipuler les composantes du SCEREL (figure 5.1), ainsi que d'écrire de petits programmes permettant à moindre coût d'avoir des retours pertinents. Cette manipulation consiste, d'une part, à interagir avec le RLS, comme par exemple rajouter / supprimer / renommer des termes / nœuds, rajouter / supprimer / annoter des relations / arcs, copier ou échanger certaines relations entre 2 termes, ajouter des raffinements de termes polysémiques (*papier>matière, papier>feuille de papier, papier>emballage, papier>document, papier>article, etc.*). D'autre part, appliquer tout type de schéma d'inférence, chercher des termes satisfaisant une ou plusieurs conditions (*tout terme étant un mammifère, ayant une queue et vivant dans l'océan*), chercher les règles valides ou visualiser les règles à valider / invalider autour d'un

terme donné ou un sous-ensemble de termes.

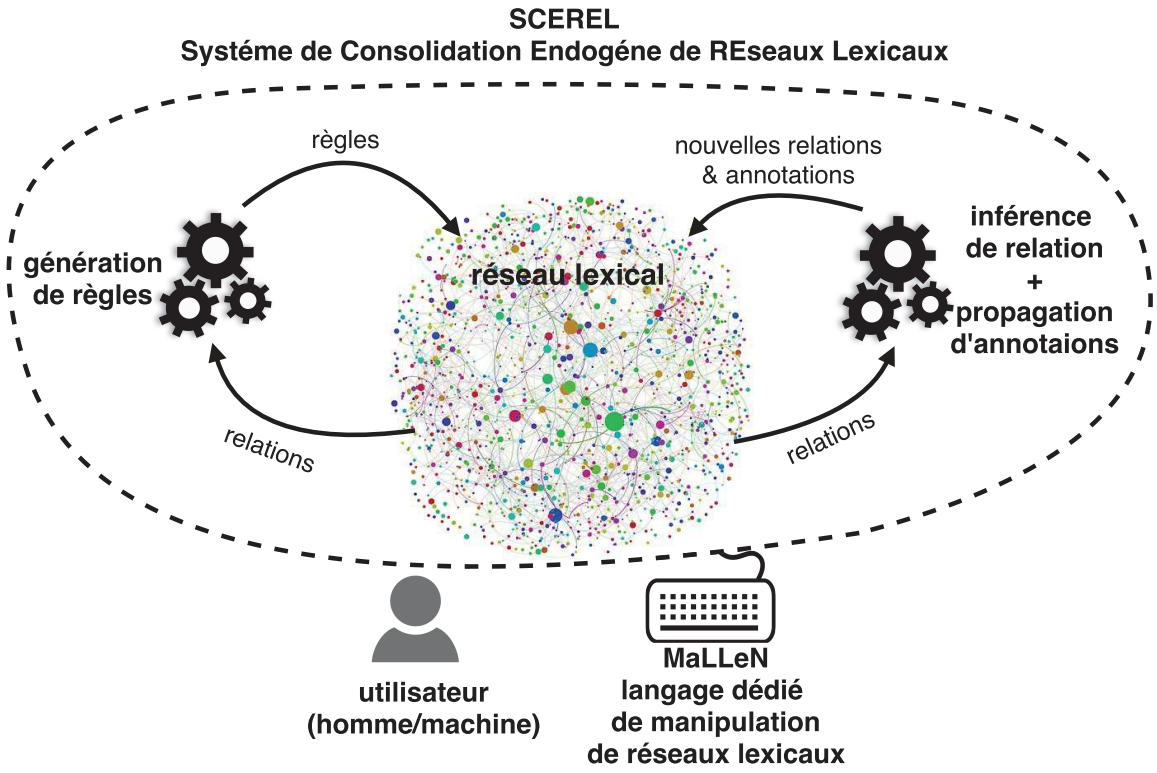

Figure 5.1 – Schéma global du système de consolidation avec les différentes composantes

5.2.2 Définition de la sémantique des fonctions du langage

NOTATION :
Nous avons utilisé la forme de Backus-Naur (BNF) (Backus, 1959) et EBNF (Scowen, 1993) pour décrire les règles syntaxiques de notre mini-langage.

En BNF, on distingue les méta-symboles, les terminaux et les non-terminaux. Les méta-symboles sont tout simplement les symboles de BNF (par exemple |). Les symboles non-terminaux sont les noms des catégories que l'on définit (label, relation_type,etc.), tandis que les terminaux sont des symboles du langage décrit (Create_term, As, From,

Between, Ref, etc.).

Prenons un exemple avec la commande qui permet de rajouter des termes dans le réseau lexical:
`<create_term_cmd> ::= create_term {<label>} [as <pos>];`
`<create_term_cmd>`, `<label>`, `<pos>` sont des non-terminaux.
`::=` est un méta-symbole signifiant "est défini par".
`Create_term` et `as` sont des terminaux.

Nous nous sommes permis quelques petites entorses par rapport au BNF:

- par souci de clarté nous avons omis les <> et écrit les terminaux en gras ;
- la majorité des non terminaux est définie en langage naturel, par exemple `relation_type ::= .. n'importe quel libellé de type de relations définies dans le tableau D.1 ..;` ;
- lorsqu'un non-terminal prédéfini apparaît plusieurs fois dans une même commande mais avec des occurrences qui doivent être différentes, nous rajoutons un indice sur chaque occurrence, par exemple si dans une commande nous avons besoin de deux *labels* différents nous les écrivons $\texttt{label}_i$ et $\texttt{label}_j$;
- les éléments suivis par (*) apparaissent zéro ou plusieurs fois;
- les éléments suivis par (+) apparaissent au moins une fois;
- les éléments entre [] sont optionnels;
- l'opérateur "|" a la signification de "ou/et".

Avec ces entorses, l'exemple donné précédemment devient (il s'agit la forme que nous allons utiliser dorénavant dans le manuscrit) :
`create_term_cmd : := create_term label [as pos];`
Les commandes du langage se divisent en des commandes de manipulation des objets du réseau lexical comme les termes, les relations et les annotations (create_term, copy_relation, share_relation, delete_annotation, etc.) et des commandes d'application de processus de schémas d'inférence

de relation, d'extraction de règles, de propagation d'annotation, etc. Nous définirons les non-terminaux au fur et à mesure de la définition des commandes du langage.

Fonctions de manipulation du réseau lexical

Create_term

Sémantique :
Créer un ou plusieurs nouveaux termes dans le réseau. Le système vérifie si les termes existent déjà dans le réseau. Si oui un message d'erreur s'affichera. Sinon de nouveaux nœuds avec les nouveaux termes se rajoutent dans le réseau.
En option, la partie du discours `(pos)` est affectée à chaque terme. Ce non-terminal doit appartenir à une liste prédéfinie de parties de discours (D.1). Pour la syntaxe voir B.1.

Exemple :
`create_term` `conscience` **`as`** `Nom:Fem+SG;`
Créer le terme `conscience` dans le réseau et créer une relation de type *pos* le reliant avec le terme `Nom:Fem+SG`.

Create_relation

Sémantique :
Créer une relation de type *relation_type* (liste prédéfinie) entre le terme $label_i$ (terme de départ) et $label_j$ (terme d'arrivée).
Il est possible d'indiquer en option le poids de la relation rajoutée `(weight)` et une ou plusieurs annotations (`annotation`). L'annotation se présente dans le réseau sous forme d'une réification avec l'identifiant de la relation reliée par une relation de type *annotation* à un terme qui désigne l'annotation. Pour la syntaxe voir B.8.

Exemple :
`create_relation` `target 50 fréquent` **`between`** `enfant` **`and`** `rougeole;`
Créer une relation de type `target`, ayant un poids de `50` et annotée `fréquent` entre le terme `enfant` et le terme `rougeole`.

Rename_term

Sémantique :
Renommer un terme $\texttt{label}_i$ par un $\texttt{label}_j$. Pour la syntaxe voir B.3.

Exemple :
rename_term avoir tord **as** avoir tort;
Renommer le terme existant avoir tord en avoir tort.

Annotate_relation

Sémantique :
Affecter une ou plusieurs annotation(s) à une relation déjà existante. Pour la syntaxe voir B.4.

Exemple :
annotate_relation consequence **between** vacciner **and** encéphalite **as** rare;
Annoter la relation de type consequence entre vacciner et encéphalite comme rare.

Delete_term

Sémantique :
Supprimer un ou plusieurs termes existants et leurs relations associées. Pour la syntaxe voir B.5.

Exemple :
delete_term obsolète;
Supprimer le terme obsolète du réseau ainsi que ces relations sortantes et entrantes.

Delete_relation

Sémantique :
Supprimer une relation ou toutes les relations sortantes d'un terme $\texttt{label}_i$ vers tous les termes ou vers certain(s) terme(s) $\texttt{label}_j^+$. Pour la syntaxe voir B.6.

Exemple :
delete_relation `all` **from** `table` **to** `pont;`
Supprimer toutes les relations allant du terme `table` au terme `pont`.

Delete_annotation

Sémantique :
Supprimer une ou plusieurs annotations sur une relation. Pour la syntaxe voir B.7.

Exemple :
delete_annotation `all` **of** `is-a` **between** `chat` **and** `félin;`
Supprimer toutes les annotations de la relation de type `is-a` entre le terme `chat` et `félin`.

Copy_relation

Sémantique :
Copier toutes les relations (sortantes et/ou entrantes) ou un certain type de relation d'un terme $\texttt{label}_i$ vers un ou plusieurs termes $\texttt{label}_j$. Pour la syntaxe voir B.8.

Exemple :
copy_relation `out all` **from** `chat` **to** `matou, minet, mistigri;`
Copier toutes les relations sortantes du terme chat vers les termes `matou`, `minet` et `mistigri`.

Exchange_relation

Sémantique :
Affecter toutes les relations (sortantes et/ou entrantes) ou un certain type de relation d'un terme $\texttt{label}_i$ à un terme $\texttt{label}_j$ et vice versa. Pour la syntaxe voir B.9.

Exemple :
exchange_relation `out in all` **between** `aigle royal` **and** `aigle impérial;`

Affecter toutes les relations entrantes et sortantes du terme `Aigle royal` à `Aigle impérial` et vice versa.

Fonctions d'application des processus de consolidation

Reason_by_deduction

Sémantique :
Appliquer le schéma de déduction 2.2.1 pour inférer des relations potentielles à partir d'un terme donné $\texttt{label}_i$ en tenant compte de tout type ou d'un certain type de relation. Il est possible d'indiquer le terme intermédiaire $\texttt{label}_j$ (terme B dans le schéma précédemment présenté). Les relations proposées seront sous la forme

$$\texttt{label}_i \xrightarrow{type-de-relation-choisi} \textit{terme d'arrivée}$$

Pour la syntaxe voir B.10.

Exemple :
`Reason_by_deduction` `sentiment` **`start`** `transplantation xénoplastique` **`end`** `opération chirurgicale;`
Inférer des relations de type `sentiment` en partant du terme `transplantation xénoplastique` et en se basant sur son hyperonyme `opération chirurgicale`. Avec cette fonction le système propose *opération chirurgicale* $\xrightarrow{sentiment}$ *angoisse>peur*. En passant par l'hyperonyme *transplantation*, aucune inférence ne sera faite à cause du manque de renseignement pour ce terme. Cette situation est très vite résolue dès que le schéma de déduction s'applique sur le terme *transplantation*.

Reason_by_induction

Sémantique :
Appliquer le schéma d'induction 2.2.2 pour inférer des relations potentielles à partir d'un terme donné $\texttt{label}_i$ en tenant compte de tout type ou d'un certain type de relation. Il est possible d'indiquer le terme intermédiaire $\texttt{label}_j$ (terme B dans le schéma précédemment présenté). Dans ce cas les relations proposées seront sous la forme

$$\texttt{label}_j \xrightarrow{typederelationchoisis} \textit{terme d'arrivée}$$

Pour la syntaxe voir B.11.

Exemple :
Reason_by_deduction `charac` **start** `coccinelle asiatique` **end** `coccinelle;`
Inférer des relations de type `charac` en partant du terme `coccinelle asiatique` et en se basant sur son hyperonyme `coccinelle`. Avec cette commande le système propose

$$coccinelle \xrightarrow{charac} aphidiphage$$

.

Reason_by_abduction

Sémantique :
Appliquer le schéma d'inférence par abduction 2.3 au terme donné `label`. Pour la syntaxe voir B.12.

Exemple :
Reason_by_abduction `all` **to** `tourterelle ;`
Inférer des relations potentielles pour le terme `tourterelle`.

Search_rules

Sémantique :
Extraire les règles potentielles autour du terme donné `label` en ne se basant que sur les relations ayant un poids $\geqslant$ `weight` ou un poids compris entre $\texttt{weight}_{min}$ et $\texttt{weight}_{max}$ (rappel sur le principe 4.1.1). Pour la syntaxe voir B.13.

Exemple :
search_rules `25 70` **from** `bâtiment ;`
Chercher les règles potentielles autour du terme `bâtiment` et en ne se basant que sur les relations ayant un poids entre `25` et `70`.

Give

Sémantique :
Lister les termes `$variable` satisfaisant toutes les `prémisses` données. Pour la syntaxe voir B.14

Exemple :
give $x **where** $x is-a cnidaire **and** venin lieu $x **and** $x has-part mésoglée **and** $x agent-1 pulluler;
Énumérer tous les termes étant un cnidaire, contenant du venin, ayant comme partie de la mésoglée et pouvant pulluler.

5.3 Conclusion et perspectives

DANS ce chapitre, nous avons présenté la différence entre les langages de programmation généralistes et les langages dédiés. Nous avons exprimé les motivations à la base de la création de MaLLeN notre langage dédié à la manipulation du système de consolidation endogène de réseaux lexicaux (SCEREL).

MaLLeN est un langage dédié conçu pour être utilisé des utilisateurs humains ainsi que des machines Il leur permet de manipuler les objets du réseau comme les termes, les relations, les annotations, etc.

Ce langage représente un premier pas et un premier essai vers un langage complet de manipulation de structures de graphe représentant de la connaissance lexico-sémantique

Une perspective très importante consiste à rendre le système d'extraction de règles d'inférence capable de générer des programmes dans ce langage, ce qui va permettre à l'utilisateur, qu'il soit homme ou machine de comprendre le raisonnement suivi et de le modifier en cas de besoin.

CONCLUSION GÉNÉRALE ET PISTES DE RÉFLEXION

Système de consolidation endogène de réseaux lexico-sémantiques

NOUS avons proposé dans cette thèse une approche endogène qui vise à enrichir et consolider des réseaux lexico-sémantiques construits par externalisation ouverte, en produisant des inférences de nouvelles relations à partir de celles existantes, et cela sans exploiter des ressources extérieures.

Dans nos travaux, nous nous fondons sur le réseau lexico-sémantique issu du projet JeuxDeMots (Lafourcade, 2007) pour développer les aspects théoriques de la consolidation endogène ainsi que pour mener des expériences pratiques.

Nous avons concrétisé cette approche sous forme d'un système de consolidation endogène de réseaux lexico-sémantiques (SCEREL). Ce dernier se compose d'un *système d'inférence de relations sémantiques et d'annotations*, un autre pour *l'extraction de règles d'inférence* ainsi qu'*un langage dédié de manipulation* de ce système de consolidation (figure 5.1).

Le système d'inférences de relations sémantiques et d'annotations utilise des schémas pour proposer de nouvelles relations potentielles à partir de celles déjà existantes dans le réseau et pour propager automatiquement les annotations de relations introduites initialement à la main.

Les schémas d'inférences qui guident le comportement de ce système sont de deux types : des schémas statiques et des schémas dynamiques. *Les schémas statiques*, comme la déduction (2.2.1) et l'induction (2.2.2), s'appliquent en tant que tels. En effet, ils gardent leurs formes de schémas et cherchent dans le graphe des combinaison de termes / relations qui leur sont conformes. Alors que *les schémas dynamiques*, comme l'abduction (2.3) et le schéma d'inférence par raffinement (2.4), s'appliquent sur les termes séparément en s'adaptant à leurs relations entrantes / sortantes.

Les relations inférées sont soumises pour vote aux contributeurs qui peuvent voter pour ou contre et par la suite proposées à une validation ou invalidation par un expert. Une grande majorité des inférences se révèle correcte. Toutefois, il convient de comprendre la raison de l'invalidation des relations rejetées. Le processus responsable de cette tâche est la réconciliation qui, menée à l'aide d'un dialogue avec l'utilisateur, essaie d'expliciter en quoi la relation considérée est incorrecte. Les causes possibles sont de trois natures : (1) erreur dans une des prémisses, (2) exception, ou (3) confusion liée à la polysémie.

Ce système, outre l'accomplissement de sa tâche principale qui consiste à consolider le réseau en densifiant les relations, s'avère être aussi un bon détecteur d'erreurs présentes dans le réseau, un classificateur par abduction, un identificateur de polysémie, un marqueur d'exception et un annotateur de relations valides non pertinentes.

Le système d'extraction de règles d'inférences propose automatiquement des règles d'inférence potentielles par l'inspection du contenu du réseau lexico-sémantique. Une règle a principalement la représentation formelle suivante $\forall\ x, \{r_i(x,B_i)\}^+ \Rightarrow \{r_j(x,C_j)\}^+$ qui peut varier selon le nombre de prémisses et conclusions de la règle. Une règle une fois validée par un utilisateur, sera appliquée afin de produire de nouvelles relations dans le réseau. Une règle valide est une règle vraie pour la plupart des cas en omettant les exceptions qui doivent rester en nombre limité pour être toujours considérées comme telles.

En mettant ce système en place, le but principal étant d'enrichir le réseau en proposant de nouvelles relations potentiellement correctes, nous nous sommes aperçu de la possibilité d'une nouvelle facette du système de consolidation. Cet aspect permet d'obtenir un réseau léger en taille, ce qui va favoriser le transfert de données, et plus flexible en termes d'utilisation.

En obtenant une base de règles valides, nous pouvons retirer du réseau toutes les relations inférables par le biais de ces règles pour ne garder que les relations utilisées dans les prémisses. Nous obtenons un réseau *déshydraté* ou *noyau*, ayant une taille minimale et sur lequel nous pouvons lancer le processus d'application des règles contenues et le moteur d'inférences pour avoir comme résultat le réseau dans son état développé. Il faut conserver les relations ayant un poids négatif même dans l'état *déshydraté* du réseau parce qu'elles présentent les exceptions et les inhibitions qui sont une information importante dans leur négativité. Ces relations sont obtenues des utilisateurs et nous ne pouvons pas les inférer automatiquement pour le moment (figure 4.1).

Nous avons mis en place une première version de *MaLLeN,* **un langage dédié** déclaratif de manipulation du système de consolidation, ce qui va donner la possibilité à des utilisateurs externes (être humain ou machine) de manipuler le réseau et d'utiliser le système de consolidation.

Ce langage réduit offre aux utilisateurs externes à travers des abstractions appropriées, une expressivité importante concentrée exclusivement sur notre système de consolidation.

Pour récapituler, nous avons donc pu à travers cette thèse répondre positivement aux questions posées préalablement en mettant en place une première brique d'une intelligence artificielle se basant sur un réseau lexico-sémantique (dans notre exemple celui de JeuxDeMots), disposant d'une capacité de raisonnement continu et autonome sur le contenu de ce réseau au fur et à mesure de la connaissance collectée à travers les jeux et l'outil contributif (figure 5.2).

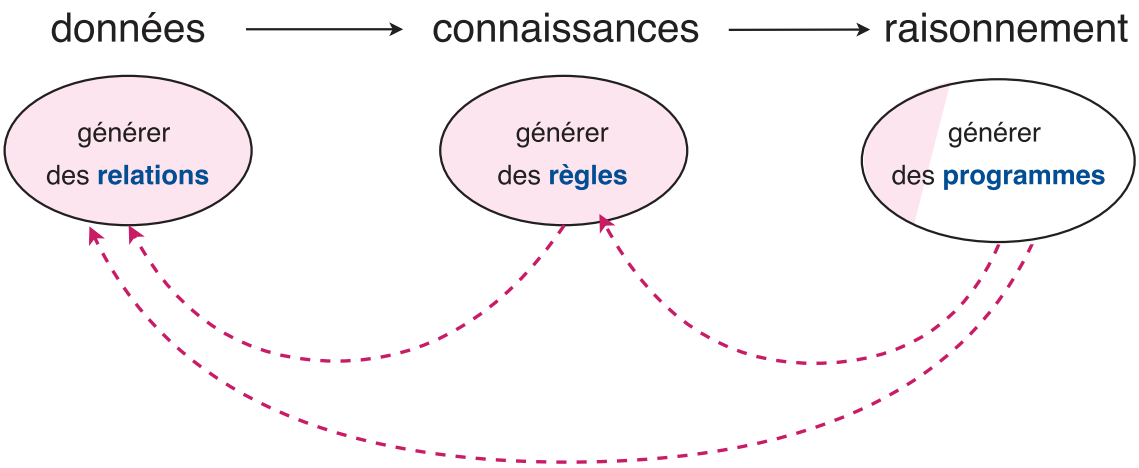

Figure 5.2 – Information - Connaissances - Intelligence. Un premier pas vers une intelligence artificielle pouvant générer des programmes avec le langage dédié présenté dans ce mémoire

Perspectives et pistes de réflexion

BIEN évidemment, notre système est susceptible de multiples améliorations et invite à de nombreuses perspectives aussi bien théoriques que pratiques.

Parmi ces perspectives nous pouvons citer la limitation de la portée des schémas d'inférence, qui pourra améliorer la qualité des inférences surtout pour la déduction et l'induction. Cette limitation consiste à savoir jusqu'à quel niveau de hiérarchie, les inférences par induction ou déduction restent valables pour un terme ou un usage donné (figure 2.11).

La (semi)-automatisation de la validation des inférences proposées (faite actuellement par des locuteurs humains) peut être un bon pas pour un système complètement ou semi-autonome. Mais comment mettre en place une approche d'évaluation de la confiance pour ce système de consolidation? Jusqu'à quel point nous devons déléguer le contrôle dessus?

L'amélioration du langage dédié, mis en place pour l'administration

du système de consolidation, peut être nécessaire pour avoir un langage de manipulation de graphes adapté aux changement de besoins.

Le système d'extraction de règles est un terrain favorable d'expérimentations avec de multiples améliorations possibles. Par exemple, optimiser l'approche de recherche de règles d'inférence afin d'augmenter la précision de ces règles, ainsi qu'améliorer la capacité du système à prédire la véracité des règles trouvées qui est à présent primitive. Ces pistes de perfectionnement sont indispensables à l'autonomie du système.

Annexes

Appendix A

Fonctions & Algorithmes

Sommaire

A.1 Fonctions de base

La structure de données utilisée suppose que chaque nœud du réseau possède une liste de ses relations sortantes, ainsi qu'une liste de ses relations entrantes. Il serait possible d'établir des fonctions ayant une complexité moindre, en ajoutant des propriétés aux nœuds (par exemple, la liste des voisins d'un nœud, classé par type de relation). Cela reste un problème correspondant au compromis espace/temps, et donc adaptable selon les contraintes posées pour une implémentation donnée.

Les fonctions de base considérées sont les suivantes :

- `Raffinements` (terme)=[raffinement$_1$,raffinement$_2$,...,raffinement$_n$] : Une fonction qui renvoie en tableau les raffinements si existants du terme donné. Complexité : $O(n)$;
- `Relations_Sortantes` (terme, type-de-relation, T[])=[T[e$_1$],T[e$_2$],...,T[e$_{n1}$]] : Une fonction qui renvoie en tableau les termes du tableau *T[]* qui vérifient la relation sortante de *terme* de type *type-de-relation* (*terme* $\xrightarrow{type-de-relation}$ *élément de T[]*). Complexité : $O(n)$;
- `Relations_Entrantes` (T[], type-de-relation, terme)=[T[e$_1$],T[e$_2$],...,T[e$_{n1}$]] : Une fonction qui renvoie en tableau les termes du tableau *T[]* qui vérifient la relation entrante vers *terme* de type *type-de-relation* (*élément de T[]* $\xrightarrow{type-de-relation}$ *terme*). Complexité : $O(n)$;
- `Poids` ($terme_1$ $\xrightarrow{type-de-relation}$ $terme_2$)=[Valeur] : Une fonction permettant d'obtenir le poids d'une relation dans le réseau. Complexité : $O(1)$;
- `Hyperonymes` (terme)=[h$_1$, h$_2$,..., h$_n$] : Une fonction permettant d'obtenir les hyperonymes h_n d'un *terme* donné. Complexité : $O(n)$;
- `Voisin` (terme)=[(R$_1$,V$_1$), (R$_2$,V$_2$),..., (R$_n$,V$_n$)] : Une fonction retournant les couples (Relation, Voisin) représentant toutes les relations sortantes d'un *terme* donné. En option on peut indiquer les types de relations demandées en utilisant la syntaxe `Voisin` (terme, type-rel$_1$, type-rel$_2$, ...). Complexité : $O(n)$;

- `Saisie` () : Une fonction récupérant la valeur saisie par l'utilisateur. Complexité : $O(1)$;

- `Inférer` ($terme_1 \overset{type-de-relation}{\longrightarrow} terme_2$) : Une fonction créant la relation en tant qu'inférence en attente de validation. Complexité : $O(1)$;

- `Liste_Relations_Sortantes` (terme | tableau de termes)=[rel_1, rel_2,..., rel_n] : Une fonction retournant un tableau avec toutes les relations sortantes de *terme* ou de *tableau de termes*. Complexité : $O(1)$;

- `Moyenne_Poids` (tableau de relations)=[Valeur] : Une fonction retournant la valeur correspondante à la moyenne des poids des relations de *tableau de relations*. Complexité : $O(n)$;

- `Trouver_Exemples_Similaires` (terme, ν_1, π_1)=[ex_1, ex_2,..., ex_n] : Une fonction renvoyant, s'ils existent, les exemples similaire à *terme*. Un *exemple* est similaire au *terme* s'il partage un minimum de ν_1 relations sortantes ayant un minimum de π_1 de poids avec le terme donné. Complexité : $O(r^2 \times n)$, où r correspond aux relations, et n aux nœuds;

- `Choisir_Candidats` (tableau_relations, ν_2, π_2)=[$relation\_candidate_1$, $relation\_candidate_2$,...,$relation\_candidate_n$] : Une fonction retournant un tableau avec les relations candidates à inférer. Ces candidates sont des éléments de *tableau_relations*, dont le 2-uplet ⟨type relation, nœud destination⟩ se répète au moins ν_2 fois et dont le poids est au minimum π_2. Complexité : $O(n^2)$;

- `Exist` (termX, termeY) : Une fonction qui retourne un booléen indiquant l'existence ou pas d'une quelconque relation sémantique dans n'importe quel sens entre le *termeX* et le *termeY*. Complexité : $O(n)$.

A.2 Déduction

(Chapitre 2. Section 2.2.1)

Fonction Filtrage-Logique-Deduction(termeA, termeB, termeC, R)

Fonction : Filtrage Logique Déduction
Entrées : termeA, termeB, termeC, R
Sorties : filtre
Complexité : $O(n^2)$
début
  filtre ← 0;
  tab_raff ← `Raffinements` (termeA);
  tab_vérif$_1$ ← `Relations_Sortantes` (termeA, is-a, tab_raff);
  tab_vérif$_2$ ← `Relations_Entrantes` (tab_raff, R, termeC);
  si *(tab _raff & tab_vérif*$_1$ *& tab_vérif*$_2$*)*$\neq \varnothing$ **alors**
    pour chaque V_1 *de tab_vérif*$_1$ *et* V_2 *de tab_vérif*$_2$ **faire**
      si $V_1 \neq V_2$ **alors**
        filtre ← 1;
  retourner *filtre*;

Fonction Filtrage-Statistique-Deduction(termeA, termeB, termeC, R)

Fonction : Filtrage Statistique Déduction
Entrées : termeA, termeB, termeC, R
Sorties : filtre
Complexité : $O(1)$
début
  filtre ← 0;
  poids_prémisse$_1$ ← `Poids` (*termeA* $\xrightarrow{is-a}$ *termeB*);
  poids_prémisse$_2$ ← `Poids` (*termeB* $\xrightarrow{R}$ *termeC*);
  poids_inférence ← (poids_prémisse$_1$ × poids_prémisse$_2$)$^{1/2}$;
  si *poids_inférence* < *seuil_filtrage* **alors**
    filtre ← 1;
  retourner *filtre*

Algorithme 1 : Algorithme de fonctionnement de l'inférence de relations par déduction

Algorithme : Déduction

Entrées : termeA
Complexité : $O(n^2)$
début
    seuil_filtrage ← `Saisie` ();
    tab_termeB ← `Hyperonymes` (termeA);
    pour chaque *termeB* ∈ *tab_termeB* **faire**
        tab_R_termeC ← `Voisin` (termeB);
        pour chaque <*R*,*termeC*> ∈ *tab_R_termeC* **faire**
            si ¬ `Filtrage-Logique-Deduction` **alors**
                si ¬ `Filtrage-Statistique-Deduction` **alors**
                    `Inférer` (*termeA* $\xrightarrow{R}$ *termeC*);

A.3 Induction

(Chapitre 2. Section 2.2.2)

Fonction Filtrage-Logique-Induction(termeA, termeB, termeC, R)

Fonction : Filtrage Logique Induction

Entrées : termeA, termeB, termeC, R
Sorties : filtre
Complexité : $O(n^2)$
début
  filtre ← 0;
  tab_raff ← `Raffinements` (termeA);
  tab_vérif$_1$ ← `Relations_Entrantes` (tab_raff, is-a, termeB);
  tab_vérif$_2$ ← `Relations_Entrantes` (tab_raff, R, termeC);
  si *(tab _raff & tab_vérif*$_1$ *& tab_vérif*$_2$*)*$\neq \varnothing$ **alors**
    pour chaque V_1 *de tab_vérif*$_1$ *et* V_2 *de tab_vérif*$_2$ **faire**
      si $V_1 \neq V_2$ **alors**
        filtre ← 1;
  retourner *filtre*;

Fonction Filtrage-Statistique-Induction(termeA, termeB, termeC, R)

Fonction : Filtrage Statistique Induction

Entrées : termeA, termeB, termeC, R
Sorties : filtre
Complexité : $O(1)$
début
  filtre ← 0;
  poids_prémisse$_1$ ← `Poids` $(termeA \xrightarrow{is-a} termeB)$;
  poids_prémisse$_2$ ← `Poids` $(termeA \xrightarrow{R} termeC)$;
  poids_inférence ← $\frac{poids\_prémisse_2^2}{poids\_prémisse_1}$;
  si *poids_inférence* < *seuil_filtrage* **alors**
    filtre ← 1;
  retourner *filtre*;

Algorithme 2 : Algorithme de fonctionnement de l'inférence de relations par induction

Algorithme : Induction

Entrées : termeA
Complexité : $O(n^2)$
début
  seuil_filtrage ← `Saisie` ();
  tab_termeB ← `Hyperonymes` (termeA);
  pour chaque *termeB* ∈ *tab_termeB* **faire**
    tab_R_termeC ← `Voisin` (termeA);
    pour chaque <*R,termeC*> ∈ *tab_R_termeC* **faire**
      si ¬ `Filtrage-Logique-Deduction` **alors**
        si ¬ `Filtrage-Statistique-Deduction` **alors**
          `Inférer` (*termeB* $\xrightarrow{R}$ *termeC*);

A.4 Abduction

(Chapitre 2. Section 2.3)

Algorithme 3 : Algorithme de fonctionnement de l'inférence de relations par abduction

Algorithme : Abduction

Entrées : termeA

Complexité : $O(r^2 \times n)$, où r correspond aux relations, et n aux nœuds.

début

- min_poids ← `Saisie` ();
- min_relations ← `Saisie` ();
- nbr_rel_sortantes ← `Taille` (`Liste_Relations_Sortantes` (termeA));
- moy_poids_rel_sortantes ← `Moyenne_Poids` (`Liste_Relations_Sortantes` (termeA));
- ν_1 ← `Max` (min_relations, nbr_rel_sortantes × 0.1);
- π_1 ← `Max` (min_poids, moy_poids_rel_sortantes × 0.5);
- tab_exemples ← `Trouver_Exemples_Similaires` (termeA, ν_1, π_1);
- nbr_exemples ← `Taille` (tab_exemples);
- moy_poids_rel_sortantes_ex ← `Moyenne_Poids` (`Liste_Relations_Sortantes` (tab_exemples));
- ν_2 ← `Max` (min_relations, nbr_exemples × 0.1);
- π_2 ← `Max` (min_poids, moy_poids_rel_sortantes_ex × 0.5);
- tab_inférences ← `Choisir_Candidats` (`Liste_Relations_Sortantes` (tab_exemples), ν_2, π_2);
- **pour chaque** *relation* ∈ *tab_inférences* **faire**
  - `Inférer` (*termeA* $\xrightarrow{type-de-relation}$ *terme-arrivée*);

A.5 SIR_R (Schéma d'Inférence de Relations avec Raffinements)

(Chapitre 2. Section 2.4)

Fonction Proposer-avec-B'(R, termeA', termeB')

Fonction : Proposer avec B'

Entrées : R, termeA', termeB'
Complexité : $O(n)$
début
  suivant *R* **faire**
    cas où *synonyme*
      tab_inférenceA' ← `Voisin` (termeB');
      tab_inférenceB' ← `Voisin` (termeA');
      pour chaque <R_T, $terme_T$> ∈ *tab_inférenceA'* **faire**
      `Inférer` (*termeA'* $\xrightarrow{R_T}$ $terme_T$);
      pour chaque <R_T, $terme_T$> ∈ *tab_inférenceB'* **faire**
      `Inférer` (*termeB'* $\xrightarrow{R_T}$ $terme_T$);
    cas où *hyperonyme*
      tab_inférenceB' ← `Voisin` (termeA');
      pour chaque <R_T, $terme_T$> ∈ *tab_inférenceB'* **faire**
      `Inférer` (*termeB'* $\xrightarrow{R_T}$ $terme_T$);
    autres cas
      `// le seul cas restant est l'hyponyme`
      tab_inférenceA' ← `Voisin` (termeB');
      pour chaque <R_T, $terme_T$> ∈ *tab_inférenceA'* **faire**
      `Inférer` (*termeA'* $\xrightarrow{R_T}$ $terme_T$);

Fonction Proposer-sans-B'(R, termeA', termeB')

Fonction : Proposer sans B'

Entrées : R, termeA', termeB
Complexité : $O(n)$
début
  suivant *R* **faire**
    cas où *hyponyme*
      tab_inférenceA' ← `Voisin` (termeB);
      pour chaque <R_T, $terme_T$> ∈ *tab_inférenceA'* **faire**
      `Inférer` (*termeA'* $\xrightarrow{R_T}$ $terme_T$);
    autres cas
      `// les cas restant sont l'hyperonyme et le synonyme qui seront traités pareil`
      tab_inférenceB ← `Voisin` (termeA');
      pour chaque <R_T, $terme_T$> ∈ *tab_inférenceB* **faire**
      `Inférer` (*termeB* $\xrightarrow{R_T}$ $terme_T$);

Algorithme 4 : Algorithme de fonctionnement du SIR_R (Schéma d'Inférence de Relations avec Raffinements)

Algorithme : SIR_R

Entrées : termeA

Complexité : $O(n^2)$

```
// les termes considérés ont au moins un
raffinement
```

début

- tab_termeA' ← `Raffinements` (termeA);
- **pour chaque** *termeA'* ∈ *tab_termeA'* **faire**
  - tab_R_termeB ← `Voisin` (termeA, hyperonyme, hyponyme, synonyme);
  - **pour chaque** <R_B,*termeB*> ∈ *tab_R_termeB* **faire**
    - **si** `Exist` *(termeA', termeB)* **alors**
      - tab_termeB' ← `Raffinements` (termeB);
      - **si** *tab_termeB'* ≠ ∅ **alors**
        - **pour chaque** *termeB'* ∈ *tab_termeB'* **faire**
          - **si** `Exist` *(termeA', termeB'* **alors**
            - `Proposer_avec_B'` (R_B, termeA', termeB');
          - **sinon**
            - `Proposer_sans_B'` (R_B, termeA', termeB);
      - **sinon**
        - `Proposer_sans_B'` (R_B, termeA', termeB);

Appendix B

La syntaxe des fonctions du MaLLeN

Sommaire

B.1 Create_term

create_term_cmd ::= **create_term** (label [**as** pos])+;
label ::= .. une chaine de caractère .. ;
pos ::= .. *(part of speech)*une chaine de caractère indiquant appartenant à la liste de *pos* définis dans le réseau (NOM_MAS_SING, NOM_FEM_SING, VERB, etc.) ..;

B.2 Create_relation

create_relation_cmd ::= **create_relation** relation_type [weight|annotation*] **between** label_i **and** label_j;
relation_type ::= ..une chaine de caractère indiquant le type de la relation souhaité figurant dans la liste de types prédéfinis (tableau D.1)..;
weight ::= ..un entier..;
annotation ::= ..une chaine de caractère..;

B.3 Rename_term

rename_term_cmd ::= **rename_term** label_i **as** label_j;

B.4 Annotate_relation

annotate_relation_cmd ::= **annotate_relation** relation_type **between** label_i **and** label_j **as** annotation^+;

B.5 Delete_term

delete_term_cmd ::= **delete_term** label^{+};

B.6 Delete_relation

delete_relation_cmd ::= **delete_relation** relation_type|all **from** label$_i$ [**to** label$^{+}_j$];

B.7 Delete_annotation

delete_annotation_cmd ::= **delete_annotation** annotation^{+}|all **of** relation_type **between** label$_i$ **and** label$_j$;

B.8 Copy_relation

copy_relation_cmd ::= **copy_relation** out|in relation_type|all **from** label$_i$ **to** label$^{+}_j$;

B.9 Exchange_relation

exchange_relation_cmd ::= **exchange_relation** out|in relation_type|all **between** label$_i$ **and** label$^{+}_j$;

B.10 Reason_by_deduction

reason_by_deduction_cmd ::= **Reason_by_deduction** relation_type|all **start** label$_i$ [**end** label$_j$];

B.11 Reason_by_induction

reason_by_induction_cmd ::= **Reason_by_induction** relation_type|all **start** label$_i$ [**end** label$_j$];

B.12 Reason_by_abduction

reason_by_induction_cmd ::= **Reason_by_abduction** all **to** label ;

B.13 Search_rules

search_rules_cmd ::= **search_rules** weight | weight$_{min}$ weight$_{max}$ **from** label ;

B.14 Give

give_cmd ::= **give** $variable | $variable$_x$ **and** $variable$_y$ **where** prémisse | prémisse$_i$ (**and** prémisse$_j$)$^+$;
$variable ::= ..une chaine de caractères indiquant le terme inconnu qu'on cherche à connaitre..;
prémisse ::= ..une chaine de caractères sous forme de relation *terme1 type-relation terme2* ..;

Appendix C

Glossaire

GWAP	Game With A Purpose
RLS	Réseau(x) Lexico-Sémantique(s)
RS	Relation(s) Sémantique(s)
LD	Langage Dédié
LG	Langage Généraliste
JDM	JeuxDeMots
GRI	Générateur de Règles d'Inférence
SCEREL	Système de Consolidation Endogène de Réseaux Lexicaux
MaLLeN	Manipulating Language for Lexical Networks (langage de manipulation pour les réseaux lexicaux)

Appendix D

Types de relation et parties du discours

Tableau D.1 – Index de types de relations utilisés tout au long du mémoire, leur correspondance dans le réseau d'origine et leur description selon le site de ce dernier
http://www.jeuxdemots.org/jdm-about-detail-relations.php

Nom utilisé	**Nom dans le réseau**	**Description**
is-a (hyper)	r_isa	Il est demandé d'énumérer les GÉNÉRIQUES (hyperonymes) du terme. Par exemple, 'animal' et 'mammifère' sont des génériques de 'chat'.
assoc	r_associated	Il est demandé d'énumérer tout terme lié d'une façon ou d'une autre au mot cible... Ce mot vous fait penser à quoi ?
hypo	r_hypo	Il est demandé d'énumérer des SPÉCIFIQUES/hyponymes du terme. Par exemple, 'mouche', 'abeille', 'guêpe' pour 'insecte'.

has-part	r_has_part	Il faut donner des PARTIES/constituants/éléments (a pour méronymes) du mot cible. Par exemple, 'voiture' a comme parties : 'porte', 'roue', 'moteur', ...
holo	r_holo	Le 'TOUT' de l'objet en question. Pour 'main', on aura 'bras', 'corps', 'personne', etc... Le tout est aussi l'ensemble comme 'classe' pour 'élève'.
location	r_lieu	Il est demandé d'énumérer les LIEUX typiques où peut se trouver le terme/objet en question.
charac	r_carac	Pour un terme donné, souvent un objet, il est demandé d'en énumérer les CARACtéristiques (adjectifs) possibles/typiques. Par exemple, 'liquide', 'froide', 'chaude', pour 'eau'.
agent-1	r_agent-1	Que peut faire ce SUJET ? (par exemple chat => miauler, griffer, etc.)
instr-1	r_instr-1	L'instrument est l'objet avec lequel on fait l'action. Dans - Il mange sa salade avec une fourchette -, fourchette est l'instrument. On demande ici, ce qu'on peut faire avec un instrument donné...
patient-1	r_patient-1	L'instrument est l'objet avec lequel on fait l'action. Dans - Il mange sa salade avec une fourchette -, fourchette est l'instrument. On demande ici, ce qu'on peut faire avec un instrument donné...
location-1	r_lieu-1	A partir d'un lieu, il est demandé d'énumérer ce qui peut typiquement s'y trouver.
location-action	r_lieu_action	A partir d'un lieu, énumérer les action typiques possibles dans ce lieu.
object-mater	r_object>mater	Quel est la ou les MATIÈRE/SUBSTANCE pouvant composer l'objet qui suit. Par exemple, 'bois' pour 'poutre'.
refinement	r_raff_sem	Raffinement sémantique vers un usage particulier du terme source.
target	r_cible	Cible de la maladie : myxomatose => lapin, rougeole => enfant, ...

consequence	r_conseq	B est une CONSÉQUENCE possible de A. A et B sont des verbes ou des noms. Exemples : tomber -> se blesser ; faim -> voler ; allumer -> incendie ; négligence –> accident ; etc.
sentiment	r_sentiment	Pour un terme donné, évoquer des mots liés à des SENTIMENTS ou des ÉMOTIONS pouvant être associer à ce terme. Par exemple, la joie, le plaisir, le dégoût, la peur, la haine, l'amour, l'indifférence, l'envie, avoir peur, horrible, etc.

• Abr: • Adj: • Adj:Card • Adj:Fem+InvPL
• Adj:Fem+PL • Adj:Fem+SG • Adj:Fem+SG:InvGen+PL
• Adj:InvGen+InvPL • Adj:InvGen+PL • Adj:InvGen+SG
• Adj:InvGen+SG:InvGen+PL • Adj:Mas+InvPL • Adj:Mas+PL
• Adj:Mas+SG • Adj:Pos • Adv: • Card: • Chunk: • Con:
• Concept: • Conj: • Date: • Det: • Det:Fem+PL
• Det:Fem+SG • Det:InvGen+PL • Det:InvGen+SG
• Det:InvGen+SG:InvGen+PL • Det:Mas+PL
• Det:Mas+SG • Expression: • Fem+SG+DemPro+Prox
• Info: • Int: • InvGen+PL+P1+PProRefl
• Mas+SG+DemPro+Prox • Mas+SGDemPro2 • Modifier:
• NNom+Fem+SG+P3+PProRefl • NNom+InvGen+SG+P1+PProRefl
• NNom+InvGen+SG+P3+PProRefl • NNom+Mas+SG+P3+PProRefl
• Nom+InvGen+SG+P3+PC • Nom: • Nom:Fem+InvPL
• Nom:Fem+PL • Nom:Fem+SG • Nom:Fem+SG+Nom
• Nom:InvGen+InvPL • Nom:InvGen+PL • Nom:InvGen+SG
• Nom:InvGen+SG:InvGen+PL • Nom:Mas+InvPL
• Nom:Mas+PL • Nom:Mas+SG • Nom:Mas+SG+Nom
• Nom:SG • Nom:SG+Card • Ono: • Pre: • Prefix:
• Pro: • Pro:3Fem+PL • Pro:3Fem+SG • Pro:3Mas+PL
• Pro:3Mas+SG • Pro:Fem+PL • Pro:Fem+SG
• Pro:InvGen+PL • Pro:InvGen+SG • Pro:Mas+PL
• Pro:Mas+SG • Pro:PL+P1 • Pro:PL+P2 • Pro:PL+P3
• Pro:SG+P1 • Pro:SG+P2 • Pro:SG+P3 • Pro:SG+P3:PL+P3
• Pro:SG+P3PL+P3 • Pro: • Relation: • Suffix:
• Symbole: • Ukn: • Ver: • Ver:CPre+SG+P1:CPre+SG+P2
• Ver:CPre+SG+P3 • Ver:IFut+PL+P1 • Ver:IFut+SG+P1
• Ver:IFut+SG+P2 • Ver:IFut+SG+P3
• Ver:IImp+PL+P1 • Ver:IImp+PL+P1:SPre+PL+P1
• Ver:IImp+PL+P2:SPre+PL+P2 • Ver:IImp+PL+P3
• Ver:IImp+SG+P1:IImp+SG+P2 • Ver:IImp+SG+P3
• Ver:IPSim+SG+P1 • Ver:IPSim+SG+P1:IPSim+SG+P2
• Ver:IPSim+SG+P2 • Ver:IPSim+SG+P3 • Ver:IPre+PL+P1
• Ver:IPre+PL+P1:ImPre+PL+P1 • Ver:IPre+PL+P2
• Ver:IPre+PL+P2:ImPre+PL+P2 • Ver:IPre+PL+P3
• Ver:IPre+PL+P3:SPre+PL+P3 • Ver:IPre+SG+P1
• Ver:IPre+SG+P1:IPre+SG+P2

- Ver:IPre+SG+P1:IPre+SG+P2:IPSim+SG+P1:IPSim+SG+P2:...
- Ver:IPre+SG+P1:IPre+SG+P2:ImPre+SG+P2
- Ver:IPre+SG+P1:IPre+SG+P3:SPre+SG+P1:SPre+SG+P3:Im...
- Ver:IPre+SG+P2
- Ver:IPre+SG+P2:SPre+SG+P2 • Ver:IPre+SG+P3
- Ver:IPre+SG+P3:IPSim+SG+P3 • Ver:IPre+SG+P3:ImPre+SG+P2
- Ver:IPre+SG+P3:SPre+SG+P3 • Ver:Imp+PL+P1
- Ver:Imp+PL+P2 • Ver:Imp+SG+P2 • Ver:Inf
- Ver:Inf+:IPre+SG+P1:IPre+SG+P3:SPre+SG+P1:SPre+SG+...
- Ver:PPas
- Ver:PPas+Fem+PL
- Ver:PPas+Fem+PL:IPre+PL+P2:ImPre+PL+P2
- Ver:PPas+Fem+SG
- Ver:PPas+Fem+SG:SPre+SG+P1:SPre+SG+P3
- Ver:PPas+Mas+PL
- Ver:PPas+Mas+PL:IPSim+SG+P1:IPSim+SG+P2
- Ver:PPas+Mas+PL:IPre+SG+P1:IPre+SG+P2:IPSim+SG+P1:...
- Ver:PPas+Mas+SG
- Ver:PPas+Mas+SG:IPSim+SG+P1:IPSim+SG+P2
- Ver:PPas+Mas+SG:IPre+SG+P3
- Ver:PPas+Mas+SG:PPas+Mas+PL
- Ver:PPas+Mas+SG:PPas+Mas+PL:IPSim+SG+P1:IPSim+SG+P...
- Ver:PPas+Mas+SG:PPas+Mas+PL:IPre+SG+P1:IPre+SG+P2:...
- Ver:PPre • Ver:PPre+Fem+SG • Ver:PPre+Mas+SG
- Ver:SImp+PL+P1 • Ver:SImp+SG+P1
- Ver:SImp+SG+P2 • Ver:SImp+SG+P3 • Ver:SPre+PL+P3
- Ver:SPre+SG+P1 • Ver:SPre+SG+P1:ImPre+SG+P2
- Ver:SPre+SG+P1:SPre+SG+P3
- Ver:SPre+SG+P1:SPre+SG+P3:SImp+SG+P1
- Ver:SPre+SG+P2 • Ver:SPre+SG+P3

Figure D.1 – Liste des parties du discours (pos) utilisables dans le langage

Références bibliographiques

Adda, G., Mariani, J., 2010. *Language resources and amazon mechanical turk : legal, ethical and other issues.* In: LISLR2010, "Legal Issues for Sharing Language Resources workshop", LREC2010.

Ashburner, M., Ball, C., Blake, J., Botstein, D., Butler, H., Cherry, M., Davis, A., Dolinski, K., Dwight, S., Eppig, J., Harris, M., Hill, D., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J., Richardson, J., Ringwald, M., Rubin, G., Sherlock, G., 2000. *Gene Ontology: tool for the unification of biology*. Nature Genetics 25, pp.25–29.

Backus, J., 1959. *The syntax and semantics of the proposed international algebraic language of the Zurich ACM-GAMM Conference*. In: Proceedings of the International Conference on Information Processing. UNESCO. pp. 125–132.

Baker, C. F., Fillmore, C. J., Lowe, J. B., 1998. *The Berkeley FrameNet Project*. Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics.

Besnard, P., Cordier, M., Moinard, Y., 2008. *Ontology-based inference for causal explanation.* Integrated Computer-Aided Engineering , IOS Press, Amsterdam , Vol. 15 , No. 4., pp.351–367.

Blanco, E., Cankaya, H., Moldovan, D., 2011. *Commonsense Knowledge Extraction Using Concepts Properties*. In: FLAIRS Conference'11.

Bouchard, A., 1878. *La langue théâtrale : vocabulaire historique, descriptif et anecdotique des termes et des choses du théâtre, suivi d'un appendice contenant la législation théâtrale en vigueur.* Arnaud et Labat.

Cassidy, P. J., 2002. *GNU Collaborative International Dictionary of English*. In: http://gcide.gnu.org.ua/.

Ceusters, W., Smith, B., 2010. *Foundations for a realist ontology of mental disease*. Journal of Biomedical Semantics 1 (1).

Chowdhury, G. G., 2003. *Natural language processing*. Annual Review of Information Science and Technology 37 (1), pp.51–89.

Collins, A., M.R., Q., 1969. *Retrieval time from semantic memory.* Journal of verbal learning and verbal behaviour, pp.240–248.

Colmerauer, A., 1970. *Les Systèmes Q ou un Formalisme pour Analyser et Synthétiser des Phrases sur Ordinateur*. In: (Internal Publication No. 43).

Colmerauer, A., Kanoui, H., Roussel, P., Pasero, R., 1973. *Un système de communication homme-machine en Français*. rapport de recherche, Groupe de recherche en Intelligence Artificielle.

Consel, C., Latry, F., Réveillère, L., Cointe, P., 2005. *A Generative Programming Approach to Developing DSL Compilers*. In: Glück, R., Lowry, M. (Eds.), Generative Programming and Component Engineering. Vol. 3676 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 29–46.

Crowston, K., 2012. *Amazon Mechanical Turk: A Research Tool for Organizations and Information Systems Scholars*. In: Bhattacherjee, A., Fitzgerald, B. (Eds.), Shaping the Future of ICT Research. Vol. 389. pp. 210–221.

d'Aquin, M., 2012. *Modularizing Ontologies*. Springer Berlin Heidelberg.

Deursen, A. V., Klint, P., Visser, J., 2000. *Domain-Specific Languages: An Annotated Bibliography*. ACM SIGPLAN NOTICES 35, pp.26–36.

Doerr, M., 2003. *The CIDOC CRM – An Ontological Approach to Semantic Interoperability of Metadata*. AI Magazine, pp.75–92.

Dong, Z., Dong, Q., 2006. *Hownet And the Computation of Meaning* . World Scientific Publishing Co., Inc. River Edge, NJ, USA.

Dong, Z., Dong, Q., Hao, C., 2010. *HowNet and Its Computation of Meaning*. Demonstrations Volume COLING (Demos), pp.53–56.

Dubois, J., Giacomo, M., Guespin, L., Marcellesi, C., Marcellesi, J., Mevel, J., 1973. *Dictionnaire de linguistique*. Larousse.

Fillmore, C. J., 1976. *Frame semantics and the nature of language*. Annals of the New York Academy of Sciences 280 (1), pp.20–32.

Fowler, M., 2010. *Domain Specific Languages*, 1st Edition. Addison-Wesley Professional.

Fox, M., 1992. *The TOVE Project: Towards A Common-sense Model of the Enterprise*. Enterprise Integration Laboratory Technical Report, 16p.

Gaines, B. R., 1993. *Representation, discourse, logic and truth: situating knowledge technology*. In Mineau, G., Moulin, W. and Sowa, J. F. (eds), Proc. 1st Int. Conf. on Conceptual Structures, ICCS'93 (Lecture Notes in Artificial Intelligence, vol. 699), Quebec, Canada, pp.36–63.

Gala, N., Zock, M., 2013. *Ressources lexicales. Contenu, construction, utilisation, évaluation*. John Amsterdam/Philadelphia, Benjamins Publishing Company.

Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L., 2002. *Sweetening Ontologies with DOLCE*. EKAW, 16p.

Grezka, A., Cartier, E., Mathieu-Colas, M., 2015. *Dictionnaires morphologiques du français contemporain : présentation de Morfetik, éléments d'un modèle pour le TAL*. TALN.

Group, X. R., 2006. *A lexicalized tree adjoining grammar for English*. Technical Report IRCS-01-03, IRCS, University of Pennsylvania.

Gruber, T., 1993. *A translation approach to portable Ontology specifications*. Knowledge acquisition, pp.199–220.

Guarino, N., (1998). *Formal ontology in information systems*. Proceedings of FOIS 1998, Trento, Italy, pp.3–15.

Harabagiu, S., 1911. *Roget's International Thesaurus*. In: 1st edition, Cromwell, New York, USA.

Harabagiu, S., 1998. *WordNet-Based Inference of Textual Cohesion and Coherence*. In: FLAIRS-98. pp. 265–269.

Havasi, C., Speer, R., Alonso, J., 2009. *ConceptNet: A lexical resource for common sense knowledge.* Recent Advances in Natural Language Processing,.

Hudak, P., Dec. 1996. *Building Domain-specific Embedded Languages*. ACM Comput. Surv. 28 (4es).

Hudak, P., 1997. *Domain-specific languages*. Handbook of Programming Languages 3, pp.39–60.

Hudak, P., 1998. *Modular domain specific languages and tools*. In: Software Reuse, 1998. Proceedings. Fifth International Conference on. IEEE, pp. 134–142.

Hutchings, B. L., Nelson, B. E., 2000. *Using general-purpose programming languages for FPGA design*. In: DAC '00: Proceedings of the 37th conference on Design automation. ACM, pp. 561–566.

Karin, K., Anna, K., Neville, R., Martha, P., 2006. *Extending VerbNet with Novel Verb Classes*. Proceedings of the Fifth International Conference on Language Resources and Evaluation – LREC'06.

Karin, K., Palmer, M., Rambow, O., 2002. *Extending PropBank with VerbNet semantic predicates*. Workshop on Applied Interlinguas, AMTA-2002, Tiburon, California.

Lafourcade, M., 1994. *Génie logiciel pour le génie linguiciel*. IMAG-UJF, Grenoble 1.

Lafourcade, M., 2007. *Making people play for Lexical Acquisition.* In Proc. SNLP 2007, 7th Symposium on Natural Language Processing. Pattaya, Thailande, 13-15 December 2007, 8 p.

Lafourcade, M., Joubert, A., 2009. *Similitude entre les sens d'usage d'un terme dans un réseau lexical.* In: Traitement Automatique des Langues (TAL). pp. 179–200.

Lafourcade, M., Zarrouk, M., Joubert, A., 2013. *Inférence de règles déductives par abduction.* In proc of Méthodes mixtes pour l'analyse syntaxique et sémantique du français (Mixer), Atelier TALN'2013, 4p.

Lauser, B., Sini, M., 2006. *From AGROVOC to the Agricultural Ontology Service/Concept Server: An OWL Model for Creating Ontologies in the Agricultural Domain*. In: Proceedings of the 2006 International Conference on Dublin Core and Metadata Applications: Metadata for Knowledge and Learning. Dublin Core Metadata Initiative, pp. 76–88.

Lebraty, J., 2007. *Vers un nouveau mode d'externalisation : le crowdsourcing.* 12éme conférence de l'AIM, 20 p.

Leclère, C., Laporte, E., Piot, M., Silberztein, M. (Eds.), 2004. *Lexique, syntaxe et lexique-grammaire. Syntax, Lexis and Lexicon-Grammar. Papers in honour of Maurice Gross*.

Lefeuvre, A., Antoine, J., Savary, A., Schang, E., Abouda, L., Maurel, D., Eshkol, I., 1998. *Les ressources lexicales pour l'étiquetage sémantique*. In: Les linguistiques de corpus. p. 24.

Lefeuvre, A., Antoine, J., Savary, A., Schang, E., Abouda, L., Maurel, D., Eshkol, I., 2014. *Annotation de la temporalité en corpus : contribution à l'amélioration de la norme TimeML.* In: Traitement Automatique des Langues Naturelles. pp. 1–4.

Lenat, D., 1995. *CYC: A Large-Scale Investment in Knowledge Infrastructure*. Communications of the ACM 38, pp.33–38.

Levin, B., 1993. *English verb classes and alternations : a preliminary investigation*. University of Chicago Press.

Liu, H., Singh, P., 2004. *ConceptNet: A Practical Commonsense Reasoning Toolkit*. BT Technology Journal 22.

Mahesh, K., 1996. *Ontology Development for Machine Translation: Ideology and methodology.* Rl report mccs - New Mexico State University, pp.96–292.

Mangeot, M., Sérasset, G., Lafourcade, M., 2003. *Construction collaborative d'une base lexicale multilingue, le projet Papillon.* In: Traitement Automatique des Langues Naturelles. pp. 151–176.

Marchetti, A., Tesconi, M., Ronzano, F., Mosella, M., Minutoli, S., 2007. *SemKey: A Semantic Collaborative Tagging System.* in Procs of WWW2007, Banff, Canada, 9 p.

Mernik, M., Heering, J., Sloane, A. M., 2005. *When and How to Develop Domain-specific Languages*. ACM Comput. Surv. 37 (4), pp.316–344.

Mihalcea, R., Chklovski, T., 2003. *Open MindWord Expert: Creating large annotated data collections with web users help.* In Proceedings of the EACL 2003, Workshop on Linguistically Annotated Corpora (LINC 2003), 10 p.

Miller, G. A., 1985. *Dictionaries of the Mind*. Language Resources and Evaluation, pp.305–314.

Miller, G. A., 1995. *WordNet: A Lexical Database for English.* ACM 38 (11), pp.39–41.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K. J., 1990. *Introduction to WordNet: an on-line lexical database*. International Journal of Lexicography 3 (4), pp.235–244.

Miller, G. A., Fellbaum, C., 2007. *WordNet then and now. Language Resources and Evaluation* 41 (2), pp.209–214.

Minsky, M., 1975. *A framework for representing knowledge*. The psychology of computer vision. New York, pp.211–277.

Nakamura, J.-i., Tsuji, J.-i., Nagao, M., 1988. *Grade: A Software Environment for Machine Translation*. Comput. Transl. 3 (1), pp.69–82.

Navigli, R., Ponzetto, S. P., 2012. *BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network*. Artificial Intelligence 193, pp.217–250.

New, B., Pallier, C., Ferrand, L., Matos, R., 2001. *Une base de données lexicales du français contemporain sur internet: LEXIQUE*. L'Année Psychologique, pp.447–462.

Niles, I., Pease, A., 2001. *Towards a Standard Upper Ontology*. In Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001).

Nirenburg, S., 1989. *Knowledge-based machine translation*. Machine Translation 4 (1), pp.5–24.

Oliveira, N., ao Varanda Pereira, M. J., Henriques, P. R., da Cruz, D., 2009. *Domain-Specific Languages - A Theoretical Survey*. In: Proceedings of the 3rd Compilers, Programming Languages, Related Technologies and Applications (CoRTA'2009). pp. 35–46.

Pease, A., Niles, I., 2002. *IEEE Standard Upper Ontology: A Progress Report*. Knowledge Engineering Review, Special Issue on Ontologies and Agents, pp.65–70.

Peirce, C. S., 1931. *Collected Papers*. Harvard University Press.

Pianta, E., Bentivogli, L., Girardi, C., 2002. *MultiWordNet: developing an aligned multilingual database*. In Proceedings of the First International Conference on Global WordNet, 10p.

Polguère, A., 2008. *Structural properties of Lexical Systems : Monolingual and Multilingual Perspectives.* In: Proceedings of the Workshop on Multilingual Language Resources and Interoperability (Coling/ACL). pp. 50–59.

Quillian, M., 1968. *Semantic Information Processing*. Semantic Memory, in M. Minsky (ed.),, pp.227–270.

Ramadier, L., Zarrouk, M., Lafourcade, M., Micheau, A., 2014a. *Annotations et inférences de relations dans un réseau lexico-sémantique: application à la radiologie*. In: Traitement Automatique des Langues Naturelles. p. 12.

Ramadier, L., Zarrouk, M., Lafourcade, M., Micheau, A., 2014b. *Inferring Relations and Annotations in Semantic Networks - Application to Radiology.* In: Journal Computación y Sistemas.Vol 18, No 2.

Ramadier, L., Zarrouk, M., Lafourcade, M., Micheau, A., 2014c. *Spreading Relation Annotations in a Lexical Semantic Network Applied to Radiology.* In: In proc of 15th International Conference on Intelligent Text Processing and Computational Linguistics (CICLING 2014). pp. 6–12.

Rich, E., Knight, K., 1999. *Artificial Intelligence (2nd Edition)*. Prentice Hall.

Richardson, S., Dolan, W., Vanderwende, L., 1998. *MindNet: acquiring and structuring semantic information from text*. COLING 1998 Volume 2: The 17th International Conference on Computational Linguistics.

Richardson, S. D., 1997. *Determining Similarity and Inferring Relations in a Lexical Knowledge Base*. TechReport.

Sagot, B., Fiser, D., 2008. *Construction d'un wordnet libre du français à partir de ressources multilingues*. In: Traitement Automatique des Langues Naturelles. p. 10.

Scowen, R. S., 1993. *Extended BNF — A generic base standard*. In: Software Engineering Standards Symposium.

Singh, P., Lin, T., Mueller, E. T., Lim, G., Perkins, T., Zhu, W. L., 2002. *Open Mind Common Sense: Knowledge Acquisition from the General Public*. In: On the Move to Meaningful Internet Systems, 2002 - DOA/CoopIS/ODBASE 2002 Confederated International Conferences DOA, CoopIS and ODBASE 2002. Springer-Verlag, pp. 1223–1237.

Snow, R., Jurafsky, D., Ng, A., 2006. *Semantic taxonomy induction from heterogenous evidence.* in Proceedings of COLING/ACL 2006., 8 p.

Sowa, J. F., 1984. *Conceptual structures : information processing in mind and machine*. The systems programming series. Addison-Wesley, Reading Ma.

Sowa, J. F., 1988. *Knowledge Representation in Databases, Expert Systems, and Natural Language*. DS-3'88, pp.17–50.

Sowa, J. F., 1992. *Semantic Networks*. Encyclopedia of Artificial Intelligence.

Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S., Tsujii, J., 2012. *BRAT: A Web-based Tool for NLP-assisted Text Annotation*. In: Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics. EACL '12. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 102–107.

Syed, Z., Viegas, E., Parastatidis, S., 2010. *Automatic Discovery of Semantic Relations using MindNet*. LREC'10.

Thaler, S., Siorpaes, K., Simperl, E., Hofer, C., 2011. *A Survey on Games for Knowledge Acquisition.* In: TI Technical Report. p. 9.

Uchida, H., Zhu, M., Senta, D., 2005. *The Universal Networking Language*. In: 2nd ed. UNDL Foundation.

Vauquois, B., Veillon, G., Veyrunes, J., 1967. *Un Metalangage de grammaires Transformationnelles Applications, aux problemes de transferts et de Generation syntaxiques*. In: Second Conference Internationale Sur Le Traitement Automatique Des Langues, COLING 1967, Grenoble, France, August 1967.

Velardi, P., Navigli, R., Cucchiarelli, A., Neri, F., 2006. *Evaluation of OntoLearn, a methodology for Automatic Learning of Ontologies.* in Ontology Learning and Population, Paul Buitelaar Philipp Cimmiano and Bernardo Magnini Editors, IOS press.

Velardi, P., Navigli, R., D'Amadio, P., 2008. *Mining the Web to Create Specialized Glossaries*. IEEE Intelligent Systems, vol.23, no. 5, pp.18–25.

Visser, E., 2008. *WebDSL: A Case Study in Domain-Specific Language Engineering*. Generative and Transformational Techniques in Software Engineering II 5235, pp.291–373.

von Ahn, L., Dabbish, L., 2008. *Designing games with a purpose.* Communications of the ACM 51 (8), pp.58–67.

von Ahn, L., Kedia, M., Blum, M., 2006. *Verbosity: A Game for Collecting Common-sense Facts*. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, New York, NY, USA, pp. 75–78.

Vossen, P. (Ed.), 1998. *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer Academic Publishers, Norwell, MA, USA.

Watt, D. A., 1990. *Programming Language Concepts and Paradigms*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Woods, W. A., 1970. *Transition Network Grammars for Natural Language Analysis*. Commun. ACM 13 (10), pp.591–606.

Yngve, V. H., 1958. *A programming language for mechanical translation*. In: Mechanical Translation. pp. 25–41.

Yvon, F., 2010. *Une petite introduction au Traitement Automatique des Langues Naturelles.* In: Conference on Knowledge discovery and data mining. ACM, pp. 27–36.

Zarrouk, M., Lafourcade, M., 2014a. *Inferring Knowledge with Word Refinements in a Crowdsourced Lexical-Semantic Network.* In: In proc of the the 25th International Conference on Computational Linguistics (COLING 2014), Dublin, Irlande. p. 9.

Zarrouk, M., Lafourcade, M., 2014b. *Relation Inference in Lexical Networks... with Refinements.* In: In proc of the 9th International Conference on Language Resources and Evaluation (LREC), Reykjavik, iceland. p. 6.

Zarrouk, M., Lafourcade, M., Joubert, A., 2013a. *Inductive and deductive inferences in a Crowdsourced Lexical-Semantic Network.* In: In proc of 9th International Conference on Recent Advances in Natural Language Processing (RANLP 2013). p. 6.

Zarrouk, M., Lafourcade, M., Joubert, A., 2013b. *Inference and Reconciliation in a Crowdsourced Lexical-Semantic Network.* In: In proc

of 14th International Conference on Intelligent Text Processing and Computational Linguistic (CICLING-2013). p. 13.

Zarrouk, M., Lafourcade, M., Joubert, A., 2013c. *Inférences déductives et réconciliation dans un réseau lexico-sémantique* . In: Traitement Automatique des Langues Naturelles. p. 14.

Zarrouk, M., Lafourcade, M., Joubert, A., 2014a. *About Inferences in a Crowdsourced Lexical-Semantic Network.* In: In proc of 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014). pp. 26–30.

Zarrouk, M., Lafourcade, M., Joubert, A., 2014b. *Consolidation de réseaux lexico-sémantiques par des inférences déductives et inductives.* In: In proc of journées internationales d'Analyse statistique des Données Textuelles (JADT). pp. 2–6.