UNIVERSITÉ DU QUÉBEC À MONTRÉAL

PROBLÈME DE L'ANCRAGE SYMBOLIQUE ET REPRÉSENTATION ABSTRAITE DU SENS

MÉMOIRE

PRÉSENTÉ

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INFORMATIQUE

PAR

NICOLAS GOULET

MAI 2025

REMERCIEMENTS

Nous ne sommes rien sans les autres, nous ne sommes que notre environnement. Tout mérite apparent n'est, en fait, que la conséquence de mes merveilleux collaborateurs et collaboratrices qui m'ont accompagné et qui ont fait de moi la personne que je suis devenue au cours de ces dernières années.

Premièrement, ma famille, sans le soutien de laquelle cette longue pente pour faire avancer les frontières des connaissances n'aurait jamais pu être entreprise. Mes grands-parents, Denis Goulet et Odette Baribeau, et mes parents, Muriel Dejean et Jean-François Goulet, ont sacrifié leurs aspirations personnelles afin que nous n'ayons pas à le faire. *Sacrifice oblige* !

Ensuite, mes amis et collègues. Dans ces 6 dernières années (incluant mes 4 années au baccalauréat), j'ai eu la chance d'être entouré des personnes les plus merveilleuses et éclectiques qui soient. Mes amis qui liront ces remerciements sauront se reconnaître mais je ne pourrais taire les noms de quelques mentors. Lucas House, à qui je dois la première invitation à une réunion de laboratoire en été 2017, marquant ainsi le début du chemin qui m'a mené jusqu'ici. Fernanda Pérez-Gay, qui fut toujours un exemple d'excellence auquel je devais aspirer pour approcher mes questions de recherche. Christian Thériault, dont la profonde intelligence et l'esprit en perpétuelle ébullition m'ont fait sentir moins seul. Odile Marcotte, sans qui beaucoup n'eut été possible. Toutes ces personnes et plusieurs autres que je garde sous silence par souci d'espace *font qui je suis aujourd'hui* et pour cela, je leur dois une gratitude éternelle.

Finalement, mes plus sincères remerciements à mes superviseurs Alexandre Blondin Massé et Étienne Harnad. *L'imitation est innée en nous, ce qu'il faut imiter n'est point aisé à reconnaître* ; j'ai derechef reconnu deux archétypes auxquels je devais dévouer les années nécessaires pour apprendre à les imiter. Merci pour votre confiance en moi et votre engagement intellectuel.

DÉDICACE

Je dédie cette maîtrise à tous les gens que j'ai connu et qui n'ont pu mener à terme le développement de leur esprit. Dans cette tour d'ivoire qu'est le monde académique, bien que nous souhaitons faire du monde une meilleure place, une quantité indénombrable d'individualités, de *plus-value* apportée par l'originalité, est en tout temps perdu dans l'indigence sociale profonde qui sévit au Québec. Pour toutes ces âmes que j'ai vues se perdre dans l'indifférence la plus complète, mon privilège est une responsabilité.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

ACRONYMES

UQAM Université du Québec à Montréal.

AMR *Abstract Meaning Representation*.

BMR *BabelNet Meaning Representation*.

TCCM transversal de circuits de cardinalité minimale.

CSM couverture par nœuds minimale.

CFC composante fortement connexe.

TALN traitement automatique du langage naturel.

RAS Représentation abstraite du sens.

DSM Désambiguïsation du sens des mots.

WN Wordnet.

TLFI Trésor de la langue française informatisé.

WEDT Wordsmyth Educational Dictionary-Thesaurus.

WLDT Wordsmyth Learner's Dictionary-Thesaurus.

WCDT Wordsmyth Children's Dictionary-Thesaurus.

WILD Wordsmyth Illustrated Learner's Dictionary.

StoG sentence-to-graph.

RÉSUMÉ

Lorsque nous lisons ou entendons un mot dont le sens nous échappe, nous pouvons consulter sa définition dans un dictionnaire. Si dans la définition, nous lisons un autre mot dont le sens nous échappe, nous pouvons chercher à son tour sa définition, répétant autant de fois que nécessaire le processus. Or, une langue ne peut être apprise par la recherche de définitions dans un dictionnaire seulement : *le sens de certains mots doit être acquis au préalable*. Expliquer comment et pourquoi les humains ont la capacité unique *d'ancrer dans des symboles arbitraires les choses auxquelles ils réfèrent* afin de briser cette régression infinie est connu sous le nom du problème de l'ancrage des symboles. Pour identifier les ensembles d'ancrage minimaux (soit les plus petits ensembles de mots devant être *ancrés* au préalable pour pouvoir apprendre tous les autres), des dictionnaires anglais et français ont été modélisés comme des graphes et ont ensuite été réduits de façon confluente, soit d'une façon qui produit des résultats uniques, afin d'en étudier les structures. De plus, les dictionnaires anglais sont plongés en représentation abstraite du sens, un formalisme sémantique d'intérêt pour la modélisation cognitive.

ABSTRACT

When we encouter a word whose meaning we don't know, we look up its definition. If, in the definition, we find another word whose meaning we don't know, then we can look that up too, repeating this process until it eventually stops, with all meanings known. However, a language cannot be fully learned from a dictionary via definition look-up alone : *the meaning of some words has to be acquired beforehand*. Explaining how and why humans have the unique capacity to *ground in arbitrary symbols the things in the world they refer to* in order to break this circularity is known as the symbol grounding problem. To identify minimal grounding sets (the smallest sets of words which need to be *grounded* by prior learning in order to define all the rest), French and English dictionaries were modeled as graphs and reduced in a confluent way, i.e. in a manner yielding unique results, in order to study their respective structures. English dictionaries were also transformed into Abstract Meaning Representation dictionaries, a semantic formalism of interest for cognitive modeling.

INTRODUCTION

Ce mémoire de maîtrise propose de nouvelles approches à des questions de recherche soulevées par Blondin Massée et al., ainsi que Vincent-Lamarre et al. (Blondin Massé *et al.*, 2008; Vincent-Lamarre *et al.*, 2017). Un des objectifs de ce cadre de recherche est d'identifier les plus petits ensembles de mots qu'il faut connaître *au préalable*, pour tout dictionnaire donné, afin de pouvoir apprendre tous les autres mots dans le dictionnaire *par définition*. Or, s'intéresser à répondre à cette question à l'aide de méthodes computationnelles soulève d'importants enjeux au cœur du traitement automatique du langage naturel (TALN) mais aussi de l'optimisation combinatoire. L'objectif de cette maîtrise était d'étudier la possibilité de représenter les définitions des dictionnaires dans un *formalisme sémantique*, l'*abstract meaning representation* (AMR), traduit librement en *représentation abstraite du sens* (RAS), afin d'en apprendre davantage sur la nature et les propriétés de ces plus petits ensembles de mots permettant d'apprendre tous les autres.

Dans un premier temps, il est utile de brièvement définir quelques notions, en commençant par *formalisme sémantique* : la représentation du *sens* d'une phrase à l'aide d'un encodage différent de celui des langages naturels, qui suit des règles formelles. Dans le mémoire, nous montrons comment il est possible de créer des graphes en combinant les définitions d'un dictionnaire traduites dans un formalisme sémantique. Un graphe est une structure de données qui permet de modéliser des entités et leurs relations. On pourrait voir les mots dans une définition comme des concepts, qui ont une *relation définitionnelle* avec le mot défini. Une autre notion importante qui sera détaillée plus loin est le *transversal de circuit de cardinalité minimale* (TCCM), soit un ensemble de nœuds de taille minimale qu'on puisse retirer à un graphe pour qu'il soit dépourvu de cycle (de chemin, dans un graphe, débutant et terminant sur le même nœud). Un autre terme souvent utilisé sera la *rétro-ingénierie* et le verbe *rétro-ingéniérer*. Il sera utilisé dans l'esprit suivant tiré des mots de Richard Feynman : si je comprends un mécanisme, je peux créer un modèle qui fait la même chose (le scientifique avait originalement écrit *what I cannot create, I do not understand*). L'acte de découvrir et recréer les mécanismes déjà existants s'appelle la rétro-ingénierie. Il sera aussi souvent sujet de la *cognition* dans ce mémoire, que nous définirons comme ce qui se passe dans le cerveau humain qui produit la capacité de faire tout ce que les humains sont capables de faire (Harnad, 2017), en l'occurrence des capacités linguistiques.

Ce mémoire se situe donc à l'intersection de l'informatique, de la science cognitive, et de la combinatoire, c'est pourquoi le **premier chapitre** couvre les motivations théoriques et pratiques. Une fois les préalables couverts, le **deuxième chapitre** introduit la principale contribution de ce mémoire : une nouvelle représentation d'un dictionnaire sous forme de graphe en utilisant un formalisme sémantique, dans l'espoir de palier des enjeux méthodologiques soulevés dans le premier chapitre. **Dans le troisième chapitre**, des *réductions de graphes* sont proposées pour réduire les instances de recherche de *transversaux de circuits de cardinalité minimale*, qui sont formellement équivalents aux plus petits ensembles de mots mentionnés dans le précédent paragraphe. Deux algorithmes y sont proposés : un premier utilisant un ensemble de réductions qui a la propriété d'être *confluent* — dont l'application, peu importe l'ordre, résulte en un unique graphe — et l'autre utilisant un ensemble non-confluent, qui lui perd la propriété de résulter en un unique graphe. Le **quatrième chapitre** couvre les expériences de créations des plongements en RAS et de réductions des graphes qui en résultent alors que le **cinquième chapitre** contient la discussion de ces résultats. Le mémoire conclut sur une récapitulation et une ouverture vers de possibles travaux futurs.

CHAPITRE 1
PROBLÉMATIQUE

Ce chapitre introduit la problématique de recherche sur laquelle se concentre ce mémoire de maîtrise. La première section introduit les motivations théoriques issues des sciences cognitives et de l'apprentissage du langage. Ensuite des notions de base sont introduites en théorie des graphes pour faciliter la discussion. Le chapitre termine en établissant un pont entre ces deux domaines.

1.1 Enjeux en science cognitive

Les mécanismes derrière l'acquisition et le développement du langage demeurent à ce jour des problèmes ouverts en recherche. Malgré les exploits fulgurants des *grands modèles de langue* (LLMs) et les importants efforts d'ingénierie en TALN, plusieurs capacités sont toujours uniques aux humains et demandent à être rétro-ingéniérées. Si on accepte la définition de cognition offerte dans l'introduction, on peut comprendre que le langage est une des principales *choses que nous faisons* et est donc un sujet important en science cognitive (étude scientifique de la cognition). L'intersection entre ces domaines est le sujet d'un vaste corps de recherche, c'est pourquoi il est pertinent de préciser plus clairement le contexte théorique qui permet de formaliser les principales problématiques du mémoire.

1.1.1 Contexte théorique de la problématique

Problème facile de la cognition. *Expliquer comment et pourquoi les êtres vivants sont capables de faire tout ce qu'ils sont capables de faire* est connu sous le nom du *problème facile de la cognition* (Chalmers, 1995; Harnad, 2017). La science cognitive devrait avant tout être vue comme une branche de la biologie et plus spécifiquement, la branche de la biologie qui s'intéresse à la rétro-ingénierie de la cognition humaine.

On parle de rétro-ingénierie puisque la cognition est le résultat du travail d'*ingénierie* fait par l'évolution au travers de l'histoire naturelle des espèces. Ce sont des mécanismes qui existent au préalable que nous tentons d'expliquer et de reproduire dans des modèles raffinés itérativement. Certains mécanismes biologiques sont plus faciles que d'autres à rétro-ingéniérer et certains ont

même une forme qui suggère une fonction, comme le cœur qui ressemble à une pompe.

Pour ce qui est de la cognition, le problème est plus complexe. Bien qu'il est difficile de douter que les mécanismes causaux de la cognition se trouvent dans le système nerveux central, regarder la structure d'un cerveau ne révèle pas ses fonctions. Qui plus est, la localisation fonctionnelle, soit associer *où* et *quand* l'activité cérébrale se déroule à des comportements, ne nous renseigne pas sur le *comment* ou le *pourquoi* (Harnad, 2012). Il est aussi digne de mention que la notion même de localisation fonctionnelle est mise à mal par de nombreux résultats récents (Noble *et al.*, 2024).

Une solution au dilemme de définir la *cognition* a été proposée par Alan Turing dans son célèbre *jeu de l'imitation*, qui propose de simplement *s'en tenir aux comportements*. Ce pourquoi la définition du *problème facile* proposée : la cognition *est* ce que la cognition *est capable de faire* (Turing, 2021).

Test de Turing. Turing propose la chose suivante : imaginons un modèle *parfaitement* indistinguable d'un humain dans ses capacités comportementales. Il y a plusieurs modalités pour évaluer l'indistinguabilité comportementale, la plus connue étant la modalité *écrite* : correspondre indéfiniment avec un modèle sans jamais ne pouvoir réaliser que l'on n'interagissait pas avec un humain. Une modalité plus intéressante serait la modalité sensori-motrice : imaginons qu'un collègue est un robot parfaitement indistinguable d'un humain dans ses capacités comportementales corporelles (y compris ses comportement verbaux).

Si le modèle, peu importe la modalité, est indistinguable d'un humain par ses capacités comportementales, on peut affirmer que le mécanisme causal derrière le fonctionnement du modèle est une rétro-ingénierie *réussie* de la cognition humaine, une réponse au *problème facile de la cognition*. C'est ainsi que Turing a pavé le chemin pour des décennies — peut-être des siècles — de recherche en rétro-ingénierie de la cognition humaine, proposant un outil de validation de cette rétro-ingénierie (Harnad, 2008).

Problème difficile du *ressenti* : Le problème difficile est défini comme *expliquer comment et pourquoi (certains) organismes vivants ressentent* (Chalmers, 1995). Bien que la dichotomie problème facile / difficile soit de Dave Chalmers, il est pertinent de noter que Turing annonçait déjà le problème. Ce dernier avançait qu'une fois le problème facile résolu (qu'un modèle indistinguable d'un humain dans ses capacités comportementales fut créé), il ne resterait plus d'espace causal pour

expliquer le *ressenti* (*feeling*), le rendant potentiellement insolvable (Harnad, 2012). Ce problème n'aura pas d'implication directe dans ce mémoire mais il est à noter que le *sens* est une chose avant tout *ressentie* et qu'on y trouve une certaine contradiction avec la notion de la *formaliser*.

Catégorisation (et cognition). Rapprochons-nous enfin du cœur de ce mémoire en définissant le concept de *catégorisation* et son lien avec la cognition. Catégoriser, *c'est faire la bonne chose avec la bonne sorte de chose*. Cette phrase, au premier abord plutôt anodine, contient à elle seule l'objectif de presque toutes les espèces vivantes. Un exemple initial pour approcher l'idée est de s'imaginer sur une île déserte où les champignons sont la seule source de nourriture. Il faut apprendre à reconnaître quels champignons sont comestibles, lesquels ne le sont pas. Une première stratégie serait l'apprentissage de ces deux catégories par essai-erreurs, en goûtant un peu à chacun. Une autre stratégie, propre aux humains, l'apprentissage verbal, est explorée un peu plus loin.

On peut déjà remarquer le lien entre la catégorisation et le *fitness évolutif* : ce qui détermine *ce qui est la bonne chose à faire* avec une sorte de chose donnée est souvent un impératif biologique qui améliore le *fitness* (soit la probabilité de reproduction du matériel génétique). Dans l'exemple ci-haut, pour assurer le maintien de l'homéostasie — soit l'ensemble des mécanismes qui assurent le maintien des différents équilibres nécessaires à la vie d'un organisme — en consommant les bons nutriments, voire en évitant la mort en ne consommant pas un champignon mortel.

Les règles de catégorisation peuvent être innées (ce qui exclut les fonctions végétatives) ou acquises dans les organismes vivants. Prenons le fameux exemple des oisons de Konrad Lorenz pour mettre en évidence la différence entre ces deux méthodes d'apprentissage (Lorenz, 1937). La forme des parents (ce vers quoi les oisons doivent diriger les signaux pour demander la nourriture) doit être apprise, puisque *ce à quoi ressemble les parents n'est pas inné*. Par opposition, la forme du prédateur naturel de l'oie est innée en l'oison : à la première exposition à un stimulus qui aura la forme d'un prédateur naturel, il adoptera un comportement défensif inné.

Quel est donc le lien entre la catégorisation et la cognition ? Si on part de la définition proposée plus tôt que la *cognition soit ce que la cognition fait*, on pourrait proposer que la *cognition est de la catégorisation* (Harnad, 2017). Loin de dire que les humains ne font que *les bonnes choses avec les bonnes sortes de choses*, il s'agit plutôt d'une proposition de la possible fonction du cerveau dans

les organismes avec systèmes nerveux centraux : de la même façon que le cœur est une pompe, les systèmes nerveux centraux ont évolué pour optimiser la capacité à *catégoriser* des organismes vivants. Dit autrement, ce qu'il faut rétro-ingéniérer en science cognitive, c'est la capacité des systèmes nerveux centraux à *faire les bonnes choses avec les bonnes sortes de choses*.

Il est à noter que l'ensemble des règles mathématiques et des définitions formelles de l'algorithmique tombent dans la catégorisation ; on n'a donc pas à se restreindre au domaine du vivant pour étudier cette dernière. Tout modèle d'intelligence artificielle est un modèle pour nous aider à *faire la bonne chose avec la bonne sorte de chose*, que ce soit de prédire, classifier, ou encore générer du texte. Il s'agit dans chaque cas d'exemple de ce qu'on pourrait appeler *catégorisation*.

Avantages évolutifs du langage. Reprenons notre exemple de champignons sur une île déserte pour rendre évident l'avantage évolutif de l'instruction verbale par rapport à l'essai-erreur (l'induction). Si un autre humain habite déjà sur l'île, il pourrait nous indiquer quels champignons sont comestibles, nous évitant de goûter à des champignons potentiellement dangereux. Il pourrait pointer physiquement les champignons comestibles ou de façon plus intéressante pour ce mémoire, il pourrait nous *fournir une description verbale* pour les identifier. Dans le deuxième cas, on utiliserait notre connaissance de catégories déjà acquises (ce qu'est un champignon, ce que à quoi réfère le mot *rouge* ou *pointillé*) pour en apprendre une nouvelle : *les champignons ici comestibles sont rouges et pointillés*.

L'avantage évolutif de l'instruction verbale par rapport à l'induction a aussi été démontré dans des simulations de formes de vie artificielles (Blondin Massé *et al.*, 2010). Cet avantage évolutif du langage est ce qui a mené la planète sur la voie de l'anthropocène et la physionomie humaine s'est adaptée pour faciliter le langage. Prenons l'exemple de la descente du larynx dans la gorge chez les humains : ce trait évolutif, qui permet la production vocale, est aussi le trait qui cause le risque d'étouffement mortel chez l'humain. Or, qu'est-ce qui explique cette capacité unique dans le domaine du vivant ? Cet enjeu théorique est la principale motivation de ce mémoire et est le sujet de la prochaine sous-section.

1.1.2 Problème de l'ancrage des symboles

Lorsque nous entendons ou lisons un mot dont le sens nous échappe, nous pouvons consulter sa définition dans un dictionnaire. Pareillement, si nous trouvons un autre mot inconnu dans cette définition, nous pouvons aller chercher à son tour sa définition ; et ainsi de suite pour tout autre mot inconnu rencontré. Cependant il est évident que tout les mots que nous connaissons n'ont pas été appris exclusivement par la recombinaison d'autres mots, ce qui reviendrait à dire que le mandarin puisse être appris à partir d'un dictionnaire mandarin-mandarin (Harnad, 2001).

Si un langage ne peut être appris que par acquisition de définitions, *il doit exister un ensemble de mots connus au préalable, dont le sens est acquis autrement que par définition verbale, qui permet ensuite d'apprendre tous les autres mots de façon combinatoire*, que ce soit en lisant des définitions dans un dictionnaire ou par instruction verbale. *Expliquer comment et pourquoi les humains ont l'unique capacité d'ancrer dans des symboles arbitraires les choses auxquelles ils font référence* est connu sous le nom du problème de l'ancrage des symboles (Harnad, 1990, 2024). Il est à noter que les symboles peuvent être écrits ou oraux et que dans les deux cas, l'arbitraire du symbole ne ressemble en rien à ce qu'il représente et l'iconicité (le fait qu'un symbole ressemble à ce qu'il représente) est absente ou perdue (Saussure *et al.*, 2020).

1.1.3 Ensemble d'ancrage (minimal)

Reprenons la notion qu'il faille apprendre certains mots au préalable, acquis avant le langage et la capacité d'apprendre des catégories par apprentissage verbale. L'ensemble de ces mots est appelé *ensemble d'ancrage* : c'est l'ensemble de mots qu'il faut connaître au préalable pour pouvoir apprendre tous les autres d'une langue donnée. Une question qu'on pourrait ensuite se poser serait de trouver la taille minimale de tels ensembles ?

Malheureusement, il n'est pas simple d'étudier les ensembles d'ancrages minimaux sans d'importantes approximations. Une telle première approximation serait d'utiliser un *dictionnaire*. Un dictionnaire est censé contenir tous les mots d'une langue donnée, dans un ensemble fini. Comment pourrait-on alors approcher le problème de l'ancrage des symboles et de l'identification d'ensembles d'ancrages minimaux en utilisant des dictionnaires ? Pour répondre à cette question, il faut d'abord définir des notions de base en théorie des graphes.

1.2 Théorie des graphes

La théorie des graphes étant centrale à ce mémoire, il faut introduire un ensemble de notions et de notations qui sont utilisés dans tous les chapitres.

1.2.1 Notions de base

Les graphes sont des structures de données permettant une grande liberté de modélisation. Ils sont composés de nœuds et d'arcs, où les *nœuds* représentent généralement des concepts ou des entités discrètes et les *arêtes* des relations entre ces concepts. Pour une discussion plus complète, voir le manuel de Rosen et pour une discussion formelle, voir le livre de Diestel (Rosen et Krithivasan, 1999; Diestel, 2024).

Un *graphe orienté* ou *digraphe* (graphe pour le reste de ce mémoire) est noté $G = (V, A)$, où V est un ensemble fini de nœuds et A un ensemble d'arêtes orientées (arcs). Les arêtes sont dites orientées puisqu'elles ont toujours un nœud d'origine qui *pointe* vers un nœud de destination. On note l'ensemble des nœuds, pour un graphe G donné, $V(G)$ alors que l'ensemble des arcs pour le même graphe est noté $A(G)$.

Un nœud dans l'ensemble V est noté v, un arc de l'ensemble A est une paire notée (u, v), où le premier élément (u) représente le nœud d'origine et le deuxième élément (v) le nœud de destination : u *pointe* vers v.

Un nœud connecté par un arc à un nœud v est nommé *voisin* et l'ensemble des voisins d'un nœud donné est $N_G(v)$. Les *successeurs* d'un nœud v correspondent à l'ensemble des nœuds destinations reliés par un arc avec v comme nœud origine (noté $N_G^+(v)$), et la cardinalité d'un tel ensemble correspond au *degré extérieur* de v (noté $deg_G^+(v)$). Inversement, les *prédécesseurs* sont les nœuds origines avec v pour nœud destination ($N_G^-(v)$),et $|N_G^-(v)| = deg_G^-(v)$, soit le *degré intérieur* de v. La somme du degré intérieur et extérieur correspond au *degré total* d'un nœud et est noté $deg_G(v)$.

Le graphe résultant de l'opération de retirer un arc est noté $G \setminus (u, v)$. Similairement, le graphe résultant du retrait d'un nœud est noté :

$$G \setminus \{n\} = (V \setminus \{n\}, A \setminus \{(u, v) \in A \mid u = n \text{ ou } v = n\})$$

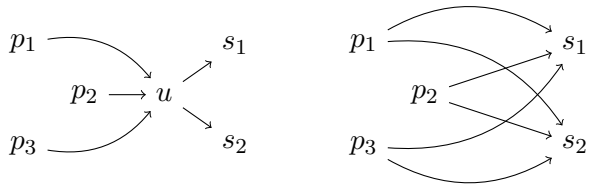


FIGURE 1.1 – Illustration du contournement d'un nœud. Le nœud u les prédécesseurs de u sont connectés aux successeurs.

De plus, on peut retirer des ensembles S de nœuds à un graphe G avec $G \setminus S$, où

$$G \setminus S = (V \setminus S, A \setminus \{(u, v) \in A \mid u \in S \text{ ou } v \in S\})$$

Les symboles $G \circ n$ dénotent le *contournement* (traduit de l'anglais *bypass*) d'un nœud, soit l'exclusion du nœud en connectant tous ses prédécesseurs à tous ses successeurs :

$$G \circ n = (V \setminus \{n\}, (A \setminus \{(x, y) \in A \mid x = n \text{ ou } y = n\}) \cup V_G^-(n) \times V_G^+(n))$$

Cette opération ajoute donc un arc de chaque prédécesseur vers chaque successeur d'un nœud contourné. Cette opération peut aussi introduire une boucle si un nœud est successeur et prédécesseur à la fois. La figure 1.1 illustre ce processus.

Un *chemin* (orienté) est une séquence de nœuds $p = (v_1, v_2, ..., v_k)$ où $v_i \in V$ pour $i = 1, 2, ..., k$ tel que $(v_i, v_{+1}) \in A$ pour $i = 1, 2, ..., k-1$ (il existe une séquence d'arcs qui mène du premier nœud au dernier nœud). Un chemin qui commence et termine au même nœud est nommé *circuit* (si $v_1 = v_k$). Un graphe dépourvu de circuits est dit *acyclique*.

Le graphe $G[S]$ représente le sous-graphe de G induit par l'ensemble de nœuds S, tel que $G[S] = (S, \{(u, v) \in A \mid u, v \in S\})$. L'ensemble des nœuds d'un sous-graphe de G induit est noté $V(G[S])$ et celui des arcs $A(G[S])$. Un sous-graphe de G peut aussi être noté G'. De plus, un sous-graphe G' de G est dit maximal s'il est le sous-graphe de G avec la plus grande cardinalité parmis la famille des sous-graphes de G, généralement selon une condition.

Une *composante fortement connexe* (CFC) d'un graphe G est un sous-graphe maximal G' de G tel qu'il existe un chemin pour toutes paires de nœuds s'y retrouvant (G' tel qu'il existe pour toutes paires $(u, v) \in V(G')$, il existe dans G' un chemin de u à v). La figure 1.2 donne un exemple de deux CFC.

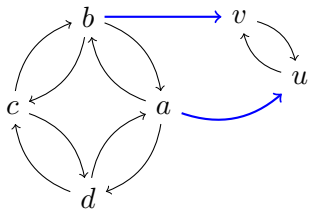


FIGURE 1.2 – Illustration de deux composantes fortement connexes. Les arcs colorés sont ceux ne faisant pas partie d'une CFC.

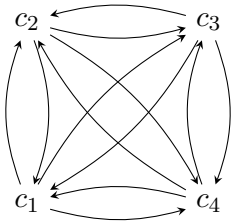


FIGURE 1.3 – Illustration d'une clique. Toutes les paires de nœuds sont reliées par des arcs bidirectionnels.

Pour un arc $a = (v, u) \in A$, si $(v, u) \in A$, a est un *arc bidirectionnel* (ou symétrique). L'ensemble des arcs bidirectionnels de G est noté $A^{\leftrightarrow} = \{(v, u) \in A \mid (u, v) \in A\}$. Pour un graphe $G = (V, A)$, le sous-graphe de G induit ne contenant que les arcs bidirectionnels est noté $G^{\leftrightarrow} = (V, A^{\leftrightarrow})$. Inversement, $G^{\rightarrow}(V, A^{\rightarrow}) = G \setminus A^{\leftrightarrow}$, soit le graphe induit par le retrait de tous les arcs symétriques entre paires de nœuds distincts (incluant le retrait des boucles).

Une *diclique* (ou clique) est un graphe maximal tel qu'il existe une paire d'arcs entre toutes les paires de nœuds. Si U est une diclique de G, alors pour tout arc $(u, v) \in A,$ il existe un arc $(v, u) \in A$. La figure 1.3 illustre une clique. Le prédicat pour noté l'existence d'une diclique U dans G est $c(G, u)$.

Combiner différents graphes ensemble est faire l'*union* de graphes, soit de faire l'union des ensembles N de nœuds et des ensembles A d'arcs de chaque graphe. Formellement, on note $G \cup G' = (V \cup V', A \cup A')$.

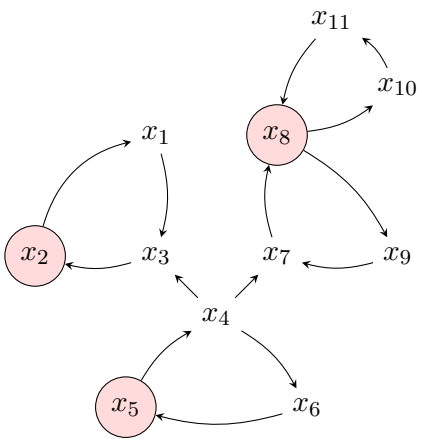


FIGURE 1.4 – Illustration d'un transversal de circuits de cardinalité minimale. Une solution optimale possible serait de retirer les nœuds x_2, x_5 et x_8.

1.2.2 Transversal de circuits de cardinalité minimale

Un *transversal de circuits* (TC) est un ensemble de nœuds U d'un graphe G tel que son retrait du graphe résulte en un graphe acyclique. Plus rigoureusement, $U \subseteq V$ tel que le graphe induit donné par $G \setminus U$ est acyclique.

Pour un graphe G, un plus petit TC est nommé transversal de circuits de cardinalité minimale (TCCM). L'identification des TCCM est NP-difficile et fait partie des problèmes originellement décrits par Karp (Karp, 2010) et est le principal problème en théorie des graphes considéré dans ce mémoire. La figure 1.4 illustre un TCCM.

Le problème du transversal de circuits de cardinalité minimale a une riche et longue histoire en amélioration successive d'algorithmes qui s'étend sur plusieurs décennies (Fomin *et al.*, 2008). Plus récemment, le problème a été au cœur de la compétition PACE 2022, qui a résulté en de nombreuses contributions originales au problème, notamment par l'équipe qui l'a remportée (Kiesel et Schidler, 2023).

Pour le reste du mémoire, l'opération de retirer un nœud et tous ses arcs adjacents d'un graphe a pour synonyme d'exclure un nœud de la solution ; plus simplement encore, on exclut un nœud de la solution en cours de construction. Tout nœud avec une boucle fait forcément partie de n'importe quel TC : puisqu'il faut briser tous les circuits, l'unique nœud d'un circuit de longueur 1 doit être

retiré pour que G soit acyclique.

Une notion à souligner est qu'il peut exister un très grand nombre de TCCM possibles. Bien qu'il n'y ait qu'une seule cardinalité minimale, plusieurs sous-ensembles de cardinalité minimale peuvent rendre acyclique un graphe par leur retrait. Tout circuit de longueur 2 doit être brisé et le choix du nœud est généralement arbitraire, ce qui entraîne une explosion combinatoire de solutions. Par exemple dans la figure 1.3, on peut choisir n'importe quelle combinaison de trois nœuds comme solution optimale. Ceci dit, il peut arriver dans certains cas que le choix du nœud d'un cycle de longueur 2 ne soit pas arbitraire.

1.3 Dictionnaires et graphes

Cette section présente les travaux antérieurs pour compléter l'introduction de la problématique. Pour approximer un ensemble d'ancrage minimal, nous pouvons nous tourner vers les dictionnaires. Un dictionnaire se veut généralement un ensemble fini de tous les mots retrouvés dans une langue donnée quoiqu'il existe de nombreux dictionnaires spécialisés, où peuvent être trouvées des définitions plus spécifiques à certains jargons professionnels par exemple. Notons au passage que la *lexicographie* est la science qui cherche à faire l'énumération et le recensement des mots et leurs définitions pour en faire un dictionnaire.

Pour faciliter la discussion qui va suivre cependant, il est utile d'introduire certaines notions, issues de la lexicographie et de la linguistique, qui seront importantes dans le processus de modélisation des dictionnaires comme graphes.
Une première telle notion est la distinction entre (1) les mots de classe dite *ouverte* (*opened-class words*) qui ont un référent et (2) les mots de classe dite *fermée* (*closed-class words*). Les mots de classes fermées ont généralement moins de *contenu sémantique* et sont plutôt utilisés pour exprimer des relations entre des mots de classes ouvertes. Par exemple, le mot *et* en français n'a pas vraiment de référent : il fait la conjonction de deux mots, ou propositions, qui ont la même fonctions (on peut pointer le référent de *chat* mais pas de *et*). La classe est dite fermée puisque pour une langue donnée, les ensembles de pronoms, de conjonctions, de déterminants etc... sont généralement *finis* et de nouveaux mots n'y sont jamais ajoutés. Notons que les mots de classe fermée sont aussi appelés mots *fonctionnels*. Les mots de classe-ouverte quant à eux consistent de noms, d'adjectifs, de verbes

et d'adverbes. Ces classes grammaticales sont dites ouvertes puisque de nouveaux mots de contenu y sont continuellement ajoutés et il n'y a donc en théorie pas de limite sur leur taille. Les mots de la classe ouverte sont aussi appelés mots de contenu, puisqu'ils contiennent l'essentiel de l'information sémantique contenue dans une phrase.

Deux autres notions qui doivent être définies sont la *polysémie* et la *désambiguïsation du sens des mots* (DSM). La polysémie est la propriété d'un symbole d'avoir plus d'un sens ; un mot polysémique aura dans un dictionnaire plus d'une définition. Lorsque nous reconnaissons le sens utilisé d'un mot qui en a plusieurs dans une phrase donnée, c'est que nous avons correctement désambiguïsé le sens du mot. Bien qu'il s'agisse d'un quasi-automatisme pour les humains, la DSM est un problème pour lequel aucune solution algorithmique n'a été proposée et demeure donc un problème ouvert, malgré les progrès des modèles de langues (Maru *et al.*, 2022).

1.3.1 Dictionnaire comme des graphes

La représentation des dictionnaires à l'aide de graphes a été proposée par Blondin Massé et al. (Blondin Massé *et al.*, 2008). Il faut en un premier temps convertir chaque définition en un graphe orienté. Le mot défini, ainsi que les mots utilisés dans la définition, forment l'ensemble des nœuds. Ensuite, un arc de chaque mot définissant (soit un mot qui est utilisé dans la définition) vers le mot défini est ajouté. Ce processus est illustré dans la figure 1.5 en utilisant une définition du mot *lexicographe*. Il est important de noter que seuls les mots de contenu — les mots de classe ouverte — sont considérés comme nœuds et tous les mots fonctionnels — de classe fermée — sont ignorés.

Pour expliquer ce choix, il faut revenir à la notion de *catégorie* discutée dans la section 1.1 et continuer le développement théorique. Nous proposons que les mots de contenu soient tous des noms de catégories et que nous puissions acquérir le *référent* d'un mot précédemment inconnu en utilisant les référents des mots déjà connus dans sa définition : en ce sens, l'instruction verbale permet une forme d'*ancrage indirect*. Dans le contexte de cette approximation, il semblait raisonnable d'exclure les mots fonctionnels qui n'ont (généralement) pas de référent et qui ne sont pas des noms de catégorie.

Une fois que chaque définition est convertie en un graphe, on peut représenter le dictionnaire comme un graphe à son tour. Pour ce faire, il suffit de prendre l'union de chaque graphe. Ainsi, il en résulte

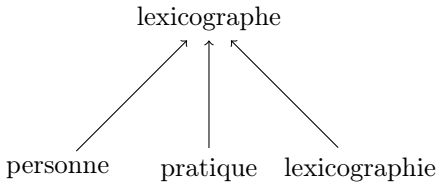


FIGURE 1.5 – Définition de *lexicographe*. *Lexicographe* est défini comme *une personne qui pratique la lexicographie* dans le *Wiktionaire*. En retirant les mots de classes fermées, on obtient *personne, pratique, lexicographie*.

un graphe où les arcs intérieurs de chaque mot proviennent des mots utilisés pour le définir et les arcs sortant de chaque mot pointent aux mots définis par ces mots.

1.3.2 TCCM et ensemble d'ancrage

Une fois les dictionnaires modélisés comme des graphes, on peut tenter de calculer les *ensembles d'ancrage minimaux*, c'est-à-dire les plus petits ensembles de mots d'un dictionnaire donné, à partir desquels la connaissance préalable permet d'apprendre tous les autres mots de contenu dans le dictionnaire donné. Voyons enfin en quoi la théorie des graphes nous aide à trouver ces ensembles d'ancrages.

Il a été démontré par Blondin Massé et al. que les ensembles de mots qu'il faut connaître au préalable pour pouvoir apprendre tous les autres sont équivalents aux transversaux de circuits (Blondin Massé *et al.*, 2008). On pourrait dire qu'il s'agit du même problème formulé différemment. Un ensemble d'ancrage minimal est un ensemble de mots qui permet d'apprendre tous les autres par définition alors qu'un TCCM est un ensemble de nœuds — de mots dans nos graphes issus de dictionnaires — dont l'exclusion permet de briser la circularité des définitions dans le dictionnaire.

1.3.3 Structures de dictionnaire et solveur de TCCMs

Il a été démontré que pour identifier le TCCM d'un graphe donné, il suffit de décomposer le graphe en composantes fortement connexes et de trouver le TCCM de chaque CFC. L'union de chacune de ces CFC constitue un TCCM du graphe donné (Lin et Jou, 2000). Voyons comment nous pouvons utiliser les structures communes aux dictionnaires en ce sens.

FIGURE 1.6 – (a) Le graphe avant la *kernelisation* (le retrait récursif des mots mots non-définis et des mots non-définissants), où x_1 n'est pas défini et où x_4 n'est pas définissant (b) Le graphe après la kernelisation, où les mots non-définis et non-définissants sont retirés récursivement ; x_2 et x_3 sont retirés après x_1 et x_4

Lors de travaux antérieurs par Vincent-Lamarre *et al.*, il a été observé que huit dictionnaires anglais partagent les structures suivantes : un *noyau*, qui est obtenu en retirant récursivement tous les mots qui n'en définissent aucun (Vincent-Lamarre *et al.*, 2016). La figure 1.6 démontre un exemple de ce processus appelé *kernelisation*. Le *cœur* est la plus grosse CFC du noyau ; toutes les autres CFC du noyau sont dites *satellites*. Dans ces dictionnaires, l'identification des TCCM des satellites est réalisée en un temps négligeable et seuls les cœurs sont assez gros pour qu'identifier leur TCCM soit un problème : il faut utiliser un solveur à l'état-de-l'art. Pour les satellites, il suffit d'employer un algorithme qui utilise des *réductions de graphes* (sujet du chapitre 3), alors que ce même algorithme ne fait que créer une *solution partielle* dans le cas du cœur, cœur dit *réduit*.

Pour *briser la circularité* de ces cœurs *réduits*, on peut employer la programmation linéaire en nombres entiers binaires (PLNE). Une approche (que j'ai implémentée dans le cadre de ma thèse d'honneur) est de modéliser les circuits comme les équations d'un système qu'il faut minimiser, où les variables sont les mots qui ne peuvent prendre pour valeur que 0 ou 1 : un mot est ou n'est pas dans un TCCM. Chaque circuit converti est une somme de nœuds (de mots) qui doit être plus grande ou égale à un, ce qui assure que chaque circuit soit brisé. Minimiser ces équations revient à choisir le plus petit ensemble de mots pouvant briser tous les circuits. Illustrons le modèle d'équations associé au graphe présenté dans la figure 1.4. La fonction objectif est la suivante :

$$\text{Minimiser} \quad x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} = 1 \tag{1}$$

sous les contraintes :

$$x_8 + x_{10} \geq 1 \tag{2}$$

$$x_1 + x_2 + x_3 \geq 1 \tag{3}$$

$$x_4 + x_5 + x_6 \geq 1 \tag{4}$$

$$x_7 + x_8 + x_9 \geq 1 \tag{5}$$

$$x_7 + x_8 + x_9 + x_{10} \geq 1 \tag{6}$$

$$x_i \in \{0, 1\} \quad \forall i \in V \tag{7}$$

De façon plus générale, on note $\mathcal{C}$ comme la famille des circuits C d'un graphe G et le programme linéaire en nombres entiers associée à cette famille, lui, est noté $IP(\mathcal{C})$. La formulation $IP(\mathcal{C})$ pour un graphe $G(V, A)$ donné est la suivante :

$$\text{Minimiser} \quad x(V) \tag{1}$$

sous les contraintes :

$$x(C) \geq 1 \quad \forall C \in \mathcal{C} \tag{2}$$

$$x_i \in \{0, 1\} \quad \forall i \in V \tag{3}$$

Convertir chaque circuit $C \in \mathcal{C}$ en équation n'est pas raisonnable de façon générale : l'énumération de circuits est un problème #P-complet bien connu et la cardinalité de $\mathcal{C}$ est infinie, à moins de s'en restreindre aux circuits élémentaires, où la cardinalité de $\mathcal{C}$ est finie mais très grande (Valiant, 1979). Pour initialement contourner le problème, on peut considérer une famille plus petite de circuits, notée $\mathcal{C}_k$, qui sera restreinte aux circuits au plus d'une certaine longueur (représentée par le paramètre k). On peut trouver une solution sous-optimale $IP(\mathcal{C}_k)$, puis augmenter itérativement la cardinalité de $\mathcal{C}_k$ en admettant des circuits plus longs. Dès que l'ajout d'un nouveau circuit rend invalide la solution sous-optimale, on met à jour $IP(\mathcal{C}_k)$.

Cependant, même en restreignant initialement le problème à des circuits assez courts, le problème de

programmation linéaire en nombres entiers binaires demeure trop complexe et lent à résoudre. C'est pourquoi une *relaxation continue* est d'abord employée, notée $LP(\mathcal{C}_k)$. Cette solution approximative est ensuite utilisée dans une modélisation plus stricte en nombres entiers binaires comme point de départ. Il est intéressant de noter que c'est la contrainte qu'une variable ne peut prendre que deux valeurs qui rend la formule de PLNE plus difficile que sa relaxation linéaire. Cette approche est résumée dans l'algorithme 1.

Algorithme 1 Résolution par la génération de contraintes

Entrée : $G(V, A)$: graphe, $\mathcal{C}^0$: famille restreinte de circuits de G

Sortie : S : un sous-ensemble de cardinalité minimale de noeuds tel que leur retrait rende le graphe G acyclique.

$\mathcal{C}_k \leftarrow \mathcal{C}^0$
$S \leftarrow$ une solution optimale de $LP(\mathcal{C}_k)$
tant que $S(C) < 1$ est vérifié pour un C $\in \mathcal{C}$ **faire**
    ajouter à $\mathcal{C}_k$ au moins un circuit violé par S
    $S \leftarrow$ une solution optimale de $LP(\mathcal{C}_k)$
$S \leftarrow$ une solution optimale de $IP(\mathcal{C}_k)$
tant que $S(C) < 1$ est vérifié pour un C $\in \mathcal{C}$ **faire**
    ajouter à $\mathcal{C}_k$ au moins un circuit violé par S
    $S \leftarrow$ une solution optimale de $IP(\mathcal{C}_k)$
retourner S

1.3.4 Problématiques

La principale difficulté de la représentation des dictionnaires comme des graphes est le traitement des mots polysémiques. Il y a aussi des enjeux d'une nature purement combinatoire : les cœurs de certaines instances demeurent trop grosses pour identifier leur TCCM en un temps raisonnable. Le sujet principal de ce mémoire est l'exploration de nouvelles stratégies pour pallier ces limitations.

La première problématique à considérer dans le traitement des mots polysémiques est la collision entre les nœuds lors de l'union des graphes, illustrée dans la figure 1.7. En effet, si on décide d'inclure toutes les définitions du mot *orange* sans leur donner d'étiquettes uniques (comme *orange-i, orange-ii*), tous les mots de contenu utilisés dans les définitions du mot orange pointeraient vers un unique nœud *orange*. La deuxième problématique est la DSM nécessaire lorsqu'un mot polysémique est

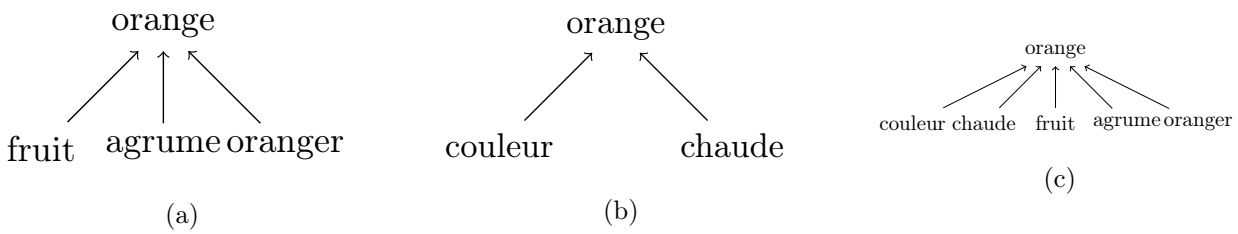


FIGURE 1.7 – **(A)** La définition d'*orange* comme fruit. **(B)** La définition d'*orange* comme couleur. **(C)** Les collisions lors de l'union des graphes.

utilisé dans une définition. Si le mot *orange* est utilisé dans une définition, même si nous avons décidé de donner une étiquette unique pour chaque définition du mot orange, nous n'avons pas de solution algorithmique satisfaisante au problème de la DSM. Pour ces raisons, Blondin Massé *et al.* et Vincent-Lamarre *et al.* ont décidé de ne conserver que la première définition de chaque mot et d'assumer qu'elle seule était utilisée dans chaque définition (Blondin Massé *et al.*, 2010; Vincent-Lamarre *et al.*, 2017).

Le chapitre 2 se penche entre autres sur l'utilisation de la *représentation abstraite du sens* comme une potentielle solution à ces problèmes. Par exemple, la représentation abstraite du sens pourrait nous permettre de représenter et définir des *compléments de catégorie* (soit ce qui ne rentre pas dans une *catégorie*) grâce à des notions comme la négation. L'espoir est que la RAS permette une représentation plus riche du contenu sémantiques des définitions, voire même des mots fonctionnels les plus importants.

Pour ce qui est des cœurs réduits encore trop larges, une approche serait d'élargir l'ensemble des réductions de graphes utilisés ou d'explorer de nouveaux algorithmes pour modéliser le problème du TCCM. Lors de travaux antérieurs par Vincent-Lamarre et al., les seules réductions employées étaient tirées de Lin et Jou, alors que de nombreuses nouvelles réductions pour l'identification des TCCM ont depuis été proposées (Lin et Jou, 2000; Vincent-Lamarre *et al.*, 2016, 2017). Pour ce qui est des nouveaux algorithmes, il serait possible de convertir les équations décrites dans la sous-section 1.3.3 en clauses d'un solveur MaxSAT, comme utilisé par l'équipe qui a remporté la compétition PACE 2022 (Kiesel et Schidler, 2023). Pour le présent mémoire cependant, la seule avenue explorée est plutôt celle d'élargir l'ensemble des réductions, et de nouvelles sont donc proposées dans le chapitre 3.

CHAPITRE 2
REPRÉSENTATION ABSTRAITE DU SENS

Ce chapitre introduit le travail réalisé dans le contexte des formalismes sémantiques, plus précisément sur la représentation abstraite du sens RAS (traduit de l'anglais *Abstract Meaning Representation*) (Banarescu *et al.*, 2013). La première section du chapitre décrit le contexte théorique autour de la RAS. La deuxième section décrit les efforts et méthodes pour obtenir la représentation du contenu de dictionnaires digitaux comme l'union de leur plongement en RAS.

2.1 Contexte théorique

Cette section commence en introduisant les travaux en formalisme sémantique qui ont joué un rôle important dans la création de la RAS et termine sur une définition de la RAS en soi.

2.1.1 Fondements de la RAS

Pour bien comprendre la RAS, il est utile de faire un bref survol de l'historique de travaux en formalismes sémantiques dont le complexe mélange a inspiré la RAS.

Le premier bloc de la RAS est la *sémantique néo-davidsonienne* (Davidson, 1969). La sémantique davidsonienne (nommée ainsi en l'honneur de son créateur) est un formalisme sémantique qui cherche à modéliser le contenu des phrases comme des évènements, évènements représentés sous la forme d'argument dans des structures logiques, par exemple *Jean mange une pomme* :

$$\exists e \, (\text{Manger}(e) \wedge \text{Agent}(e, \text{Jean}) \wedge \text{Patient}(e, \text{Pomme}))$$

Dans l'exemple ci-haut, e représente une variable d'évènement qui encode l'action. L'agent (Jean) est l'entité performant l'action alors que le patient (la pomme) est l'entité qui reçoit l'action. Une telle représentation du sens est assez limitée, c'est pourquoi les sémantiques néo-davidsoniennes ont progressivement été raffinées pour introduire des relations entre les différents éléments d'un évènement, comme illustré dans l'exemple qui suit pour la phrase *Jean mange une pomme au parc* :

$$\exists e\,(\text{Manger}(e) \wedge \text{Agent}(e, \text{Jean}) \wedge \text{Patient}(e, \text{Pomme}) \wedge \text{Lieu}(e, \text{Parc}))$$

Le prochain bloc élémentaire à connaître avant d'aborder plus en détail la RAS est la *notation Penman* (Matthiessen et Bateman, 1991). Il s'agit d'une représentation par graphe du contenu sémantique qui était originellement utilisée pour générer du langage naturel. Elle a aussi la particularité de pouvoir être représentée textuellement sous forme de triplets, particularité qu'on retrouve dans plusieurs formalismes sémantiques. En ce sens, nous considérons plus loin qu'un graphe RAS est aussi un graphe Penman valide. La notation Penman étant plus générique et flexible, ainsi qu'agnostique au langage, nous pourrions reprendre la phrase *Jean mange une pomme au parc* et l'écrire sous notation Penman, tel qu'illustré dans la figure 2.1.

```
(m / manger
  :agent (j / Jean)
  :patient (p / pomme)
  :lieu (p2 / parc))
```

FIGURE 2.1 – Représentation Penman de la phrase *Jean mange une pomme au parc*. Dans cette notation, / représente une instance d'un concept. Pour chaque concept, on a un mot (Jean, pomme, parc) et un pointeur unique (j, p et $p2$). Les mots commençant par : indiquent une fonction ou une relation.

Ensuite, il faut discuter du projet *PropBank* et de ses *frames*, traduites ici librement par *cadres* (Palmer *et al.*, 2005). *PropBank* est un corpus annoté de verbes (nommés prédicats), où chaque verbe est détaillé dans un *cadre* (une *frame*) avec ses *arguments* — soient les différents éléments sémantiques en lien avec un verbe donné. Chaque argument définit le rôle d'un élément sémantique dans une phrase. Généralement, l'*ARG0* (le premier argument d'un prédicat) est l'agent et l'*ARG1* (le deuxième argument d'un prédicat) est le patient. Le cadre de *eat-01* est illustré à la figure 2.2.

Ce qu'il faut surtout retenir de *PropBank*, c'est sa structure *prédicat-arguments*, formalisée comme cadre, qui est utilisé dans la RAS. L'objectif du corpus est d'offrir un ensemble de cadres mais ne fournit pas de phrases annotées (des phrases *traduites* en *PropBank*). Il est intéressant de noter que les cadres de *PropBank* peuvent couvrir plusieurs sens pour un même verbe comme illustré dans la figure 2.3.

```
eat.01 - consume, consuming
Aliases:
  eat (v.)
  eating (n.)
Roles:
  ARG0-PAG: consumer, eater
  ARG1-PPT: meal
```

FIGURE 2.2 – Cadre *eat-01* de Propbank. Le cadre commence par une définition du sens regardé : *eat.01* est ici défini comme *consume, consuming*. Ensuite quelques synonymes puis les rôles, qui sont les relations sémantiques qu'un verbe entretient avec d'autre mot.

Cependant, *PropBank* demeure assez limité par la décision de n'avoir des cadres que pour les verbes. C'est là qu'intervient le dernier fondement de la RAS dont il faut discuter, le projet *OntoNotes* (Hovy *et al.*, 2006). Il s'agit d'un corpus multilingue d'articles de nouvelles annotées par des linguistes-experts, qui cherche à enrichir la méthodologie de PropBank (OntoNotes est bâtie à partir de ce dernier) en ajoutant aux relations prédicats-arguments la désambiguïsation du sens des mots et des sens annotés pour d'autres mots que les verbes, l'ajout d'entité-nommée et plusieurs *couches* sémantiques. Ce corpus se démarque aussi par le rigoureux processus d'annotations qui exige un taux d'accord inter-expert (*annotator agreement*) d'environ 90%. Somme toute, *OntoNotes* est un projet historiquement important dans le traitement du langage naturel et constitue une composante fondamentale de la RAS.

2.1.2 Représentation abstraite du sens

La RAS, développé par Banarescu *et al.*, est un formalisme sémantique qui cherche à obtenir des représentations par graphes orientés acycliques du *sens* des phrases, en s'affranchissant de la syntaxe (Banarescu *et al.*, 2013). Elle est librement inspirée de plusieurs notions décrites dans la sous-section précédente (*sémantique néo-davidsonienne*, *notation Penman*, *corpus PropBank* et *corpus OntoNotes*). Un objectif de ce formalisme est que différentes phrases construites différemment mais ayant le même sens doivent être représentées par le même graphe RAS et il est à noter que la RAS ne peut pas être utilisée pour représenter plusieurs phrases : le formalisme ne considère qu'une phrase à la fois. Cet objectif implique que pour tout graphe RAS donné, on ne peut pas tirer une

```
devolve.01 - pass on to a successor
Aliases:
  devolve (v.)
  devolution (n.)
Roles:
  ARG0-PAG: giver, agent
  ARG1-PPT: thing given
  ARG2-GOL: recipient
```

```
devolve.02 - grow worse
 Aliases:
   devolve (v.)

 Roles:
   ARG0-PAG: agent, causer of devolution
   ARG1-PPT: patient, thing growing worse
   ARG2-DIR: start state
   ARG3-PRD: end state
```

FIGURE 2.3 – Exemple d'un verbe avec plusieurs sens couverts par un même cadre Propbank. Le mot *devolve* est défini comme *pass on to a successor* ou *grow worse*, et chacun a son propre propre ensemble d'arguments.

phrase unique : plusieurs phrases d'anglais sont validement traduites à partir d'un même graphe RAS. Cette propriété est illustrée dans la figure 2.5.

Notons aussi que les auteurs proposent de voir la création des graphes RAS à partir de phrases comme la traduction de l'anglais vers une autre langue. Essentiellement, la RAS est une base de données d'un peu plus de 50 000 phrases d'anglais traduites par un groupe d'annotateurs-experts en graphes RAS, disponible à l'achat d'une licence. Une fois la licence obtenue, le jeu de données peut être utilisé pour entraîner son propre modèle et compléter plusieurs tâches, comme la traduction automatique (Li et Flanigan, 2022), la création de résumés de texte (Dohare *et al.*, 2017; Kouris *et al.*, 2024) ou même pour faciliter la communication humain-robot (Bonial *et al.*, 2023) ; voir la revue de Tohidi et Dadkhah pour une revue plus systématique des cas d'applications du formalisme (Tohidi et Dadkhah, 2022). Une autre caractéristique importante de la RAS est qu'elle n'est pas une interlingua et qu'elle ne peut être utilisée pour représenter le sens de phrases d'une langue autre que l'anglais.

Il existe cependant plusieurs tentatives d'adapter spécifiquement ce formalisme à d'autres langues, ici listées non exhaustivement, comme le turc (Oral *et al.*, 2024), l'espagnol (Wein *et al.*, 2022), le portugais-brésilien (Cabezudo et Pardo, 2019) ou encore le persan (Takhshid *et al.*, 2022). Il est intéressant de noter qu'il existe aussi des formalismes sémantiques inspirés de la RAS qui ont

pour objectif d'être des représentations abstraites du sens des phrases *multilingue*, comme l'*Uniform Meaning Representation* (Van Gysel *et al.*, 2021), qui propose d'étendre le formalisme à plusieurs langues, dont certaines dites à faibles ressources (pour lesquelles il n'existe pas de grands jeux de données facilement accessibles), ou encore le *BabelNet-Meaning-Representation* (Martínez Lorenzo *et al.*, 2022), bâti à l'aide d'un jeu de données multilingues et multimodales (avec des *mappings* entre les différentes traductions d'un même mot, avec images et sons associés). Cette dernière pourrait être d'intérêt lors de travaux futurs associés à ce mémoire.

Voyons maintenant plus en détail le formalisme de la RAS. Comme mentionné plus tôt, les phrases traduites en RAS deviennent des graphes orientés et acycliques, où la racine de chaque graphe est le *focus* de la phase. Cette propriété joue un rôle très important qui est approfondi dans la prochaine section et peut être utilisée comme critère dans la création de graphe RAS à partir des définitions. Les nœuds dans ces graphes sont généralement des concepts alors que les arcs forment des relations sémantiques, avec des occasionnelles relations syntaxiques, comme pour la négation, la conjonction, et autres éléments du langage fonctionnels qui jouent un rôle important dans le sens des phrases. Les nœuds sont étiquetés de *cadres* d'OntoNotes lorsque possible et de mots anglais dans les autres cas. La figure 2.5 présente le graphe RAS obtenu à la fois à partir de la phrase *apple is defined as a red round fruit* et/ou à partir de la phrase *a red round fruit is the definition of apple*. Un graphe RAS peut aussi être représenté dans un format textuel : par exemple, la phrase *category is defined as a general concept that mark division or coordination in a conceptual scheme* donne le graphe RAS illustré à la figure 2.4.

Ce formalisme pourrait permettre une meilleure représentation du contenu des dictionnaires pour plusieurs raisons. Premièrement les différents sens d'un même mot ont des étiquettes uniques et numérotées comme illustré à la figure 2.6. Cette propriété, absente des dictionnaires, permettrait d'éviter les collisions lors de l'union des différentes définitions d'un même mot. Similairement, un mot avec plusieurs définitions pourrait être traduit en un symbole RAS complètement différent, selon le sens de la définition, puisque le formalisme peut être considéré comme la traduction de l'anglais vers une autre langue.

Un autre avantage de la RAS sur le langage naturel est une représentation moins atomique du sens, qui résulterait en des graphes moins denses, lorsque le sens implicite d'un mot serait traduit en

```
(d / define-01
      :ARG1 (c / category)
      :ARG2 (c2 / concept
            :ARG1-of (g / general-02)
            :ARG0-of (m / mark-01
                  :ARG1 (o / or
                        :op1 (d2 / divide-02
                              :ARG1 (s / scheme
                                    :mod (c3 / conceptual)))
                        :op2 (c4 / coordinate-01
                              :ARG1 s)))))
```

FIGURE 2.4 – Représentation textuelle d'un graphe RAS. Comme pour le notation Penman, les symboles / représentent des instances, constitué d'un mot et d'un pointeur unique. En plus des `:ARGX`, on remarque des marqueurs de relations de juxtaposition comme `:op1` et `:op2`. Lorsqu'un même mot réapparait dans la structure du graphe RAS, le pointeur est utilisé au lieu du mot (puisque un même mot peut survenir plusieurs fois dans une phrase, il faut pouvoir distinguer la fonction de chacun).

plusieurs concepts en relation entre eux. De plus, alors que les mots fonctionnels sont complètement exclus de la modélisation dans les travaux de Vinvent-Lamarre *et al.*, la RAS pourrait conserver certains mots fonctionnels qui peuvent jouer un rôle significatif dans la sémantique des phrases, comme la négation (Vincent-Lamarre *et al.*, 2017). Pour conclure l'introduction de la RAS, nous mentionnons qu'il s'agit d'un passage vers une approche plus *sémantique* que *lexicographique* dans l'identification des ensembles d'ancrages minimaux.

2.2 Dictionnaires comme graphes RAS

Nous sommes maintenant prêts à introduire la principale contribution de ce mémoire, la représentation du contenu d'un dictionnaire comme graphe RAS dans l'espoir de palier certaines limitations de la représentation du contenu des dictionnaires comme graphes réguliers (Vincent-Lamarre *et al.*, 2017).

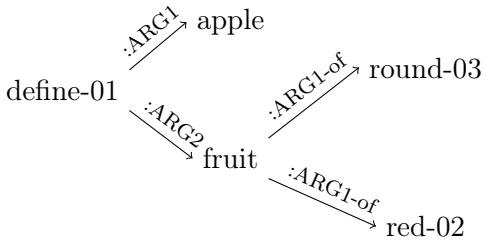


FIGURE 2.5 – Graphe RAS obtenu en traduisant les phrases *apple is defined as a red round fruit* et *a red round fruit is the definition of apple*. Puisque cette phrase est une définition *define-01* est le focus de ce graphe RAS. Dans *PropBank*, l'`ARG1` de *define-01* est la chose définie (*thing defined*) et l'`ARG2` est la *definition*.

```
admonish.01 - persuade warningly
Aliases:
  admonition (n.)
  admonish (v.)
Roles:
  ARG0-PAG: persuader
  ARG1-GOL: persuaded agent
  ARG2-PPT: persuaded action
```

```
admonish.02 - chastise
Aliases:
  admonish (v.)

Roles:
 ARG0-PAG: chastiser
ARG1-PPT: chastised
ARG2-CAU: wrongdoing
```

FIGURE 2.6 – Exemples de différents sens pour un même mot (propre à RAS, qui ne se trouvent pas dans PropBank). Pour mettre en évidence l'utilisation des différentes étiquettes numérotées pour le sens d'un mot.

2.2.1 Plongement des définitions en RAS

La première étape est la traduction, ou plongement, des définitions en langage naturel vers la RAS. Simplement isoler les définitions, les traduire en graphes RAS et ensuite en faire l'union semble moins adaptée : une telle approche perdrait la référence vers le mot défini. Nous préférons donc l'approche de créer des phrases à l'aide de chaque paire de *mot défini* - *définition* (symbole - définition, s - d) en utilisant une des structures de phrases présentées dans le tableau 2.1.

Il est à noter que la première formulation est celle utilisée au départ. Une fois les phrases générées, il faut les traduire en graphe RAS. Une première approche pour les traduire pourrait être d'apprendre l'annotation et générer soi-même des graphes puisqu'il n'existe pas de solutions algo-

1. "s is defined as d."
2. "The definition of s is d."
3. "s has for definition d".
4. "d is the definition of s."
5. "s is defined by d."
6. "s gets defined as d."

TABLE 2.1 – Ensemble de formulation de phrases pour mettre l'accent sur la relation définitionnelle. Ces formulations de définitions sont successivement utilisées jusqu'à l'obtention d'un graphe RAS valide.

rithmiques exactes au plongement en RAS. Or, si on considère la quantité de définitions dans un seul dictionnaire, on réalise vite que cette approche est irréaliste. C'est là qu'intervient l'architecture *transformer*, principale composante des architectures de *grands modèles de langues* (Vaswani, 2017). En effet, ces modèles rendus célèbres par des récents *chatbots* permettent l'automatisation de tâches en TALN dont on croyait l'exécution propre aux humains.

La question est alors de soit créer son propre modèle, soit utiliser un modèle préentraîné. Cette dernière option est celle qui a été retenue : ce mémoire ne s'intéresse pas à l'optimisation des techniques d'apprentissage-machine pour le plongement des phrases en RAS. Aucune théorie cognitive ne justifie ce choix et nous utilisons ici un grand modèle de langues comme un *outil*. Heureusement, il existe une bibliothèque pour le langage *Python* qui permet d'utiliser les modèles à l'état-de-l'art du *sentence-to-graph* (phrases-vers-graphes), *Amrlib* (qui est discutée dans le prochain chapitre), pour aisément plonger des phrases en leur graphe RAS (Jascob, 2023).

2.2.2 Contraintes et spécificités

Il faut maintenant introduire les spécificités désirées des graphes RAS obtenus. Les cas idéaux sont illustrés dans les figures 2.5 et 2.7. Dans ces cas, le nœud racine est étiqueté *define-01*, démontrant que le modèle a correctement *prédit* que le *focus* de la phrase est d'être une définition. Le cadre *define-01* a deux arguments qui sont ici d'intérêt : le premier argument (`ARG1`) qui représente ce qui est défini (soit le s du tableau 2.1) et le deuxième argument (`ARG2`) qui représente la définition (soit le d du tableau 2.1). Dans ces cas idéaux, le graphe RAS produit n'a que ces deux arcs émanant de la

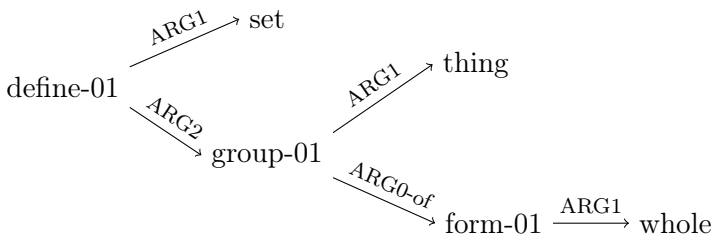


FIGURE 2.7 – RAS généré à partir de la définition *set is defined as a group of things that form a whole*.

racine. Une autre contrainte importante était la traduction de s en un sous-graphe d'un seul nœud. La non-atomicité de la RAS est un avantage pour les définitions mais rendrait inutilisable pour nos fins un graphe qui aurait un sous-graphe de plusieurs nœuds pour représenter le concept défini. C'est qu'il faut garder en tête que nous souhaitons ultimement avoir une forme similaire à celle de la figure 1.5 afin d'obtenir des TCCM comparables (i.e., qui représentent le plus petit ensemble de symboles qu'il faut connaître au préalable tel que tous les autres symboles soient apprenables par définition). Concaténer les étiquettes de ces sous-graphes pour former un nœud unique n'aide pas non plus : nous ne ferions que créer des mots artificiels qui ne surviennent dans aucune définition.

Ces cas idéaux produisent des graphes RAS dits *valides*. Malheureusement, il arrive souvent que des graphes RAS invalides soient produits, tel qu'illustré et décrit dans la figure 2.8. C'est pourquoi nous proposons deux stratégies générales pour optimiser le nombre de graphes valides. La première stratégie s'applique si le graphe avait un nœud racine *define-01* et s'il y avait un seul nœud au bout du premier argument (`:ARG1`). Dans ces cas, on peut ne conserver que les parties valides du graphe RAS, supprimer le reste et générer, grâce au modèle de langue, un nouveau sous-graphe définitionnel à partir de d seulement. Ce sous-graphe peut ensuite être concaténé à *define-01* avec un arc étiqueté *ARG2*. Ce processus est illustré dans la figure 2.9.

Pour les cas d'erreurs n'étant pas couverts par la première stratégie, une autre stratégie consiste à essayer une autre structure de phrase du tableau 2.1 et voir si on obtient un graphe RAS valide. Lorsque ces nouveaux RAS étaient eux-mêmes invalides, ils étaient à leur tour des nouveaux candidats sur lesquels essayer la première stratégie. La stratégie générale d'optimisation du nombre de graphes valides peut être résumée ainsi :

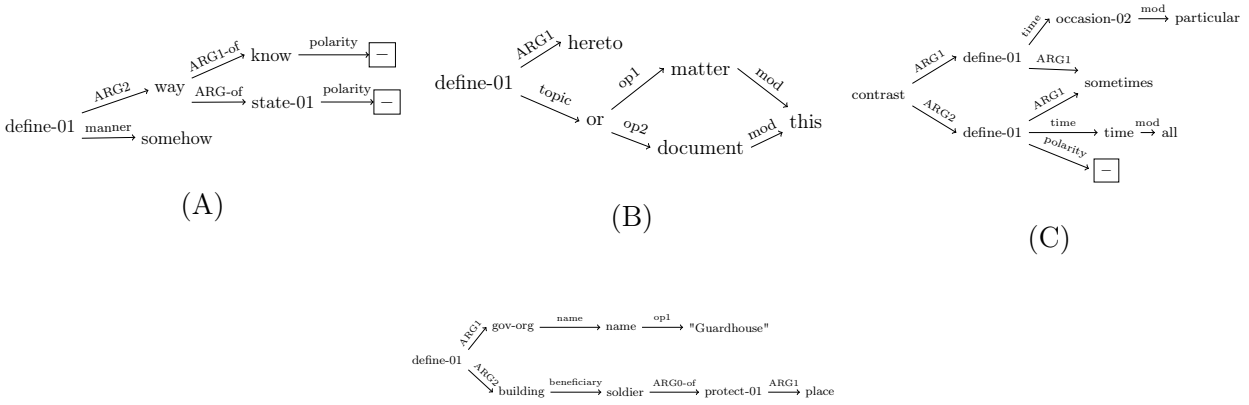


FIGURE 2.8 – **(A)** Un RAS sans ARG1. L'arc incorrectement étiqueté *manner*. **(B)** Un RAS sans ARG2. L'arc est ici incorrectement étiqueté *topic*. **(C)** Un RAS avec le mauvais nœud racine. Le modèle a incorrectement interprété la phrase comme le contraste de deux définitions. **(D)** Un RAS où s est traduit par plus d'un nœud.

— Appliquer la première stratégie.
— Tant qu'il reste des RAS invalides, essayer avec une nouvelle structure de phrase.
— Chaque nouveau candidat invalide est vérifié comme candidat pour la première stratégie.
— S'il reste des graphes invalides après que toutes les structures de phrases aient été essayées, ils ne sont pas conservés dans le jeu de données (les définitions sont perdues).

2.2.3 RAS et polysémie

Une fois que la quantité de graphes valides a été maximisée, il reste à valider le traitement de la polysémie lors du plongement des définitions en RAS. Contrairement aux travaux antérieurs, nous avons tenté d'utiliser toutes les définitions des mots polysémiques au lieu que de ne conserver que la première. Ce choix crée deux sources de collisions pour lesquelles il n'existe pas d'autres solutions que le retrait de certaines définitions.

Premièrement, les cadres de l'AMR ne peuvent pas contenir tous les sens possibles d'un mot au travers de plusieurs dictionnaires. Prenons $s - RAS$ comme le plongement RAS du mot défini s. Il va donc arriver qu'un mot polysémique ait le même $s - RAS$ pour différentes définitions de s.

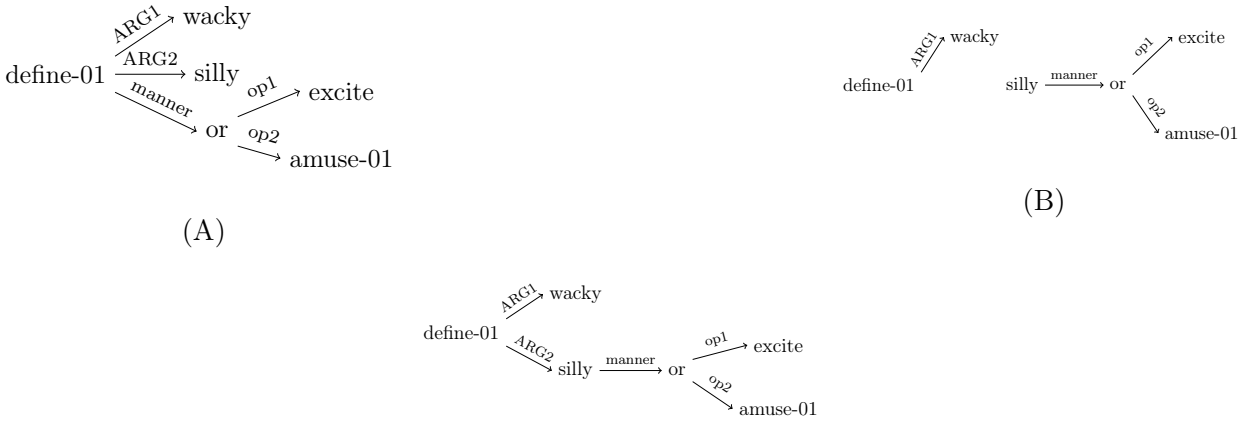


FIGURE 2.9 – Processus de correction pour la définition *wacky is defined as silly in an exciting or amusing way.* **(A)** Un graphe RAS invalide avec un arc en trop. **(B)** Les éléments valides sont préservés ; un nouveau RAS est crée à partir de d (*silly in an exciting or amusing way*). **(C)** Le nouveau RAS est connecté au nœud racine par un arc étiqueté ARG2.

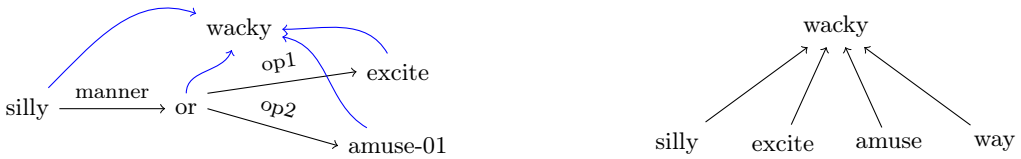
Dans ces cas, il faut utiliser une stratégie similaire à celle des travaux de Vincent-Lamarre *et al.* : ne conserver que la première occurrence de chaque $s-RAS$ et rejeter toutes les autres (Vincent-Lamarre *et al.*, 2017) .

Deuxièmement, il faut vérifier s'il n'y a pas de telles collisions *entre les différents mots*. Il est possible que différents mots de langages naturels aient le même $s - RAS$, auquel cas on applique la même stratégie que celle du précédent paragraphe : on passe sur l'ensemble des $s - RAS$ des graphes valides et on ne conserve que la première occurrence de chaque $s - RAS$.

2.2.4 Format final du plongement

Nous sommes maintenant prêts à décrire les dernières étapes dans la création des jeux de données : le contournement du nœud de *define-01* pour recréer les relations définitionnelles et l'union subséquente des graphes RAS.

Le *contournement du nœud racine* est une étape importante pour créer des graphes dont la structure est similaire à celle de Vincent-Lamarre et al. décrite dans le premier chapitre (Vincent-Lamarre



(A) (B)

FIGURE 2.10 – **(A)** Un graphe RAS où la racine a été contournée. Chaque arc de couleur bleue a été ajouté après ce contournement et est étiqueté *define-01* pour signifier une relation définitionnelle. **(B)** La précédente modélisation par graphe (Vincent-Lamarre *et al.*, 2016). Les deux sont faits à partir de la définition du mot *wacky*.

et al., 2017). Une fois qu'il reste seulement des RAS avec des nœuds racines *define-01*, des $s-RAS$ uniques et atomiques (composés d'un seul nœud) et avec des représentations par un unique sous-graphe de d (soient des sous-graphes partant de l'*ARG2* de *define-01*), il peut être supposé que chaque nœud du sous-graphe définitionnel est utilisé pour définir son $s-RAS$ correspondant. Le nœud racine peut alors être retiré et un arc étiqueté *define-01* partant de chaque nœud du graphe définitionnel vers *s-RAS* est ajouté. La proposition est que ces arcs capturent la même relation définitionnelle que ceux de la méthodologie proposée par Blondin Massé et al. et Vincent-Lamarre et al., tel qu'illustrée dans la figure 2.10 (Blondin Massé *et al.*, 2008; Vincent-Lamarre *et al.*, 2017).

Une fois que les nœuds racines de tous les graphes RAS ont été contournés, on peut simplement procéder à l'union de ces derniers pour obtenir la représentation par graphe du contenu d'un dictionnaire plongé en RAS. Ces étapes sont cruciales pour aligner le formalisme RAS avec la méthodologie précédemment décrite dans l'approche du *problème de l'ancrage des symboles*. En effet, si on retire tous les arcs qui ne sont pas étiquetés *define-01*, les mêmes méthodes d'analyses par graphe peuvent être utilisées. Ceci dit, ces arcs qui ont été retirés sont conservés en mémoire pour permettre leur analyse dans différentes structures du dictionnaire. Ainsi, les propriétés des relations sémantiques entre les mots des ensembles ancrages ou des différentes structures d'un dictionnaire peuvent être étudiées.

CHAPITRE 3
RÉDUCTIONS DE GRAPHES

Ce chapitre introduit le travail réalisé sur les réductions de graphes. La première section décrit les réductions implémentées dans le cadre de ce mémoire. La deuxième section introduit un algorithme pour réduire les graphes qui utilisera un ensemble confluent de réductions et un ensemble non-confluent de réductions.

3.1 Réductions de graphes

Une réduction de graphes est une opération qui permet de modifier un graphe dans l'optique de *réduire* sa taille, que ce soit en retirant un nœud ou un arc. Dans le cadre de ce mémoire, on souhaite réduire la taille des graphes en retirant des arcs ou des nœuds qui respectent certaines conditions, dans l'espoir de faciliter l'identification des TCCM. Il faut garder à l'esprit que la description précédente d'une réduction est une simplification puisque, comme nous le verrons sous peu, nous définissons formellement une réduction plutôt comme une fonction qui associe un graphe à une version réduite de ce même graphe. L'application de réductions aide aussi l'identification d'un TCCM par la construction d'une solution partielle : certaines des réductions ajoutent des nœuds dans un TCCM (on parle d'inclure dans la solution) alors que d'autres identifient des nœuds ne faisant pas partie d'un TCCM (on parle d'exclure de la solution).

3.1.1 Définition formelle d'une réduction

Soit $\mathcal{G}$ comme la classe finie de tous les graphes. Une fonction $T : \mathcal{G} \to \mathcal{G}$ est appelée *réduction*, si pour tous graphes $G, G' \in \mathcal{G}$, tels que $G = (V, A), G' = (V', A')$ et $G' = T(G)$, soit $|V'| < |V|$, ou soit $|V| = |V'|$ et $|A'| < |A|$. Dit autrement, une réduction associe à un graphe orienté un autre graphe orienté ayant strictement moins de nœuds ou si la quantité de nœuds est la même, ayant moins d'arcs.

Considérons une réduction T, un graphe G et un graphe réduit $G' \in \mathcal{G}$ tels que $G' = T(G)$, avec un sous-ensemble de nœuds $U \subseteq V(G)$ nommé solution partiellement construite par T. Nous pouvons dire que T préserve le TCCM de G par rapport à U si $U \cap V' = \emptyset$ et si pour tout TCCM U' de G'

nous avons que l'ensemble $U' \cup U$ est un TCCM de G.

Une autre manière de voir les réductions de graphe est de les considérer comme des relations binaires sur la classe finie des graphes possibles $\mathcal{G}$ (noté $\mathscr{R} \subseteq \mathcal{G} \times \mathcal{G}$). Donc, pour $G, G' \in \mathcal{G}$ (où $G' = T(G)$, soit un graphe transformé par une réduction), si on peut réduire G à G' par une réduction R, on peut écrire $(G, G') \in \mathscr{R}$. Pour une relation de réduction $\mathscr{R}$, un graphe $G \in \mathcal{G}$ est dit $\mathscr{R}$-irréductible (ou simplement irréductible) s'il n'existe pas de $G' \in \mathcal{G}$ tel que $(G, G') \in \mathscr{R}$.

3.1.2 Réductions connues

Plusieurs réductions pour le problème de calculer un TCCM ont été proposées dans la littérature (Levy et Low, 1988; Stege et Fellows, 1999; Lin et Jou, 2000; Lemaic, 2008; Xiao et Nagamochi, 2013; Fellows *et al.*, 2018; Kiesel et Schidler, 2023). Dans ce mémoire, nous nous limitons à la description de celles qui ont été implémentées.

Avant de décrire plus formellement chacune de ces réductions, il est utile de distinguer la transformation (la réduction en soi), qui est définie comme une fonction

$$T : \mathcal{G} \to \mathcal{G}$$

et la condition d'application de la réduction, qui peut être vérifiée par un *prédicat*.

Un prédicat est une fonction propositionnelle qui vérifie si ses arguments respectent une condition, suite à quoi une valeur de vérité est retournée. Plus formellement, un prédicat est une fonction où $P(x_1, x_2, ..., x_k)$ est vrai pour tout x_1, x_2, ..., x_k si et seulement si les arguments respectent une condition spécifique, comme n'être définis que par un seul mot ($deg^+_G(v) = 1$) ou ne pas être utilisés dans la définition d'autres mots ($deg^-_G(x_k) = 0$) Les prédicats dans les paragraphes suivants sont utilisés pour noter la vérification de la condition d'application des réductions.

Les premières réductions dans la littérature pour simplifier les instances de TCCM ont été proposées par Levy et Low (Levy et Low, 1988).

LOOP : Pour un nœud v, si $(v, v) \in A$ (si le nœud a une boucle), le nœud est inclus dans la solution. Noté LOOP(v) ou $\ell(G, v)$ sous forme de prédicat. Il s'agit en fait de la seule réduction qui inclut

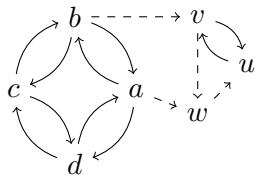


FIGURE 3.1 – Illustration d'arcs ne faisant pas partie d'une CFC et pouvant être retirés.

des nœuds dans la solution (qui indique qu'un nœud fait partie du TCCM).

OUT0 : Pour un nœud v, si $deg_G^-(v) = 0$, le nœud est exclu de la solution (qui indique qu'un nœud ne fait pas partie du TCCM). Notée OUT0(v). Cette réduction s'applique sur les mots qui ne sont pas utilisés pour en définir un autre.

IN0 : Pour un nœud v, si $deg_G^+(v) = 0$, le nœud est exclu de la solution. Notée IN0(v). Cette réduction s'applique sur les mots qui ne sont pas définis.

OUT1 : Pour un nœud v, si $deg_G^-(v) = 1$, le nœud est contourné (OUT1(v)). S'applique sur les mots ne définissant qu'un seul autre mot de contenu.

IN1 : Pour un nœud v, si $deg_G^+(v) = 1$, le nœud est contourné (IN1(v)). S'applique sur les mots défini par un seul autre mot de contenu.

Ces réductions précédentes sont relativement simples et ne permettent pas de réduire significativement les instances les plus complexes. Lin et Jou ont donc proposé d'agrandir l'ensemble de ces réductions en proposant 3 autres réductions (Lin et Jou, 2000).

PIE : La réduction PIE est reliée à l'idée qu'il suffit d'identifier le TCCM de chaque CFC de G et d'en faire l'union pour trouver le TCCM de G. Essentiellement, l'opération PIE retire tous les arcs bidirectionnels pour obtenir $G^{\rightarrow}$, puis elle identifie les arcs ne faisant partie d'aucune composante fortement connexe avant de les retirer. La figure 3.1 permet de visualiser de tels arcs.

CORE : Pour un nœud v, si $c(G, V_G(v))$ (soit si un nœud constitue une clique avec tous ses voisins), v est exclu de la solution et tous les autres nœuds de la clique sont exclus de la solution.

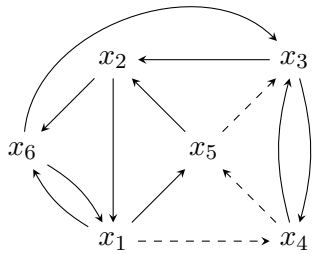


FIGURE 3.2 – Illustration du concept d'arc dominé. Les arcs en pointillés sont des arcs qui font parties de cycles non-minimaux : ils sont dominés et peuvent être retirés.

DOME : La première intuition pour comprendre cette réduction est l'idée qu'il suffit de considérer les *cycles minimaux* pour obtenir le TCCM de G. Tout cycle contenant un cycle plus minimal serait brisé si ce dernier était brisé. Ainsi, tout arc faisant partie d'un cycle non minimal peut être retiré. L'opération DOME se restreint cependant à la domination par des cycles de longueur 2. La figure 3.2, illustre un cas où PIE ne peut être appliquée sur les arcs pointillés mais où DOME s'applique.

Les cliques étant une structure de graphes avec des propriétés intéressantes pour les TCCM, Lemaic a proposé un nouvel ensemble de réductions poussant plus loin que CORE l'utilisation des ces structures (Lemaic, 2008).

OUTCLIQUE : Si tous les successeurs d'un nœud v forment une clique et que v est dépourvu de boucle, le nœud peut être contourné. L'opération est notée OUTCLIQUE(G, v) et s'applique si $\neg\ell(G, v)$ et $c(G, V_G^-(v))$. Ceci revient à dire que si tous les mots définis par un mot donné sont utilisés pour se définir, le *référent* du mot donné est inclus dans les successeurs. Le mot n'est donc pas nécessaire pour apprendre tous les autres mots du dictionnaire. La figure 3.3 illustre un cas d'application d'OUTCLIQUE.

INCLIQUE : Si tous les prédécesseurs d'un nœud donné forment une clique et que le nœud est dépourvu de boucle, le nœud est contourné. L'opération est notée INCLIQUE(G, v) et s'applique si $\neg\ell(G, v)$ et $c(G, V_G^+(v))$. La figure 3.4 illustre un cas d'application d'INCLIQUE.

DICLIQUE-2 : Étant donné un nœud u et un ensemble de nœuds S où $S = N_G(u) \cup \{u\}$, si le sous-graphe $G[S]$ contient deux cliques C_1 et C_2, tels que $C_1 \cap C_2 = \emptyset$ et $C_1 \cup C_2 = S$, le nœud

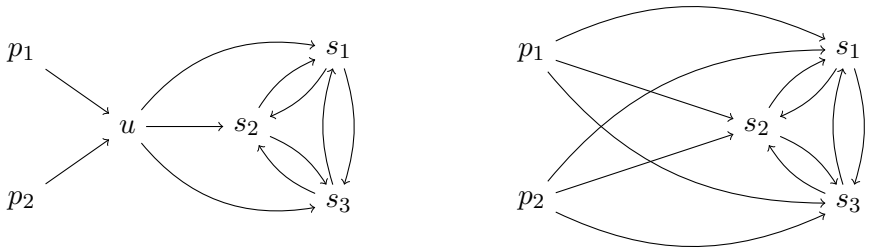


FIGURE 3.3 – OUTCLIQUE(G, u) s'applique dans le sous-graphe gauche puisque $N_G^+(u) = \{s_1, s_2, s_3\}$ forment une clique. On voit à droite le résultat de la réduction.

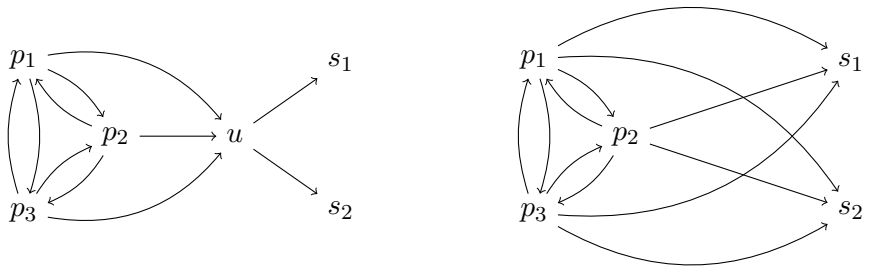


FIGURE 3.4 – INCLIQUE(G, u) s'applique dans le sous-graphe de gauche puisque $N_G^-(u) = \{p_1, p_2, p_3\}$ forment une clique. On voit à droite le résultat de la réduction.

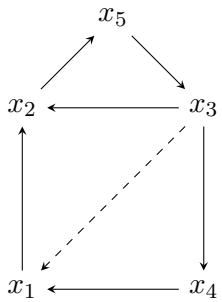


FIGURE 3.5 – Représentation du concept d'un arc dominé par un cycle de longueur arbitraire. DOMEPP, contrairement à DOME, va détecter la domination du cycle $\{x_5, x_3, x_2, x_5\}$ sur le cycle $\{x_5, x_3, x_1, x_2, x_5\}$ et va retirer l'arc (x_3, x_1).

peut être contourné. Pour vérifier cette condition, il faut obtenir le complément de $G^{\leftrightarrow} = (N, A^{\leftrightarrow})$. Si le complément du graphe symétrique est un graphe biparti, alors le nœud peut être contourné. On note DICLIQUE-2(G, u).

Les auteurs qui ont remporté la compétitions PACE 2022, centrée sur le problème du TCCM, ont introduit la réduction DOMEPP, généralisation de DOME. Alors que cette dernière ne considère que les chemins de longueur 2 pour la domination d'arcs, DOMEPP prend en compte les chemins de longueur arbitraire (Kiesel et Schidler, 2023).

DOMEPP : Tout arc faisant partie d'un cycle dominé par un de longueur plus courte est superflu pour briser la circularité d'un graphe. Pour vérifier si DOMEPP s'applique sur un arc (u, v), il faut commencer par retirer les arcs symétriques pour obtenir $G^{\rightarrow}$. Ensuite, il faut définir $S = N^{-}_{G^{\rightarrow}}(u) \setminus \{v\}$ et $P = N^{+}_{G^{\rightarrow}}(v) \setminus \{u\}$ avant de les retirer pour obtenir $G^{\rightarrow} = G^{\rightarrow}[V \setminus (S \cup P)]$. Proposons $p(u, v)$ comme le prédicat pour indiquer s'il existe un chemin de u à v ; si $\neg p(u, v)$ dans $G^{\rightarrow}$, alors (u, v) est un arc superflu et peut être retiré. Un cas d'un arc pouvant être retiré par DOMEPP est illustré dans la figure 3.5.

Avant de décrire la dernière réduction implémentée, il faut définir le *problème de la couverture par nœuds minimale* (CSM). Ce problème NP-complet, initialement décrit par Karp, cherche à identifier, dans un graphe non-orienté, un ensemble minimal de nœuds tel que tous les arcs soient *couverts* par les nœuds de l'ensemble (Karp, 2010). Pour qu'un arc soit couvert, il suffit de choisir un nœud auquel un arc est connecté. La figure 3.6 illustre le problème.

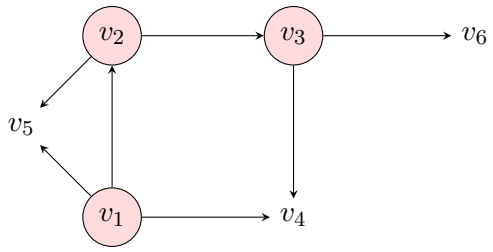


FIGURE 3.6 – Illustration du problème de la couverture par nœuds minimale (CSM). Les nœuds v_1, v_2 et v_3 forment une couverture minimale, car tous les arcs sont couverts.

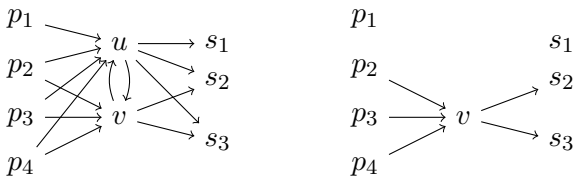


FIGURE 3.7 – Illustration de l'application de SUBSET. Effectivement, $\{p_2, p_3, p_4\} \subseteq \{p_1, p_2, p_3, p_4\}$ et $\{s_2, s_3\} \subseteq \{s_1, s_2, s_3\}$. SUBSET$(G, v)$ s'applique et u peut exclu de la solution, ce qui résulte en le sous-graphe de droite.

Une stratégie pertinente pour réduire les instances de TCCM est d'utiliser les réductions issues du problème de la CSM. Alors que certaines réductions peuvent être utilisées seulement si certaines conditions sont respectées, d'autres peuvent être utilisées plus librement (Kiesel et Schidler, 2023).

SUSBSET : Pour une paire de nœud v et u, SUBSET s'applique si et seulement si v et u sont reliés par une paire d'arcs symétriques, si $N_G^-(v) \subseteq N_G^-(u)$ et si $N +_G (v) \subseteq N_G^+(u)$. Si ces conditions sont vérifiées, alors u peut être exclu de la solution. Est notée SUBSET(G, v) et est illustrée dans la figure 3.7.

3.2 Algorithmes de réduction

Avant d'introduire l'algorithme de réduction pour les instances de TCCM, il faut définir au préalable ce qu'est la *confluence* ainsi que justifier le choix de ne pas retenir certaines réductions.

3.2.1 Confluence

La confluence, plus formellement connue sous le nom de *propriété Church-Rosser*, est une propriété que peut avoir un ensemble de *réductions*. Un ensemble confluent de réductions de graphe implique que l'ordre d'application de ces dernières n'a pas d'impact sur le graphe résultant ; on peut donc en toute sécurité appliquer autant que possible des réductions de cet ensemble sur un graphe donné et toujours arriver au même résultat, sans égard à l'ordre d'application des réductions. Cette propriété est d'intérêt dans l'étude des corrélats psycholinguistiques des structures des dictionnaires et de leur contenu : avoir les mêmes structures pour un dictionnaire donné permettrait entre autres une meilleure reproductibilité des résultats. Elle a aussi des implications algorithmiques, comme la parallélisation et l'accélération d'algorithmes dans l'analyse des graphes (Rosen, 1976).

Il peut alors être intéressant de se poser la question *quel est le plus grand ensemble de réductions confluent* ? La confluence d'ensembles utilisant les réductions implémentées décrites précédemment a déjà été démontrée dans des travaux antérieurs :

— {LOOP, IN0, OUT0, IN1, OUT1} par Levy et Low (Levy et Low, 1988)
— {LOOP, INDICLIQUE, OUTDICLIQUE} par Lemaic (Lemaic, 2008)
— {LOOP, IN0, OUT0, IN1, OUT1, INDICLIQUE, OUTDICLIQUE, PIE, CORE, SUSBET}, par Abdendi et al. (Abdenbi *et al.*, 2024)

On remarque alors que le dernier ensemble a la plus grande cardinalité. À notre connaissance, il n'existe pas de plus grand ensemble de réductions confluent connu pour le TCCM.

3.2.2 Réductions retenues

Nous proposons deux ensembles de réductions : un confluent et un non-confluent. Un premier critère dans ce choix est la propriété que des réductions peuvent être *subsumées* par d'autres, c'est-à-dire que l'intégralité de leurs cas d'application se retrouve dans une autre réduction. Pour nous aider dans ce choix, le tableau 3.1, librement traduit à partir d'un article de Kiesel et Schidler, est très utile (Kiesel et Schidler, 2023).

On voit que, dans le plus gros ensemble de réductions confluent, {IN0, OUT0, IN1, OUT1, CORE}

Nom de la réduction	Origine	Strictement subsumé par
LOOP	(Levy et Low, 1988)	-
IN0/IN1	(Levy et Low, 1988)	INDICLIQUE
OUT0/OUT1	(Levy et Low, 1988)	OUTDICLIQUE
INDICLIQUE	(Lemaic, 2008)	-
OUTDICLIQUE	(Lemaic, 2008)	-
DICLIQUE-2	(Lemaic, 2008)	-
DICLIQUE-3	(Lemaic, 2008)	-
PIE	(Lin et Jou, 2000)	ALLCYCLES
DOME	(Lin et Jou, 2000)	ALLCYCLES
DOMEPP	(Kiesel et Schidler, 2023)	ALLCYCLES
ALLCYCLES	(Kiesel et Schidler, 2023)	-
CORE	(Lin et Jou, 2000)	IN/OUTDICLIQUE
"Reduction 2"	(Stege et Fellows, 1999)	SUBSET (TCCM)
SUBSET (CS)	(Stege et Fellows, 1999)	SUBSET (TCCM)
SUBSET (TCCM)	(Kiesel et Schidler, 2023)	-
2FOLD	(Xiao et Nagamochi, 2013)	MANYFOLD (TCCM)
"Reduction 4"	(Stege et Fellows, 1999)	MANYFOLD (TCCM)
"Reduction 5"	(Stege et Fellows, 1999)	MANYFOLD (TCCM)
"Reduction 7.2"	(Stege et Fellows, 1999)	MANYFOLD (TCCM)
MANYFOLD (CS)	(Fellows *et al.*, 2018)	MANYFOLD (TCCM)
MANYFOLD (TCCM)	(Kiesel et Schidler, 2023)	-
4PATH (CS)	(Fellows *et al.*, 2018)	4PATH (TCCM)
4PATH (TCCM)	(Kiesel et Schidler, 2023)	-
UNCONFINED (CS)	(Fellows *et al.*, 2018)	UNCONFINED (TCCM)
UNCONFINED (TCCM)	(Kiesel et Schidler, 2023)	-
3EMPTY	(Stege et Fellows, 1999)	-
TWIN	(Xiao et Nagamochi, 2013)	3EMPTY + MANYFOLD (TCCM)
FUNNEL	(Xiao et Nagamochi, 2013)	SUBSET + MANYFOLD (TCCM)
DESK	(Xiao et Nagamochi, 2013)	-

TABLE 3.1 – Liste des réductions pour TCCM dans la littérature, adapté à partir de Kiesel et Schidler (Kiesel et Schidler, 2023). Il est à noter que la réduction ALLCYCLES implique l'énumération complète des cycles et est donc inutilisable dans le cadre du mémoire. Les réductions avec la mention CS réfèrent aux réductions pour le problèmes de la **couvertures par sommets** : celles indiquées TCCM sont leur adaptation le **l'identification du transversal de circuits de cardinalité minimale**.

sont subsumés par d'autres. L'ensemble de réductions confluent (noté $\mathcal{R}^c$) que nous retenons est donc {LOOP, INDICLIQUE, OUTDICLIQUE, SUBSET, PIE}.

Pour l'ensemble de réductions non-confluent, puisque DOME est subsumée par DOMEPP, ce n'est que cette dernière que nous considérons. Des tests préliminaires ont démontré que DICLIQUE-2 ne s'appliquait dans aucune instance créée à partir d'un dictionnaire et le coût computationnel était considérable, c'est pourquoi elle n'a pas été retenue. L'ensemble de réductions non-confluent que nous retenons est donc composé des réductions de l'ensemble confluent auxquelles on ajouterait seulement la réduction DOMEPP.

3.2.3 Algorithme

Nous sommes maintenant prêts à introduire l'algorithme 2, utilisé dans le mémoire pour réduire les instances de graphes de dictionnaire et de graphes RAS. Étant donné la confluence de $\mathcal{R}^c$, on peut en toute sécurité appliquer autant de fois que possible n'importe quelle réduction de l'ensemble et arriver à un même résultat unique (à isomorphisme près).

Algorithme 2 Réduire avec un ensemble $\mathcal{R}$ de réductions

Entrée : G : un digraphe, $\mathcal{R}$: un ensemble de réductions,
ρ : une fonction de priorité
Sortie : Un digraphe $\mathcal{R}$-*irréductible*

```
1: fonction RÉDUIRE(G, R, ρ)
2:     m ← max{ρ(R) | R ∈ R}
3:     retourner RÉDUIRE(G, R, ρ, m)
4: fonction RÉDUIRE(G, R, ρ, p)
5:     si p > 0 alors
6:         G' ← G
7:         faire
8:             G'' ← G'
9:             G'' ← RÉDUIRE(G'', R, ρ, p − 1)
10:            pour R ∈ R faire
11:                si ρ(R) = p alors
12:                    G'' ← R(G'')
13:        jusqu'à G' = G''
14:        retourner G'
15:    sinon
16:        retourner G
```

Pour des considérations d'efficacité algorithmique, il serait pertinent d'établir quel ordre d'application de ces réductions résulterait en l'exécution la plus rapide de l'algorithme (qui retourne un graphe *réduit* autant que possible) ou si cet ordre d'application était sans conséquence pour la vitesse d'exécution. Malheureusement, les expériences nécessaires pour une démonstration empirique ou théorique de cette question s'éloigne trop des portées en science cognitive de ce projet et l'ordre d'application des réductions a donc été déterminé de façon arbitraire. Nous proposons néanmoins ici quelques arguments pour justifier notre choix.

Si l'on suppose que l'ordre d'application peut avoir un effet sur la durée de l'exécution de l'algorithme, on pourrait proposer la stratégie d'appliquer autant que possible les réductions peu coûteuses avant d'appliquer les réductions plus coûteuses, dans l'espoir qu'un graphe préalablement réduit par les premières faciliterait le travail des dernières. Nous proposons LOOP, INCLIQUE, OUTCLIQUE et SUBSET comme *peu coûteuses* et PIE comme *coûteuse* pour les raisons suivantes.

Les premières vérifient des critères locaux basés sur des nœuds alors que PIE s'évalue en parcourant le graphe complet (détection des CFC et des arcs n'en faisant pas partie). Ensuite, les réductions peu coûteuses s'appliquent au fur et à mesure qu'elles sont *détectées*, alors que pour PIE une version *globale* est utilisée, c'est-à-dire que tous les cas d'application sont détectés avant d'être appliqués. Finalement, bien que la complexité algorithmique de INCLIQUE, OUTCLIQUE et SUBSET soit plus grande ($o(n^3)$) que celle de PIE ($\mathcal{O}(n+m)$) dans les pires cas, les premières pourraient s'exécuter très rapidement en pratique si les graphes considérés sont peu denses.

C'est pourquoi en des termes les plus simples, l'algorithme 2 peut être expliqué ainsi : appliquer autant que possible les réductions peu coûteuses, puis les réductions plus coûteuses ; ajouter un autre bloc d'exécution pour d'éventuelles réductions encore plus coûteuses.

Voyons plus formellement l'algorithme 2 qui est une abstraction du concept qui vient d'être exprimé. Il prend en entrée un ensemble $\mathcal{R}$ de réductions et une fonction de priorité ρ. Cette fonction, définie comme $\rho : \mathcal{R} \to \mathbb{N}_{>0}$, associe chaque réduction avec un entier positif (le plus petit est l'entier, la plus haute est la priorité). Par exemple, avec l'ensemble $\mathcal{R}^c$ et une réduction donnée R faisant partie de l'ensemble, nous avons pour ρ :

$$\rho_c(R) = \begin{cases} 1. & \text{si } R \in \{\text{LOOP, INCLIQUE, OUTCLIQUE, SUBSET}\}, \\ 2. & \text{si } R = \text{PIE.} \end{cases}$$

Pour l'ensemble de réductions non-confluent, nous avons :

$$\mathcal{R}^{nc} = \{\text{LOOP, INCLIQUE, OUTCLIQUE, SUBSET, PIE, DOMEPP}\},$$

qui donne la fonction de priorité ρ suivante :

$$\rho_c(R) = \begin{cases} 1. & \text{si } R \in \{\text{LOOP, INCLIQUE, OUTCLIQUE, SUBSET}\}, \\ 2. & \text{si } R = \text{PIE,} \\ 3. & \text{si } R = \text{DOMEPP} \end{cases}$$

CHAPITRE 4
EXPÉRIENCES

Ce chapitre débute par une description du jeu de données utilisé puis couvre les principales expériences réalisées dans le cadre de ce mémoire : la création de graphes à partir de dictionnaires français et anglais, la création de graphes RAS à partir des dictionnaires anglais, et la réduction subséquente de tous ces différents graphes. L'analyse et l'exploration des *ensembles d'ancrages minimaux*, de même que l'étude des effets de l'algorithme de réduction sur l'identification d'un TCCM, sont laissées pour des travaux futurs.

4.1 Jeu de données

Cette section introduit le jeu de données utilisé pour les expériences, composé de huit dictionnaires anglais et deux dictionnaires français. Les dictionnaires anglais sont utilisées dans les expériences décrites dans les sous-sections 4.2 et 4.3. Le contenu des dictionnaires français n'est pas plongé dans un formalisme sémantique et est donc uniquement étudié à l'aide de la méthodologie proposée par Vincent-Lamarre et al., décrite dans la sous-section 4.2 (Vincent-Lamarre *et al.*, 2017).

4.1.1 Dictionnaires anglais

Les dictionnaires anglais sont les plus nombreux et les plus utilisés de ce mémoire. Deux de ces dictionnaires, le *Longman Dictionary of Contemporary English* (LDOCE) (Procter, 1978) et le *Cambridge International Dictionary of English* (CIDE) (Procter, 1995), sont construits en utilisant un *vocabulaire de contrôle*, c'est-à-dire que les mots dans ces dictionnaires sont définis en utilisant le plus petit ensemble de mots possible. Tous deux sont des dictionnaires pour usagers avancés de l'anglais. Le CIDE a par contre la particularité d'être basé sur le *Cambridge corpus*, un corpus activement maintenu par la *Cambridge University Press* et uniquement basé sur des données linguistiques réelles (données linguistiques produites par des humains dans des contextes naturels, pas par des modèles). Le troisième dictionnaire est la onzième édition du *Merriam-Webster's Collegiate Dictionnary* (MWC) (Merriam-Webster, 2003).

Les quatre prochains dictionnaires proviennent de *Wordsmyth*, un projet éducatif en linguistique

(Wordsmyth, 2017). Le premier de ces dictionnaires est le *Wordsmyth Educational Dictionary-Thesaurus* (WEDT), développé en 1980 pour des usagers avancés de l'anglais. Ensuite vient le *Wordsmyth Learner's Dictionary-Thesaurus* (WLDT) qui est développé pour les adultes débutants. Les deux autres dictionnaires du projet éducatif, le *Wordsmyth Children's Dictionary-Thesaurus* (WCDT) et le *Wordsmyth Illustrated Learner's Dictionary-thesaurus* (WILD), ont eux été conçus pour les enfants. Le dernier dictionnaire anglais considéré est *Wordnet* (WN) (Fellbaum, 1998). Techniquement, il s'agit d'un *réseau sémantique* (un ensemble de mots annoté avec des relations sémantiques) duquel une structure de données sous forme de dictionnaire a été extraite. WN organise les mots en groupe de synonymes, nommés *synsets*. En plus de fournir des définitions aux mots, WN propose des relations sémantiques qui les relient, comme l'hyponymie et l'hyperonymie. La première est le lien que forme un mot spécifique (un hyponyme) avec un mot général (hyperonyme) et l'hyperonymie représente la relation inverse, propriétés utiles pour l'organisation de taxonomies et d'ontologies. Une autre paire de relations représentée dans WN est la méronymie et l'holonymie, où la première signifie qu'un mot forme une partie d'un autre mot qui constitue le tout : par exemple, la roue est méronyme d'une voiture, et la voiture holonyme de la roue.

4.1.2 Dictionnaires français

Deux dictionnaires français ont été considérés dans notre étude, le **Wiktionaire** et le **Trésor de la langue française informatisé** (TLFI). Le premier des deux a la particularité d'être un dictionnaire collaboratif ouvert et tout utilisateur qui respecte les conditions d'utilisations peut contribuer. Le Wiktionnaire, bien que centré autour du français, a une composante multilingue : on retrouve du contenu d'autres langues, souvent traduit de ou vers le français. Ce dictionnaire contient aussi beaucoup d'informations quant aux différents régionalismes du français. Ces particularités font du Wiktionaire un outil intéressant en linguistique computationnelle : un ensemble de mots, continuellement agrandi par les utilisateurs en fonction des néologismes et de l'usage courant, rendu disponible pour tous sous forme digitale.

Alors que le Wiktionaire peut contenir du langage familier, voir de l'argot (*slang*), le TLFI est le dictionnaire définitif pour ce qui a trait au français soutenu. Le *Trésor de la langue française* est un dictionnaire imprimé en plusieurs volumes entre 1971 et 2004, avec plus de 270 000 définitions et plus de 430 000 exemples, souvent tirés des plus grandes oeuvres littéraires de la francophonie. La

numérisation et mise en accès-libre subséquente ont donné le TLFI, opération conjointe du CNRS (Centre national de la recherche scientifique français) et de l'Université de Lorraine.

4.2 Expérience 1 : création et structures des *dico*

La première expérience présentée dans le mémoire est une reproduction avec raffinements méthodologiques de Vincent-Lamarre et al. (Vincent-Lamarre *et al.*, 2016). Il était pertinent de répliquer cette méthodologie pour trois raisons : la comparaison entre la représentation des dictionnaires comme graphes et la représentation des dictionnaires comme graphes RAS, la comparaison des structures de dictionnaires anglais avec la structure de dictionnaires français, et finalement, pour tirer avantage des nouvelles réductions de l'expérience 4.4 dans l'espoir d'obtenir les TCCM encore inaccessibles lors de travaux futurs.

4.2.1 Méthodologie

La première étape pour représenter le contenu d'un dictionnaire digital en un graphe est de transposer ledit contenu en une structure de données utilisable. C'est lors de cette étape initiale du prétraitement que les enjeux de polysémie et de désambiguïsation du sens des mots sont considérés. Bien que dans la conclusion de ce mémoire, une solution pour créer des dictionnaires *désambiguïsés* est proposée, nous n'avons pas eu le choix que de reprendre les approximations suivantes de Vincent-Lamarre et al., : ne conserver que la première définition de chaque mot et supposer que c'est le sens de cette dernière définition qui est utilisée à chaque fois que le mot se retrouve dans la définition d'un autre, en plus de ne considérer que les mots de contenu (Vincent-Lamarre *et al.*, 2016). Une structure de données riche et flexible est produite en un premier temps, qui contient une liste de tous les mots définis avec certains de leurs attributs, suivi d'une liste de toutes les définitions retenues. Pour chaque définition et pour les mots qui s'y trouvent, nous avons, entre autres, des attributs comme le lemme et la classe grammaticale. Le lemme est particulièrement important puisqu'il est utilisé pour reconnaître le symbole des verbes, noms et adjectifs lorsque conjugués dans une définition. La classe grammaticale est particulièrement d'intérêt pour des analyses psycholinguistiques lors de travaux futurs. Pour les fins de ce mémoire cependant, une représentation moins riche, qui ne contient qu'une liste des mots (les nœuds) et une liste d'arcs (les relations définitionnelles), est extraite puis utilisée dans la prochaine étape. Tout le code pour cette étape est implémenté en Python à l'aide de la bibliothèque Spacy et est utilisable au travers d'un outil en ligne de commande (Honnibal *et al.*,

2020). Ce pré-traitement, même pour les dictionnaires les plus gros, ne prend que quelques minutes lorsqu'il est exécuté sur un ordinateur personnel.

La deuxième étape dans la méthodologie est l'extraction des structures d'un dictionnaire, définies dans la sous-section 1.3.3. Il faut cependant noter que, dans le cadre du mémoire, le *noyau* a une particularité supplémentaire : en plus des mots non-définissants (qui ne sont pas utilisés pour en définir un autre), les mots non-définis sont récursivement retirés. Il faut rappeler que la récursion est ici un critère important : retirer des mots non-définissants ou non-définis d'un dictionnaire peut résulter en de nouveaux non-définis ou non-définissants. L'extraction des CFC (pour identifier le cœur et les satellites) est faite avec une implémentation de l'algorithme de Tarjan (Tarjan, 1972). Tout le code pour l'extraction des structures est implémenté en C++ et utilise en entrée la version minimaliste de la structure de données décrite dans le précédent paragraphe. Encore une fois, ce travail peut être exécuté sur un ordinateur personnel puisqu'il ne prend que quelques secondes par dictionnaire.

4.2.2 Structures des dictionnaires anglais

Voyons en un premier temps les métriques des structures de dictionnaires anglais. Les résultats se trouvent dans le tableau 4.1 et sont illustrés dans la figure 4.1. L'observation la plus frappante est que moins est retiré par le processus de *kernelisation* dans les dictionnaires pour enfants ou apprenants, surtout lorsque comparés aux dictionnaires pour adultes et usagers avancés de l'anglais. L'autre remarque initiale est la taille des cœurs : tous les dictionnaires présentent une CFC principale qui constitue à elle seule la quasi-totalité des noyaux. De plus, il est à noter que les satellites sont généralement de très petite taille.

On pourrait ensuite vouloir étudier la distribution des mots non-définissants selon le niveau de profondeur. Une évaluation qualitative des dictionnaires suggère qu'on retrouve des mots dans les 4 premières couches de récursion et une quantité négligeable dans le reste (lorsqu'on évalue individuellement la quantité de mots à chaque couche subséquente). C'est pourquoi dans la figure 4.2 on retrouve 5 classes pour étudier la distribution des mots non-définissants. On remarque qu'en général plus du trois quarts des mots non-définissants le sont initialement et que plus on avance, moins ils sont nombreux.

	WordNet	MerWeb	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
Taille initiale	146703 / 100%	97715 / 100%	52794 / 100%	35834 / 100%	21322 / 100%	12579 / 100%	4687 / 100%	3366 / 100%
Taille reste	139913 / 95.37%	89007 / 91.09%	46070 / 87.26%	33804 / 94.33%	19485 / 91.38%	9733 / 77.37%	3476 / 74.16%	1963 / 58.32%
Taille noyau	6790 / 4.63%	8708 / 8.91%	6724 / 12.74%	2030 / 5.67%	1837 / 8.62%	2846 / 22.63%	1211 / 25.84%	1403 / 41.68%
Taille coeur	5484 / 3.74%	6548 / 6.70%	5150 / 9.75%	1579 / 4.41%	1636 / 7.67%	2370 / 18.84%	1029 / 21.95%	1341 / 39.84%
Taille satellites	1306 / 0.89%	2160 / 2.21%	1574 / 2.98%	451 / 1.26%	201 / 0.94%	476 / 3.78%	182 / 3.88%	62 / 1.84%
Non-définis	81 (0)	299 (1)	40 (0)	21 (0)	14 (0)	5 (1)	2 (0)	0 (0)
Non-définissants	138700 (13)	86186 (14)	44672 (10)	33518 (6)	19092 (5)	9374 (10)	3363 (7)	1917 (7)
Qté arcs	822137	534363	299155	199581	127559	71628	23300	30499
Qté arcs noyau	34502	49378	37505	11370	12542	17070	5842	12063

TABLE 4.1 – Métrique des graphes dicos, dont la quantité absolue et proportionnelle de mots (taille) en fonction de la structure. La taille du reste correspond à la quantité de mots retirés pour obtenir le noyau. La taille des satellites réfère à la somme des mots dans l'ensemble des satellites. Pour les rangées *Non-définis* et *Non-définissants*, la valeur entre parenthèses représente le plus profond niveau de récursion lors de la *kernelisation*.

Finalement, nous nous intéressons aux mots les plus utilisés pour en définir d'autres dans les dictionnaires respectifs. Ces mots se trouvent dans le tableau 4.2. On remarque dans le top 10 de Wordnet plusieurs mots dans le champs lexical de la botanique : *genus*, *flower*, *leaf* *plant* et *tree*. Les autres dictionnaires anglais utilisent généralement les mêmes mots, comme *person*, *people* et *use*. Il est à noter que l'étude des structures lexicographiques et des corrélats psycholinguistiques pourraient être poussées beaucoup plus loin et en ce sens un des objectifs du présent mémoire est de rendre disponibles à des psycholinguistes de telles données. Par exemple, on pourrait se demander la liste des mots communs aux 8 dictionnaires anglais qui sont aussi dans la liste des 50 mots les plus utilisés pour en définir d'autres et on obtiendrait : *body*, *large*, *long*, *make*, *part*, *person*, *place*, *small*, *time*, *use* et *water*.

4.2.3 Structures des dictionnaires français

Voyons maintenant les métriques pour les structures des dictionnaires français, retrouvés dans le tableau 4.3. Comme pour les dictionnaires anglais, une illustration de la proportion des mots dans chaque structure se trouve dans la figure 4.3 et une illustration de la distribution des mots non-définissants selon la profondeur de récursion à laquelle ils ont été identifiés. On remarque que les noyaux sont essentiellement composés d'une unique composante fortement connexe (le cœur), les autres étant de tailles négligeables lorsque prises individuellement. Le résultat le plus surprenant est

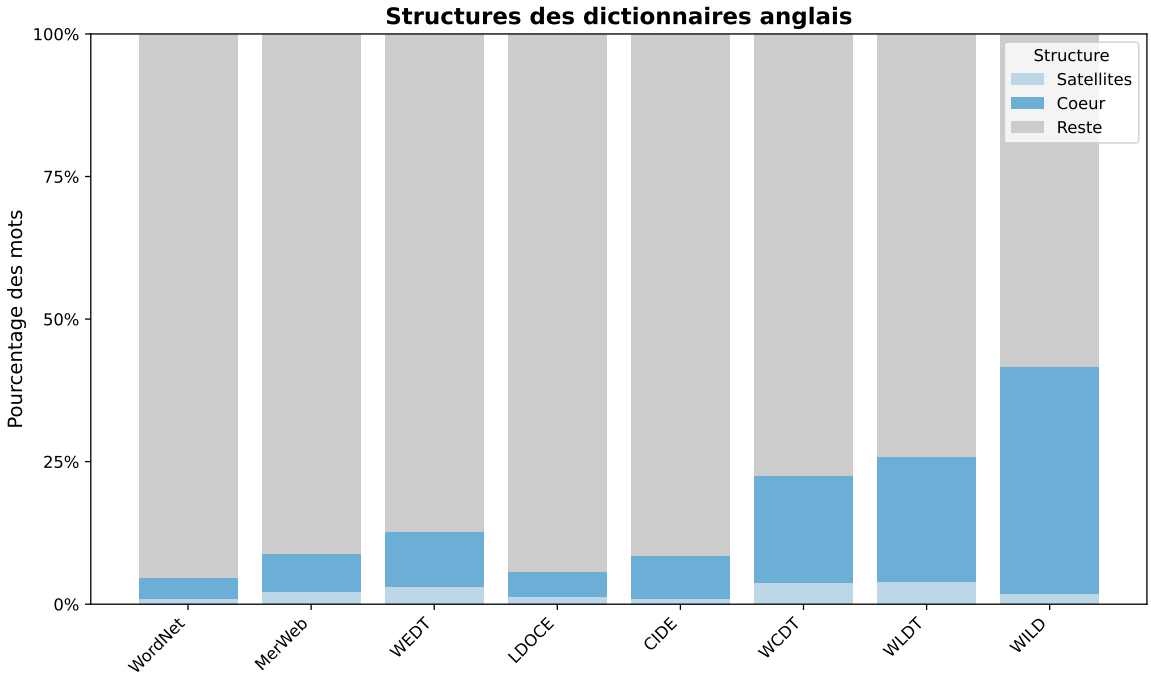


FIGURE 4.1 – Illustrations du pourcentage des mots présents dans chaque structure pour chaque dictionnaire anglais. Le *Reste* correspond à la -proportion des mots retirés par *kernelisation*. Le *cœur* est la proportion de mot dans la plus grosse CFC et *Satelittes* représente la proportion des mots dans toutes les autres CFC restantes. La proportion du *noyau* correspond à la somme des proportions du cœur et des satellites.

la grande taille du noyau du TLFI. On retrouve, pour les dictionnaires anglais, un noyau similaire pour le WILD ; or, le TLFI se veut un dictionnaire avancé et le WILD un dictionnaire illustré pour enfants. Les noyaux du Wiktionnaire et du TLFI sont de tailles absolues similaires mais le premier représente 11% du dictionnaire alors que le deuxième représente 44% du dictionnaire. Il est à noter qu'il n'y a pas de tel écart entre les dictionnaires anglais qui ne sont pas issus du projet éducatif *Wordsmyth*.

Encore une fois, comme pour les dictionnaires anglais, nous proposons la liste des 10 mots de contenus les plus utilisés pour en définir d'autres, retrouvés dans le tableau 4.4.

WordNet	MerWeb	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
use (8478)	use (5836)	use (5863)	use (3541)	esp (2856)	use (1218)	use (455)	make (653)
small (5502)	especially (5604)	combine (4989)	make (3266)	use (2126)	make (1057)	make (408)	people (599)
genus (5352)	relate (4411)	use (3828)	people (2766)	person (1847)	other (775)	other (318)	thing (568)
flower (5351)	usually (4330)	make (2020)	especially (2340)	make (1800)	person (684)	person (314)	use (449)
large (4708)	make (2974)	small (1543)	very (1986)	small (1295)	small (532)	thing (310)	other (443)
make (4142)	form (2496)	like (1750)	small (1740)	other (978)	large (497)	people (267)	very (349)
leaf (3302)	call (2390)	other (1666)	do (1505)	part (973)	do (486)	do (233)	do (348)
plant (3280)	also (2335)	often (1203)	way (1337)	part (973)	people (456)	part (220)	often (344)
especially (3239)	act (2174)	person (1542)	piece (1287)	very (961)	part (442)	small (179)	country (303)
form (3209)	large (2118)	plant (870)	piece (1287)	place (323)	act (387)	place (177)	large (249)

TABLE 4.2 – Liste des 10 mots de contenu les plus utilisés pour en définir d'autres dans les dictionnaires anglais. La valeur entre paranthèses correspond à la quantité de définitions où le mot est utilisé.

	WIKI	TLFI
Taille initiale	169305 (100%)	48877 (100%)
Taille reste	149956 (88.57%)	26989 (55.22%)
Taille noyau	19349 (11.43%)	21888 (44.78%)
Taille coeur	17950 (10.60%)	21592 (44.18%)
Taille satellites	1399 (0.83%)	296 (0.61%)
Non-définis	157 (1)	14 (0)
Non-définissants	149415 (8)	26860 (5)
Qté arcs	1098057	1044125
Qté arcs noyau	302937	590424

TABLE 4.3 – Métriques des dictionnaires français. WIKI correspond au Wiktionnaire et TLFI au Trésor de la langue française informatisé.

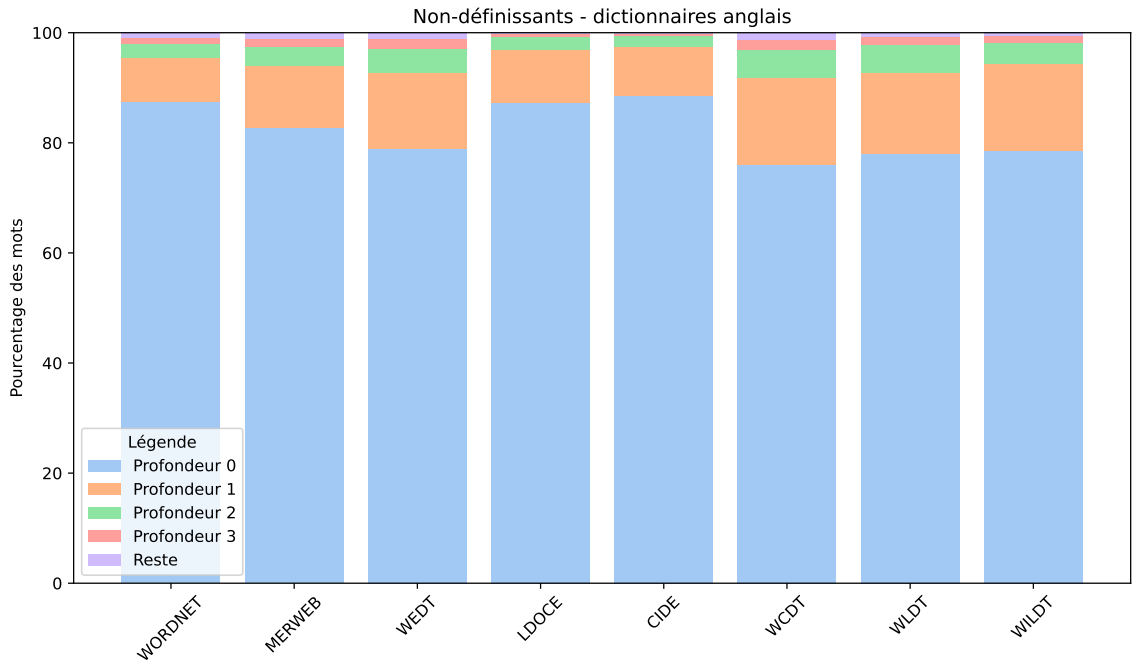


FIGURE 4.2 – Illustration du pourcentage des mots selon 5 niveaux de profondeur dans les dictionnaires anglais. La profondeur correspond au niveau de récursion lors du retrait du mot. Le reste correspond à l'ensemble des mots retirés dans des couches subséquentes à la troisième.

4.3 Expérience 2 : création et structures des *dicoAMR*

Il est maintenant temps de décrire l'expérience du plongement du contenu des dictionnaires anglais en représentation abstraite du sens (RAS).

4.3.1 Méthodologie

La première étape est l'extraction, pour chaque dictionnaire dans un format texte léger, de chaque paire (mot défini, définition) . La deuxième étape est la génération de graphe RAS à partir des paires de la première étape. C'est lors de cette étape que le filtrage des graphes invalides est effectué. Il en résulte un fichier texte qui contient la représentation textuelle de chaque RAS retenu (soit des RAS valides qui ne causent pas de collisions entre les étiquettes des mots définis). Finalement, la troisième étape consiste en le contournement du nœud racine *define-01* de chaque graphe RAS suivi de leur union pour compléter le plongement du contenu de dictionnaires anglais comme graphes RAS.

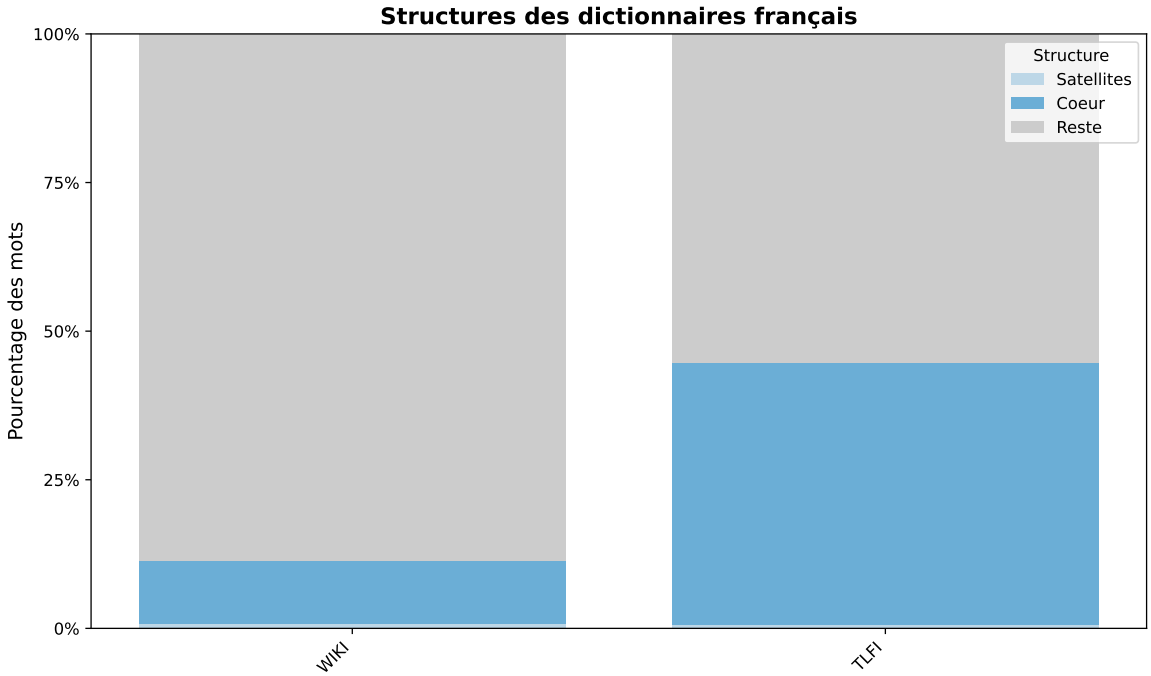


FIGURE 4.3 – Illustration du pourcentage des mots présents dans chaque structure pour chaque dictionnaire anglais.

Pour la première étape et troisième étape, un simple script en Python a suffi et pouvait s'exécuter en quelques secondes sur un ordinateur personnel. Le contournement du nœud racine lors de la troisième étape et les manipulations des graphes RAS lors de la deuxième et troisième étape ont été fait au travers de la bibliothèque Penman (Goodman, 2020). Cette bibliothèque permet de manipuler les graphes RAS comme des triplets Penman et constituent une solution rigoureusement testée pour la validation de ces graphes.

Pour la deuxième étape, de nombreuses bibliothèques ont dû être utilisées, notamment Amrlib (Jascob, 2023). Cette bibliothèque, facilement installable, offre des interfaces pour traduire des phrases de l'anglais vers les graphes (*sentence-to-graph* ou StoG) ou pour produire des phrases à partir de graphes RAS. Amrlib propose des modèle avec une architecture de *transformeurs* (Vaswani, 2017) . Effectivement, la RAS a connu un regain de popularité suite à l'introduction de cette architecture, offrant un nouvel état-de-l'art en matière de modèle *StoG*. Il existe des modèles StoG qui n'utilisent pas le mécanisme d'attention mais les performances ont été jugés trop faibles pour que la traduction automatique de RAS soit utilisable en dehors des contextes de recherche. L'auteur d'Armlib propose, sans fournir de données, que le modèle *Bart* de Facebook soit le plus performant pour le

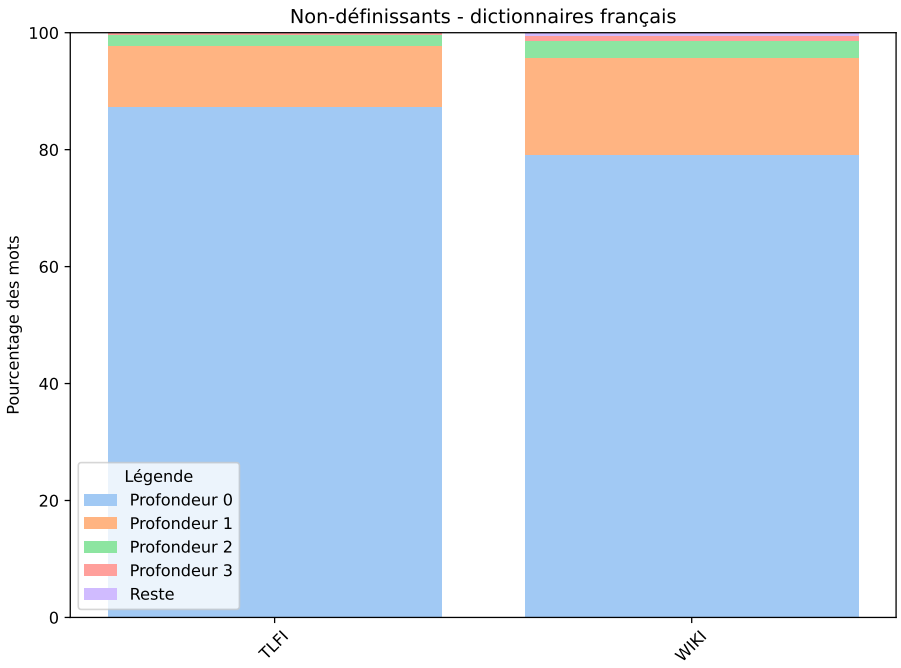


FIGURE 4.4 – Illustration du pourcentage des mots selon 5 niveaux de profondeur dans les dictionnaires français. La profondeur correspond au niveau de récursion lors du retrait du mot. Le reste correspond à l'ensemble des mots retirés dans des couches subséquentes à la troisième.

StoG de graphes RAS et c'est pourquoi c'est celui qui est utilisé dans ce mémoire. L'auteur propose aussi que les modèles de type encodeur-décodeur (connu sous le nom de *seq-to-seq* en anglais) soient plus performants que les modèles de types décodeurs uniquement (comme les GPTx et Llama).

C'est cette étape qui était la plus coûteuse computationnellement, étant donné les inférences faites à partir du modèle pré-entraîné. Une carte graphique personnelle avec 16GB de VRAM n'étant pas suffisante, la plateforme de calcul haute performance Narval, de Calcul Québec, a été utilisée. Puisque les inférences pouvaient être faites indépendamment, chaque dictionnaire a été séparé en 26 fichiers textes (un pour chaque lettre) et un script SLURM, dont l'exécution pouvait prendre quelques jours, a été implémenté pour traiter en parallèle les différents fichiers. Pour extraire les structures des dictionnaires plongés en RAS, le même code C++ de la section précédente a pu être utilisé.

WIKI	TLFI
relatif (27537)	faire (7907)
commune (22519)	personne (5853)
pluriel (21997)	chose (5710)
français (21715)	action (4288)
masculin (21569)	pouvoir (3726)
situer (21523)	forme (3724)
département (20924)	servir (3476)
nouveau (6386)	petit (3464)
action (5929)	ensemble (3339)
personne (5637)	donner (3090)

TABLE 4.4 – Liste des 10 mots de contenu les plus utilisés pour en définir d'autres dans les dictionnaires français.

4.3.2 Plongement des définitions

Les premiers résultats à considérer pour évaluer l'efficacité de la RAS dans la représentation du contenu des dictionnaires sont la quantité de définitions retenues. Effectivement, de nombreuses définitions ont été retirées, soit en étant invalides ou en causant des collisions. Le tableau 4.5 représente les métriques générales de la création des graphes RAS et la figure 4.5 illustre les proportions des définies perdues par dictionnaire.

Le plus flagrant constat est la faible proportion des définitions conservées dans les dictionnaires

	MerWeb	WordNet	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
Qté de définitions	301 240	206 185	86 949	80 086	49 787	22 563	6 900	4 709
Qté invalides initiale	98 349	98 443	28 548	17 030	10 484	3 910	739	796
Qté sauvées	43 632	32 195	13 721	9 538	7 564	1 936	400	231
Qté invalides finale	54 717	66 248	14 827	7 492	2 920	1 974	339	565
Filtrées par polysémie	141 255	46 486	25 137	39 257	24 296	7 822	1 600	1 085
Filtrées par collision	45 587	41 298	14 335	11 468	3 659	2 097	486	5
Qté finale	59 411	52 153	32 650	21 869	18 912	10 620	4 475	3 064

TABLE 4.5 – Métriques pour la création des graphes RAS

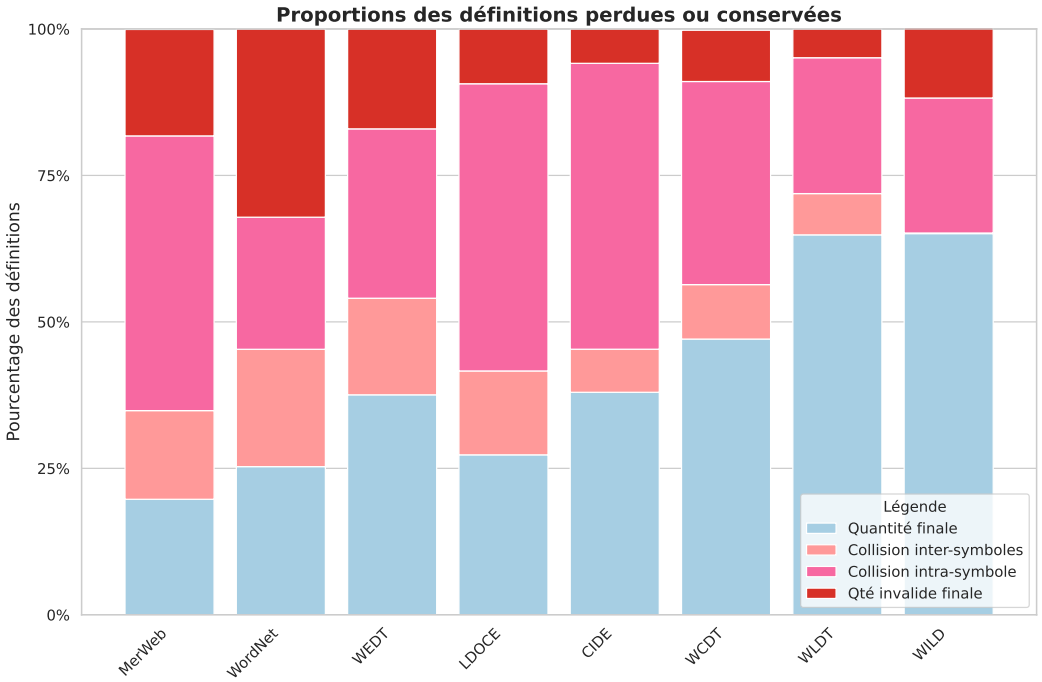


FIGURE 4.5 – Illustration des proportions de définitions *perdues* ou *conservées*. La *Qté invalide finale* représente la proportion des définitions perdues puisqu'il n'a pas été possible d'en produire un graphe RAS. La *Collision inter-symboles* représente les définitions perdues puisqu'elles engendrent des collisions entre les étiquettes des mots définis pour différents mots du dictionnaire. La *Collision intra-symbole* représente la proportion des définitions perdues par collisions entre les étiquettes des différentes définitions de mots polysémiques. La *Quantité finale* représente la proportion finale de graphes préservés.

non-issus du projet *Wordsmyth*. Ensuite, vient le constat que la principale cause du rejet d'une définition est la collision intra-symbole, soit les définitions perdues dans la gestion de la polysémie.

	WordNet	MerWeb	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
Taille initiale	52153 / 100%	59411 / 100%	32650 / 100%	21869 / 100%	18912 / 100%	10620 / 100%	4475 / 100%	3064 / 100%
Taille reste	45314 / 86.89%	51732 / 87.07%	27607 / 84.55%	19675 / 89.97%	16860 / 89.15%	8704 / 81.96%	3582 / 80.04%	2385 / 77.84%
Taille noyau	6839 / 13.11%	7679 / 12.93%	5043 / 15.45%	2194 / 10.03%	2052 / 10.85%	1916 / 18.04%	893 / 19.96%	679 / 22.16%
Taille coeur	6530 / 12.52%	6872 / 11.57%	4584 / 14.04%	1972 / 9.02%	1963 / 10.38%	1646 / 15.50%	781 / 17.45%	589 / 19.22%
Taille satellites	309 / 0.59%	807 / 1.36%	459 / 1.41%	222 / 1.02%	89 / 0.47%	270 / 2.54%	112 / 2.50%	90 / 2.94%
Qté arcs	311321	337376	190470	147319	144308	56102	21683	16579
Qté arcs noyaux	50620	58260	35308	16058	17459	10295	4306	3728

TABLE 4.6 – Quantité absolue et proportionnelle de mots (taille) en fonction de la structure, pour les dictionnaires plongés en RAS.

4.3.3 Structure d'un dictionnaire RAS

Le tableau 4.6 contient les métriques des structures des dictionnaires anglais plongés en RAS. La figure 4.6 est une illustration de certaines de ces métriques. On remarque une troisième fois que le noyau est essentiellement composé d'une unique composante fortement connexe, le cœur.

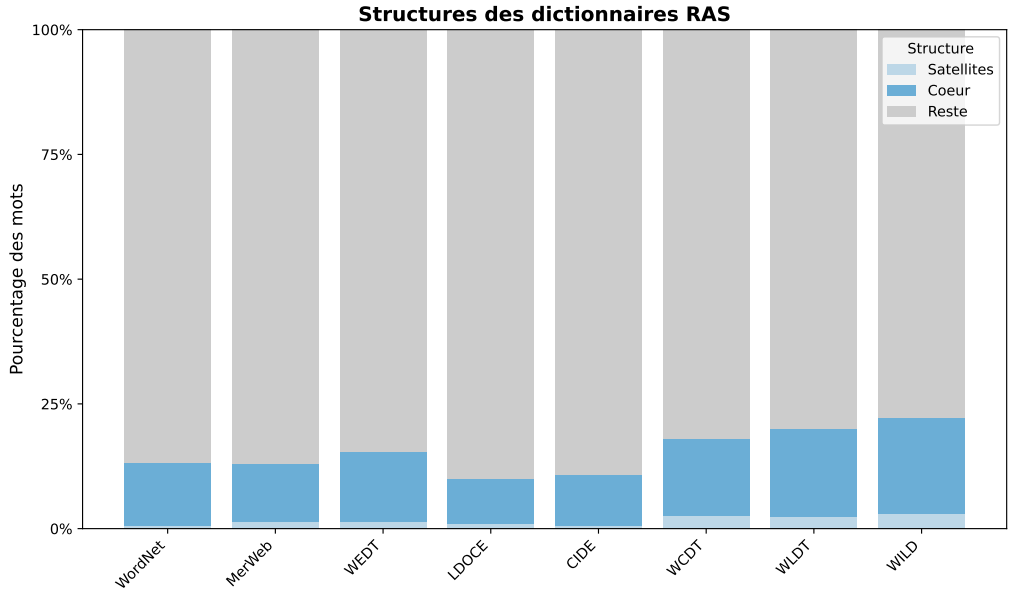


FIGURE 4.6 – Illustration des proportions de mots selon la structure dans un dictionnaire anglais plongé en RAS. Le reste correspond aux mots retirés pour obtenir le noyau.

4.4 Expérience 3 : réductions de digraphes

La dernière expérience décrite dans ce mémoire est la réduction des digraphes obtenus dans les deux premières expériences.

4.4.1 Méthodologie

Une représentation légère en format texte (liste de nœuds et liste d'arcs) est extraite pour les graphes des trois sources de données : les graphes de dictionnaires anglais, les graphes des dictionnaires français, et le plongement des dictionnaires anglais en RAS. Les graphes sont ensuite réduits en un premier temps en utilisant un ensemble confluent de réductions puis l'exercice est répété avec un ensemble non-confluent. Comme décrit dans la sous-section 3.2.3, l'algorithme utilisé applique autant que possible, une à une, les réductions de l'ensemble, de la moins coûteuse à la plus coûteuse. Tout le code a été implémenté en C++. Les temps d'exécution pour l'algorithme de réduction était dans la majorité des cas négligeables et tous ces résultats ont pu être obtenus à partir d'un ordinateur personnel.

4.4.2 Réductions des dictionnaires anglais

Le tableau 4.7 contient les résultats de l'algorithme utilisant un ensemble confluent de réductions alors que la table 4.8 contient les résultats pour l'ensemble non-confluent. Pour un aperçu du temps pris par chaque réduction dans l'ensemble confluent, voir la figure 4.7 et pour l'ensemble non-confluent, voir la figure 4.8. Pour l'ensemble non-confluent, la figure 4.14 supplémentaire est générée pour représenter le temps dédié à la réduction DOMEPP. Ce choix est justifié par le fait que cette réduction est significativement plus coûteuse que les autres. On remarque que pour les réductions confluentes, la réduction PIE est la plus coûteuse.

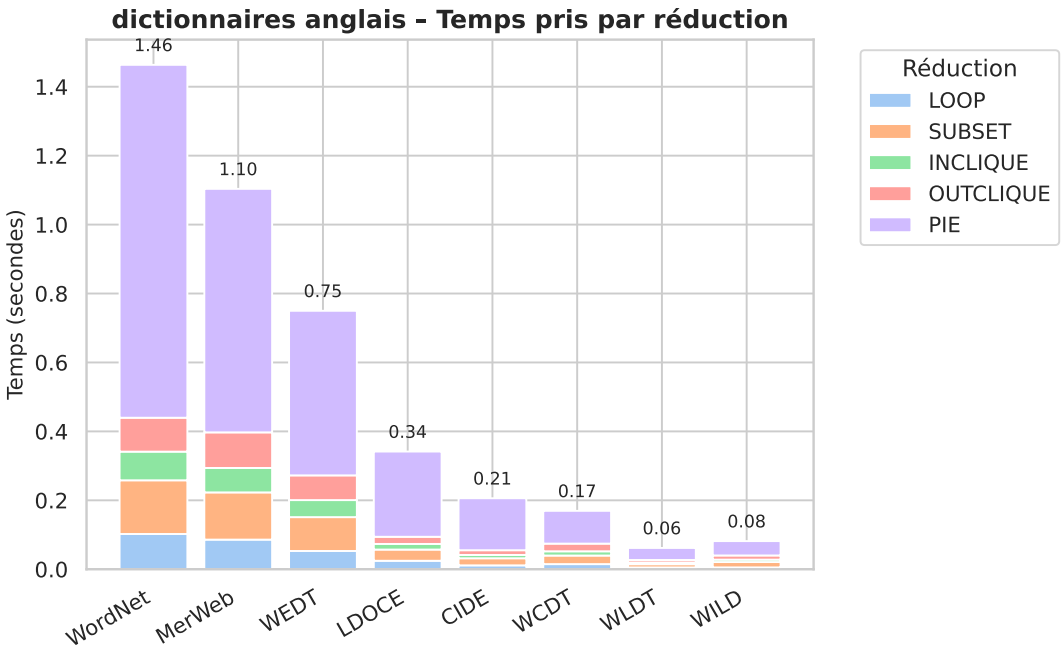


FIGURE 4.7 – Illustration du temps absolu pris selon la réduction d'un ensemble **confluent**, dans les dictionnaires anglais. Les valeurs au-dessus des barres représentent le temps total pour l'ensemble des réductions.

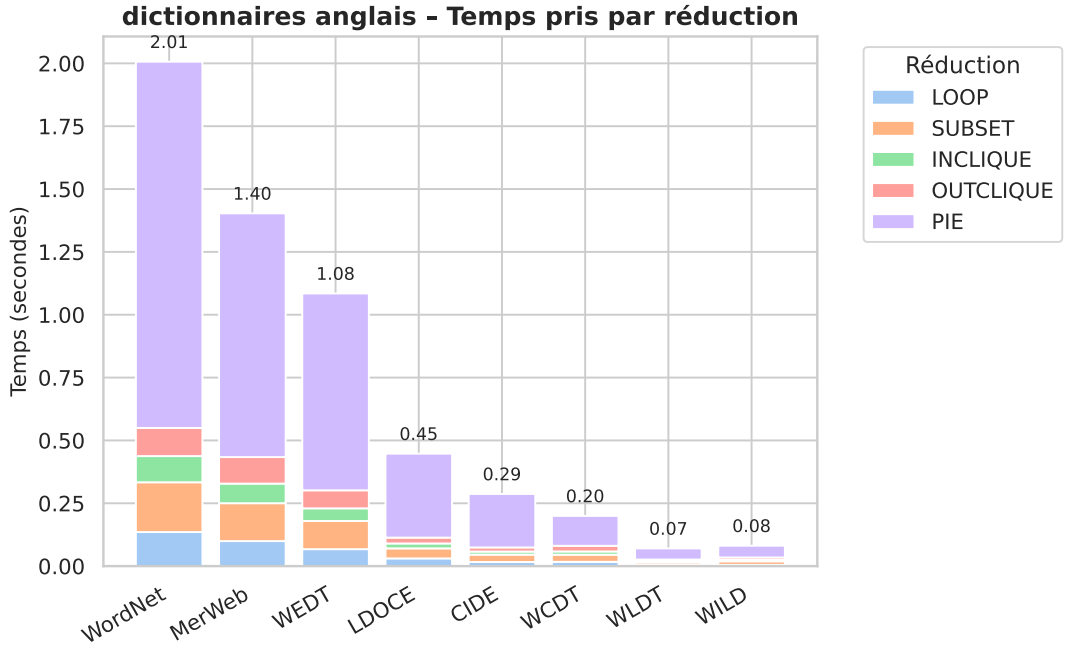


FIGURE 4.8 – Illustration du temps absolu pris selon la réduction d'un ensemble **non-confluent**, dans les dictionnaires anglais. Les valeurs au-dessus des barres représentent le temps total pris par l'ensemble des réductions.

	WordNet	MerWeb	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
Qté restants	2299	2196	1814	635	852	544	260	611
Qté inclus	432	1034	852	322	173	440	200	145
Qté exclus	143972	94485	50128	34877	20297	11595	4227	2610
Qté réductions	4727	6994	5394	1535	1065	2478	990	913
Qté LOOP	350 (7.40%)	748 (10.70%)	806 (14.95%)	300 (19.54%)	157 (14.74%)	434 (17.51%)	195 (19.70%)	144 (15.77%)
Qté INCLIQUE	243 (5.14%)	673 (9.62%)	273 (5.06%)	115 (7.49%)	59 (5.54%)	227 (9.16%)	118 (11.92%)	52 (5.69%)
Qté OUTCLIQUE	3897 (82.41%)	5084 (72.69%)	3820 (70.83%)	979 (63.80%)	767 (72.03%)	1638 (66.08%)	635 (64.14%)	595 (65.16%)
Qté SUBSET	1 (0.02%)	3 (0.04%)	6 (0.11%)	1 (0.07%)	2 (0.19%)	2 (0.08%)	3 (0.30%)	1 (0.11%)
Qté PIE	236 (4.99%)	482 (6.89%)	484 (8.97%)	140 (9.12%)	80 (7.51%)	176 (7.10%)	39 (3.94%)	121 (13.25%)
Qté arcs restants	17355	16728	13411	4251	7037	3664	1556	5449

TABLE 4.7 – Résultats de l'exécution de l'algorithme de réductions sur les des dictionnaires anglais réduits de façon confluente. Les trois premières rangées correspondent à des quantité de mots. Les autres rangées réfèrent à des quantités de réductions appliquées. Les valeurs entre parenthèses sont le pourcentage du total de réductions appliquées.

	WordNet	MerWeb	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
Qté restants	2254	2162	1769	616	845	518	255	602
Qté inclus	438	1037	859	326	174	444	200	145
Qté exclus	144011	94516	50166	34892	20303	11617	4232	2619
Qté réductions	5165	7444	5922	1737	1225	2770	1087	1246
Qté DOMEPP	393 (7.61%)	416 (5.59%)	474 (8.00%)	183 (10.54%)	153 (12.49%)	266 (9.60%)	92 (8.46%)	324 (26.00%)
Qté LOOP	355 (6.87%)	751 (10.09%)	813 (13.73%)	304 (17.50%)	158 (12.90%)	437 (15.78%)	195 (17.94%)	144 (11.56%)
Qté INCLIQUE	265 (5.13%)	690 (9.27%)	291 (4.91%)	124 (7.14%)	61 (4.98%)	236 (8.52%)	121 (11.13%)	57 (4.57%)
Qté OUTCLIQUE	3914 (75.79%)	5100 (68.50%)	3843 (64.88%)	984 (56.64%)	771 (62.94%)	1652 (59.64%)	636 (58.50%)	598 (47.99%)
Qté SUBSET	2 (0.04%)	3 (0.04%)	6 (0.10%)	1 (0.06%)	2 (0.16%)	3 (0.11%)	3 (0.28%)	1 (0.08%)
Qté PIE	236 (4.57%)	482 (6.47%)	493 (8.32%)	140 (8.06%)	80 (6.53%)	176 (6.35%)	39 (3.59%)	121 (9.71%)
Qté arcs restants	16389	16264	12789	4023	6878	3334	1460	5141

TABLE 4.8 – Métriques des dictionnaires anglais réduits de façon non-confluente. Les trois premières rangées correspondent à des quantités de mots. Les autres rangées réfèrent à des quantités de réductions appliquées. Les valeurs entre parenthèses sont le pourcentage du total des réductions appliquées.

4.4.3 Réductions des dictionnaires français

Les résultats obtenus pour les dictionnaires français sont présentés sous le même format : le tableau 4.9 représente les résultats de l'ensemble confluent et le tableau 4.10 les résultats de l'ensemble

non-confluent ; on retrouve trois figures, une pour chaque ensemble de la proportion du temps prise par chaque réduction (4.9, 4.10) puis une figure pour le temps pris par la réduction DOMEPP 4.11. Il faut noter que DOMEPP prend plusieurs heures à s'exécuter sur les dictionnaires français, avec TLFI qui s'élèvent à plus de 36 heures.

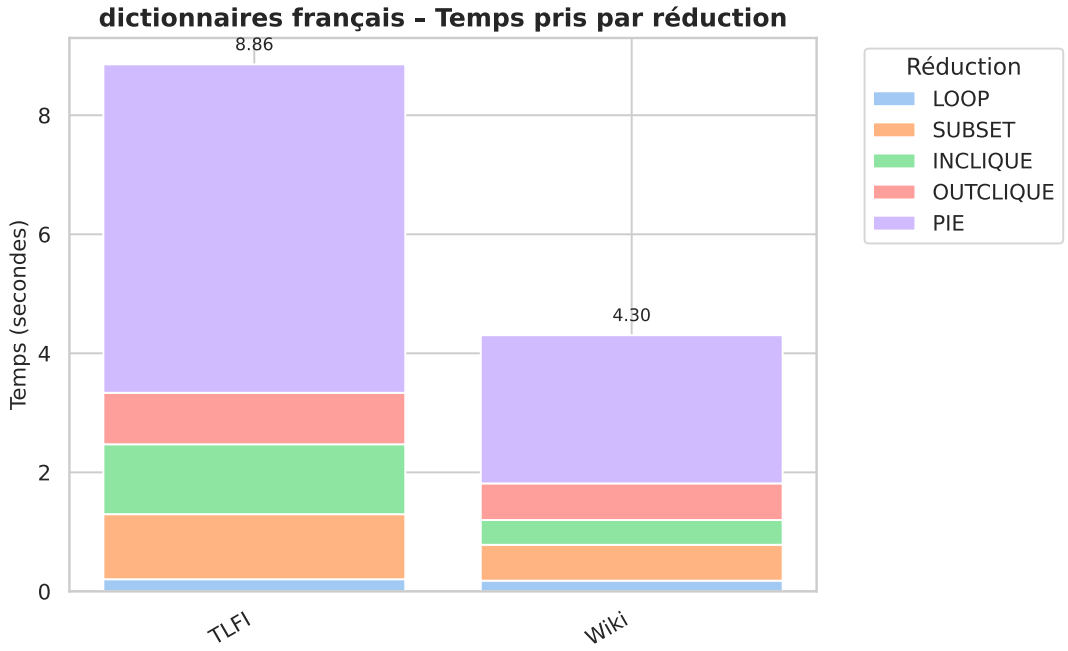


FIGURE 4.9 – Illustration du temps absolu pris selon la réduction d'un ensemble **confluent**, dans les dictionnaires français. Les valeurs au-dessus des barres représentent le temps total pris par l'ensemble des réductions.

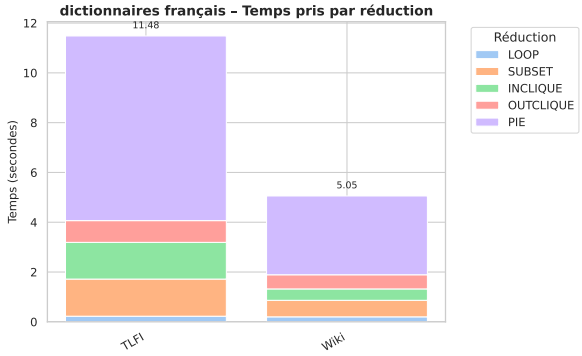


FIGURE 4.10 – Illustration du temps pris selon la réduction d'un ensemble **non-confluent**, dans les dictionnaires français. Les valeurs au-dessus des barres représentent le temps total pris par l'ensemble des réductions.

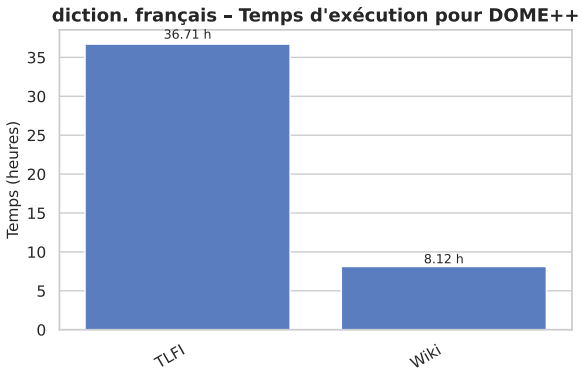


FIGURE 4.11 – Illustration du temps pris par la réduction DOMEPP selon le dictionnaire français. Noter que l'axe vertical est mesuré en heures.

	WIKI	TLFI
Qté restants	10193	15129
Qté inclus	1257	730
Qté exclus	157855	33018
Qté réductions	10088	7373
Qté LOOP	1096 (10,86%)	716 (9,71%)
Qté INCLIQUE	262 (2,60%)	164 (2,22%)
Qté OUTCLIQUE	7788 (77,20%)	5879 (79,74%)
Qté SUBSET	5 (0.05%)	0 (0.00%)
Qté PIE	932 (9,24%)	614 (8,33%)
Qté arcs restants	209757	590424

TABLE 4.9 – Résultats de l'exécution de l'algorithme de réduction sur les des dictionnaires français réduits de façon non-confluente. Les trois premières rangées correspondent à des quantité de mots. Les autres rangées réfèrent à des quantités de réductions appliquées. Les valeurs entre parenthèses sont le pourcentage du total de réductions appliquées.

	WIKI	TLFI
Qté restants	10123	15095
Qté inclus	1263	735
Qté exclus	157919	33047
Qté réductions	12490	12419
Qté DOMEPP	2332 (18,67%)	5012 (40,36%)
Qté LOOP	1102 (8,82%)	721 (5,81%)
Qté INCLIQUE	308 (2,47%)	176 (1,42%)
Qté OUTCLIQUE	7810 (62,53%)	5895 (47,47%)
Qté SUBSET	5 (0,04%)	0 (0,00%)
Qté PIE	932 (7,46%)	614 (4,94%)
Qté arcs restants	207186	582633

TABLE 4.10 – Résultats de l'exécution de l'algorithme de réduction sur les des dictionnaires français réduits de façon non-confluente. Les trois premières rangées correspondent à des quantité de mots. Les autres rangées réfèrent à des quantités de réductions appliquées. Les valeurs entre parenthèses sont le pourcentage du total de réductions appliquées.

4.4.4 Réductions des dictionnaires RAS

Les résultats obtenus pour les dictionnaires plongés en RAS sont présentés sous le même format : le tableau 4.11 représente les résultats de l'ensemble confluent et le tableau 4.12 les résultats de l'ensemble non-confluent ; on retrouve trois figures, une pour chaque ensemble de la proportion du temps prise par chaque réduction (4.12, 4.13) puis une figure pour le temps pris par la réduction DOMEPP 4.15.

	WordNet	MerWeb	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
Qté restants	2948	2846	1836	931	1067	374	95	216
Qté inclus	4697	6233	4465	1411	1643	1591	692	457
Qté exclus	44508	50332	26349	19527	16202	8655	3688	2391
Qté réductions	3969	4879	3386	1291	998	1545	812	463
Qté LOOP	258 (6.50%)	417 (8.55%)	360 (10.63%)	155 (12.01%)	93 (9.32%)	253 (16.38%)	133 (16.38%)	68 (14.68%)
Qté INCLIQUE	495 (12.47%)	929 (19.05%)	433 (12.79%)	147 (11.39%)	113 (11.32%)	205 (13.27%)	160 (19.70%)	35 (7.56%)
Qté OUTCLIQUE	3137 (79.03%)	3487 (71.46%)	2414 (71.31%)	961 (74.44%)	779 (78.06%)	1082 (70.02%)	504 (62.07%)	360 (77.75%)
Qté SUBSET	1 (0.03%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	2 (0.13%)	1 (0.12%)	0 (0%)
Qté PIE	78 (1.96%)	46 (0.94%)	179 (5.29%)	28 (2.17%)	13 (1.30%)	3 (0.19%)	14 (1.72%)	0 (0%)
Qté arcs restants	32277	33268	19147	8943	11940	3665	1149	1828

TABLE 4.11 – Métriques des dictionnaires anglais plongés en RAS réduits de façon confluente. Les trois premières rangées correspondent à des quantité de mots. Les autres rangées réfèrent à des quantités de réductions appliquées. Les valeurs entre parenthèses sont le pourcentage du total de réductions appliquées.

	WordNet	MerWeb	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
Qté restants	2928	2807	1822	915	1058	353	85	184
Qté inclus	4697	6234	4466	1412	1644	1594	693	466
Qté exclus	44528	50370	26362	19542	16210	8673	3697	2414
Qté réductions	4481	5421	3653	1488	1253	1721	879	827
Qté DOMEPP	492 (10.98%)	503 (9.28%)	474 (6.93%)	181 (12.16%)	246 (19.63%)	155 (9.01%)	57 (6.48%)	328 (39.66%)
Qté LOOP	258 (5.76%)	418 (7.71%)	361 (9.89%)	156 (10.48%)	93 (7.42%)	256 (14.88%)	134 (15.25%)	77 (9.31%)
Qté INCLIQUE	504 (11.25%)	945 (17.43%)	442 (12.10%)	155 (10.41%)	117 (9.34%)	220 (12.78%)	165 (18.77%)	53 (6.41%)
Qté OUTCLIQUE	3146 (70.23%)	3509 (64.72%)	2418 (66.19%)	968 (65.03%)	783 (62.47%)	1085 (63.03%)	508 (57.81%)	364 (44.01%)
Qté SUBSET	1 (0.02%)	0 (0%)	0 (0%)	0 (0%)	1 (0.08%)	2 (0.12%)	1 (0.11%)	0 (0%)
Qté PIE	78 (1.74%)	46 (0.85%)	179 (4.90%)	28 (1.88%)	13 (1.04%)	3 (0.17%)	14 (1.59%)	4 (0.48%)
Qté arcs restants	31766	32574	18849	8741	11660	3460	1076	1424

TABLE 4.12 – Métriques des dictionnaires anglais plongés en RAS réduits de façon non-confluente. Les trois premières rangées correspondent à des quantités de mots. Les autres rangées réfèrent à des quantités de réductions appliquées. Les valeurs entre parenthèses sont le pourcentage du total des réductions appliquées.

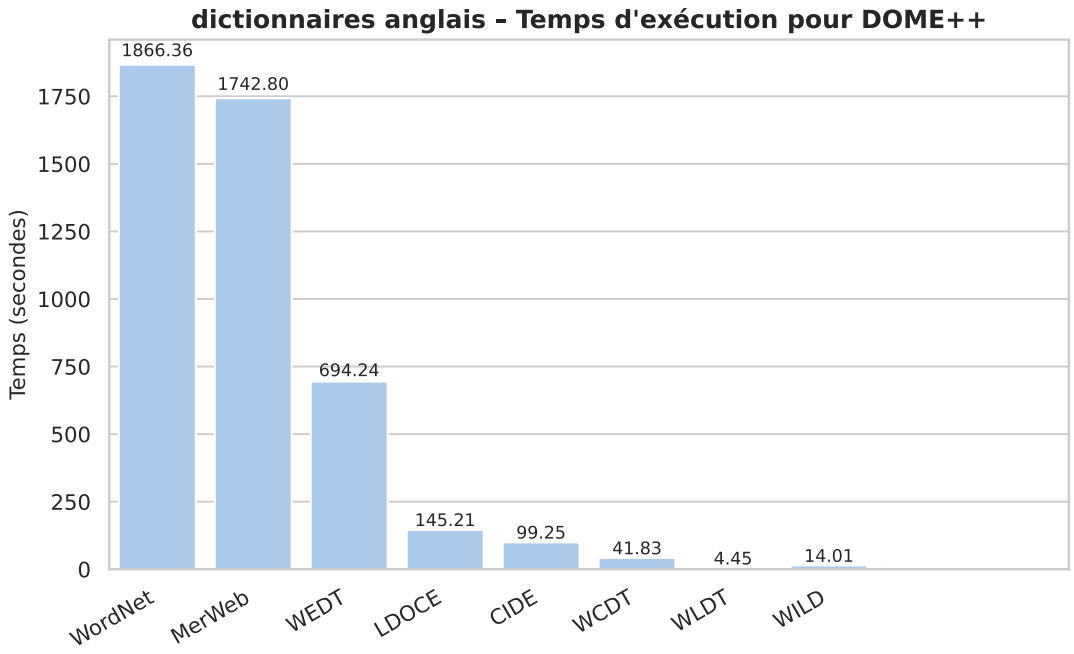


FIGURE 4.14 – Illustration du temps pris par la réduction DOMEPP selon le dictionnaire anglais. L'axe vertical est mesuré en secondes. Le temps d'exécution pour Wordnet est d'environ 30 minutes.

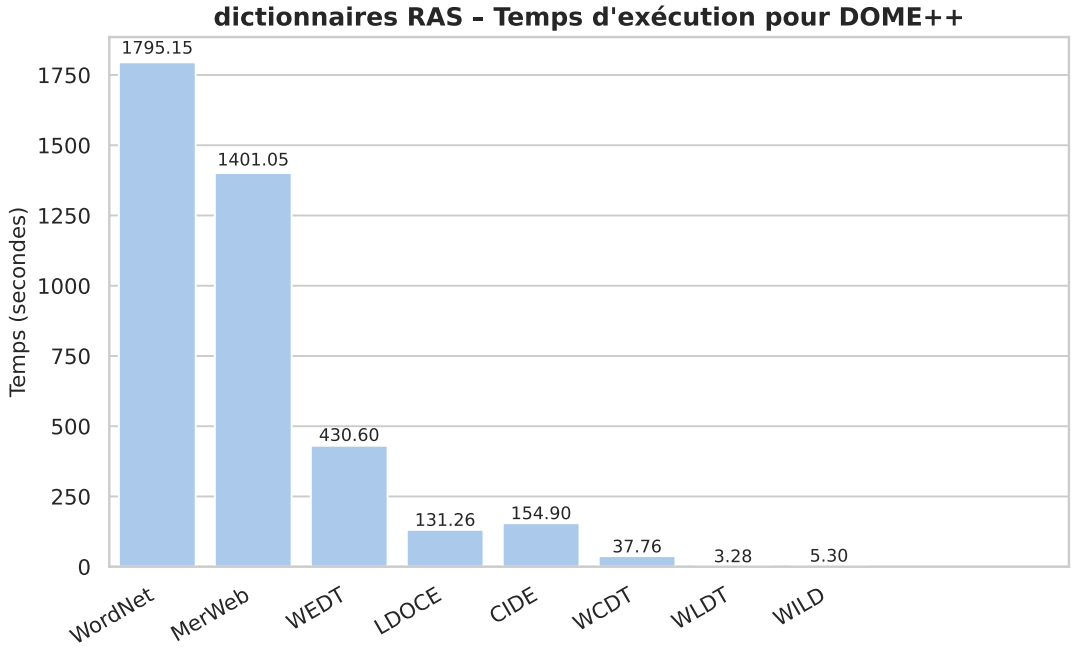


FIGURE 4.15 – Illustration du temps pris par la réduction DOMEPP selon le dictionnaire anglais plongé en RAS. L'axe vertical est mesuré en heures.

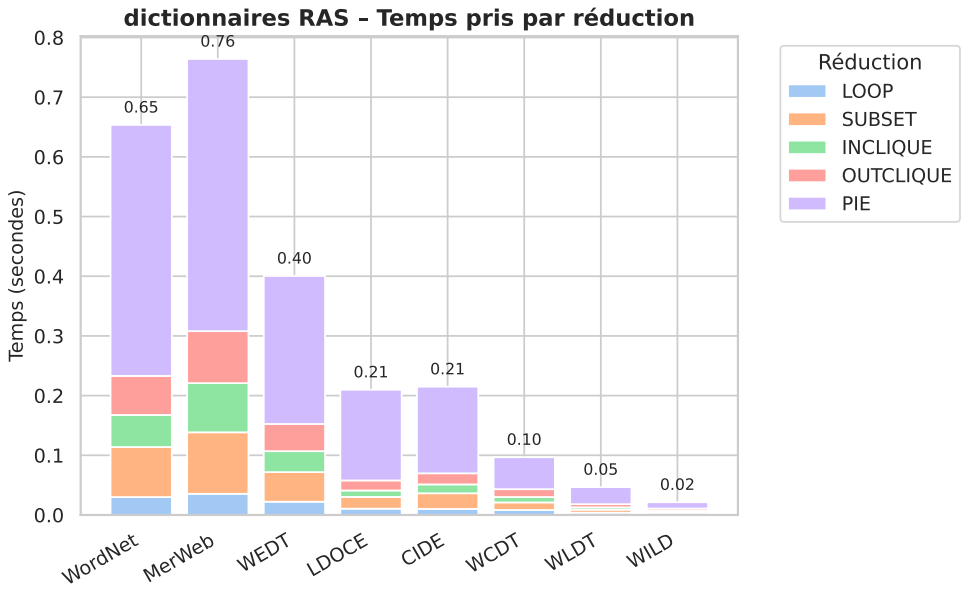


FIGURE 4.12 – Illustration du temps pris selon la réduction d'un ensemble **non-confluent**, dans les dictionnaires plongés en RAS. Les valeurs au-dessus des barres représentent le temps total en secondes pris par l'ensemble des réductions.

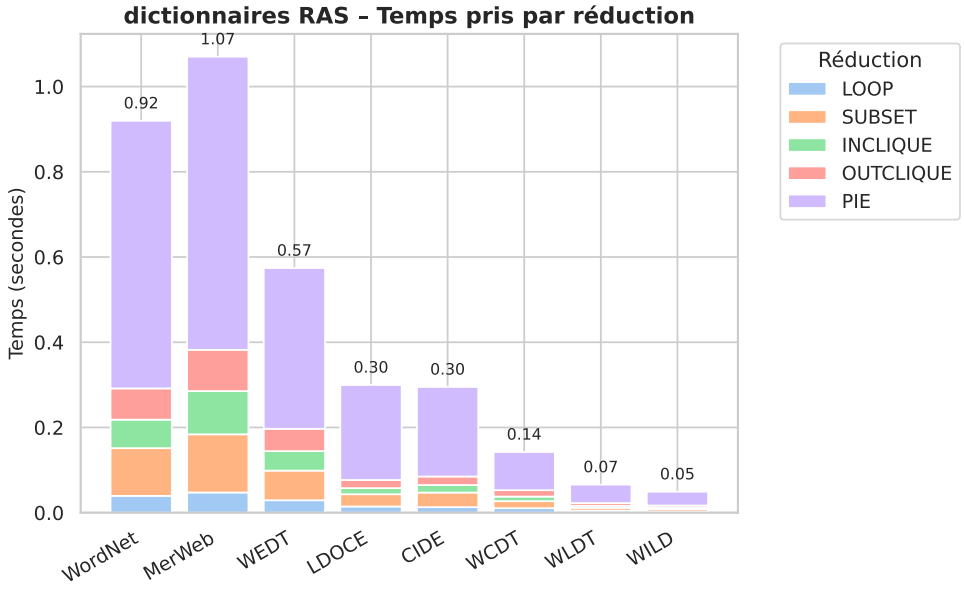


FIGURE 4.13 – Illustration du temps pris selon la réduction d'un ensemble **non-confluent**, dans les dictionnaires plongés en RAS. Les valeurs au-dessus des barres représentent le temps en secondes total pris par l'ensemble des réductions.

CHAPITRE 5
ANALYSES ET DISCUSSION

Ce chapitre couvre la discussion des résultats des différentes expériences décrites au chapitre 4. La première section s'intéresse à la principale contribution du mémoire : la représentation du contenu des dictionnaires comme graphes RAS. La deuxième section contient une analyse plus approfondie des résultats de l'algorithme de réduction et des implications pour des travaux futurs. Finalement, le chapitre termine sur une discussion des différences entre dictionnaires français et anglais.

5.1 Efficacité du plongement RAS

La première expérience ici considérée est le plongement du contenu des dictionnaires en RAS et son efficacité par rapport à la méthodologie originale de Vincent-Lamarre et al. (Vincent-Lamarre *et al.*, 2016). Ensuite, la section termine sur une discussion plus générale autour du processus de plongement.

5.1.1 Comparaison entre graphes de mots et graphes RAS

Le plus important critère pour évaluer l'applicabilité de la RAS était une comparaison de l'efficacité entre les dictionnaires anglais comme graphe et leur plongement en graphe RAS. L'efficacité sera ici définie comme la proportion des définitions retenues après la transformation d'un dictionnaire en graphe régulier ou graphe RAS. En effet, dans les deux cas, des définitions étaient perdues : les graphes réguliers, en ne conservant *que les premières définitions des mots polysémiques* et les graphes RAS en ne conservant que les *RAS valides qui n'engendrent pas de collisions intra-symbole ou inter-symboles*. Pour faciliter cette comparaison, voyons la figure 5.1 produite à partir des données du précédent chapitre, qui illustre les proportions de définitions retenues selon le dictionnaire et la méthode de plongement (régulier ou RAS).

Pour chaque dictionnaire anglais, on retrouve une paire de barres dans la figure. Dans ces barres, les parts bleues correspondent à la proportion des définitions conservées dans le plongement en graphes. En comparant la part bleue de chacun des dictionnaires, on peut constater que le plongement en RAS préserve significativement moins de définitions dans la plupart des cas et que dans aucun cas

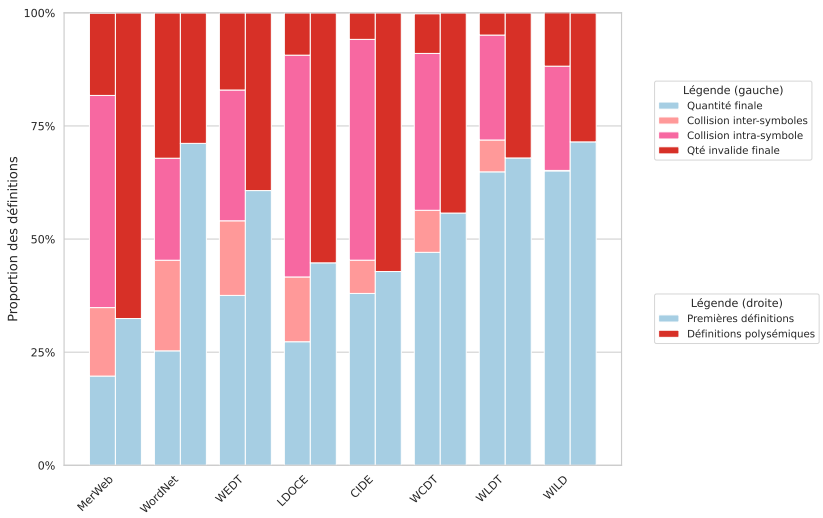


FIGURE 5.1 – Comparaison des définitions retenues en fonction de la méthode de plongement. Une paire de barres est utilisée pour représenter chaque dictionnaire. La barre de gauche représente le plongement RAS alors que la barre de droit le plongement régulier. La première représente la proportion des graphes RAS rejetés parce qu'invalides, ceux rejetés par collisions intra et inter-symboles et finalement les graphes valides conservés. La deuxième représente la proportion des définitions polysémiques (rejetées) et les premières définitions de chaque mot (utilisées).

on ne retrouve plus de définitions préservées par les graphes RAS que les graphes réguliers. Or, qu'est-ce qui explique cette perte d'efficacité ?

La plus grande cause de pertes de définitions est la **collision intra-symbole**, qui survient lorsque le *s-RAS* (soit l'étiquette du mot défini lorsqu'une définition est plongée en RAS, définie dans la sous-section 2.2.3) est partagé par différentes définitions d'un même mot. Deux constats sont à tirer de ces collisions. Premièrement, le modèle StoG n'a pas pu tirer avantage des *sens numérotés* des cadres d'*OntoNotes* pour attribuer des *s-RAS* uniques aux mots polysémiques. On aurait pu effectivement s'attendre à ce que les mots polysémiques dans les dictionnaires le soient aussi dans les cadres d'OntoNotes ; ainsi, une certaine proportion des définitions avait la chance d'être retenue, pourvu qu'on assigne des *s-RAS* qui ne causent pas de collision. Bien qu'il n'était pas réaliste de s'attendre à ce que toutes les définitions possibles des mots dans les 8 dictionnaires anglais soient couvertes par des cadres uniques, l'espoir était qu'une proportion moins grande engendrerait de telles collisions.

Deuxièmement, il y avait probablement une boucle de rétro-action positive pour la génération de ces collisions intra-symbole inhérente à la méthodologie décrite dans la section 2.2. Pour comprendre cette proposée boucle de rétro-action positive, il faut prendre en compte plusieurs points qui suivent. Pour rappel, un critère rendant invalide un graphe RAS est que le *s-RAS* soit représenté par un sous-graphe, c'est-à-dire que le *s-RAS* n'est pas constitué d'un seul nœud. Un autre rappel ici important est la tendance à représenter de façon moins atomique le sens des mots en RAS, c'est-à-dire que le sens contenu dans un mot anglais peut être traduit en plusieurs arcs et nœuds dans un graphe RAS. Cette tendance est d'intérêt pour les analyses psycholinguistiques et l'étude d'ensembles d'ancrage minimaux lorsqu'elle survient dans les définitions mais cause le rejet de graphe RAS lorsqu'elle survient dans le plongements des *s-RAS*. Lorsqu'un graphe RAS était rejeté puisqu'invalide, un nouveau graphe RAS était produit avec la prochaine structure de phrases. Puis, les données démontrent que la principal cause d'invalidité d'un graphe RAS est la non-atomicité du *s-RAS*. Tous ces éléments ont donc renforcé la tendance à produire des étiquettes *s-RAS* identiques pour les mots polysémiques : à chaque fois que le modèle StoG tentait de représenter de façon nuancée le *s-RAS*, sa tentative était rejetée jusqu'à la production d'un *s-RAS* atomique, qui dans la majorité des cas causait une collision intra-symbole.

5.1.2 Comparaison des structures

Un dernier point de discussion est la comparaison entre les structures des dictionnaires anglais et leurs équivalents RAS. Bien qu'en terme absolu les structures soient similaires (des proportions similaires encore une fois pour le cœur et le noyau), les cœurs des dictionnaires RAS sont généralement plus denses que leurs équivalents anglais. Alors qu'une grande partie de l'effet est dû aux différences dans le pré-traitement et la génération de ces deux types de graphes différents, les résultats semblent indiquer que la RAS résulte en des représentations plus compactes du sens. Ainsi, il est très important de garder en tête que ces deux structures ne sont pas nécessairement composées des mêmes définitions, ce qui limite les comparaisons pouvant être faites. Néanmoins, il semble que le plongement RAS résulte en des plus petits ensembles de mots qui sont utilisés plus souvent, probablement explicable par la réification de certains concepts en des arcs.

5.1.3 Applicabilité du plongement RAS

La principale question qui suit est si la structure de graphe comme RAS mérite d'être explorée davantage lors de travaux futurs et quelles seraient les implications d'ensembles d'ancrages minimaux de dictionnaires dont le contenu a été plongé dans des formalismes sémantiques ? Les résultats suggèrent malheureusement que le plongement en RAS soit une version encore plus approximative du contenu d'un dictionnaire que pour les plongements en graphes réguliers et qu'en ce sens, davantage de travaux seraient nécessaires pour tirer avantage des ensembles d'ancrage minimaux de dictionnaires plongés en RAS. Par contre, une étude qualitative des mots *inclus dans les solutions* des plongements en RAS relève la présence de mots fonctionnels qui ne sont pas pris en considération dans les graphes réguliers. Ces mots fonctionnels sont d'intérêts sémantiques et leur présence vont de pair avec l'intérêt croissant qu'ils suscitent dans la littérature récente, étant donné le rôle qu'ils jouent dans la capacité des modèles génératifs (Portelance *et al.*, 2024). Ces capacités génératives supportent l'idée que les mots fonctionnels jouent un rôle plus important que précédemment cru dans la génération des phrases. La conclusion de cette section n'est certes pas que les formalismes sémantiques ne soient pas d'intérêt pour l'étude des ensembles d'ancrage mais que dans l'état actuel, pour la RAS, il faut soit attendre des meilleures performances des modèles StoG ou se tourner vers d'autres formalismes sémantiques.

5.2 Efficacité des réductions

Avant d'entrer plus en détail sur l'efficacité des réductions, il est à noter que ce mémoire ne contient pas l'identification des transversaux de circuits de cardinalité minimale et qu'en ce sens, l'efficacité des réductions ne peut être étudiée en fonctions de leurs effets sur les instances d'identifications de TCCM. Cette étude est donc réservée pour des travaux futurs. Néanmoins, les différents effets et résultats de ces réductions méritent une discussion.

5.2.1 Quantité d'applications

La première métrique qui peut être utilisée pour évaluer l'efficacité et l'importance des réductions est le *nombre de fois que les réductions sont respectivement appliquées*. Cette approche donne OUT-CLIQUE comme la réduction la plus efficace, tel qu'indiqué dans les tableaux 4.7, 4.8, 4.9, 4.10, 4.11 et 4.12, où la réduction représente à elle seule plus de 50% des réductions. Il n'y a pas d'autres

tendances claires communes à tous les dictionnaires (anglais, français ou RAS). Il est intéressant de noter qu'INCLIQUE est appliquée significativement moins souvent qu'OUTCLIQUE alors que l'algorithnme applique INCLIQUE avant OUTCLIQUE ; pour mieux comprendre ces résultats, il pourrait être intéressant de jouer avec l'ordre des réductions et voir si un changement s'opère.

Généralement, la deuxième réduction la plus appliquée est soit LOOP ou INCLIQUE mais de façon intéressante pour les dictionnaires français, c'est DOMEPP qui est la plus souvent appliquée après OUTCLIQUE. Rappelons ce que vérifie DOMEPP. Il s'agit d'une réduction de graphe qui détecte des arcs *dominés par d'autres* ; un arc est dominé s'il fait parti d'un cycle qui contient un cycle plus minimal. Ceci implique donc qu'une proportion significativement plus grande de définitions dans les dictionnaires français forme des cycles et que ces cycles en contiennent de plus minimaux. La proportion est particulièrement haute pour le TLFI, pour lequel la réduction DOMEPP s'est appliquée plus de 5000 fois ! La cause de ce résultat est explicable certes en partie par la quantité de nœuds et d'arcs beaucoup plus grande dans les cœurs des dictionnaires français mais une étude comparative plus approfondie pourrait révéler des propriétés ou caractéristiques propres au français. Tentons quand même une explication : la quantité des cycles révèlent un travail plus rigoureux des lexicographes probablement. Plus de concepts sont correctement reliés entre eux, qui reflètent la circularité de certains ensembles d'apprentissages de définitions.

SUBSET est la réduction la moins efficace : elle consomme plus de temps qu'INCLIQUE et ne s'applique que très rarement, voire aucune fois dans certains dictionnaires. Néanmoins, le coût est négligeable en terme absolu et elle s'applique des fois ; il n'y a donc pas de raisons de l'exclure pour des fins d'identifications de TCCM.

5.2.2 Coût des réductions

Une autre considération pour évaluer l'efficacité d'une réduction est son coût computationnel, ici mesuré en termes de temps d'exécution. Les résultats sont clairs : en dehors de DOMEPP et PIE, les coûts computationnels pour réduire les graphes sont négligeables. Les temps d'exécution pour l'algorithme (excluant DOMEPP) ne sont que de quelques secondes par dictionnaire, et dans chaque cas, c'est PIE qui prend couvre la majorité du temps d'exécution, comme indiqué dans les tableaux 4.7, 4.8, 4.9, 4.10, 4.12 et 4.13. Ces résultats vont dans le sens des prédictions de la sous-section

3.2.3, qui proposait entre autres que la complexité de la vérification des conditions pour appliquer INCLIQUE, OUTCLIQUE et SUBSET étant bornée par la densité des graphes considérés, PIE serait la réduction la plus coûteuse de l'ensemble de réductions confluent.

L'autre réduction dont le temps d'exécution mérite discussion est DOMEPP (tableaux 4.14, 4.11 et 4.15). Dans les dictionnaires français, l'exécution a pris 36h pour le TLFI et 8h pour le Wiki, alors dans que les plus gros dictionnaires anglais et RAS(Wordnet et le Merriam-Webster), la réduction s'appliquait en une trentaine de minutes environ. Cette différence doit être due à la plus grande densité des cœurs des dictionnaires français, qui ne contiennent pas seulement plus de nœuds mais aussi beaucoup plus d'arcs (de relations définitionnelles). Par exemple, le cœur du TLFI contient environ 3 fois plus de nœuds que celui du Merriam-Webster (21952 contre 8708) alors qu'il contient plus de 10 fois le nombre d'arcs (590424 contre 49378).

5.2.3 Effet de la non-confluence

Le dernier élément qui mérite discussion pour ce qui a trait à l'efficacité des réductions est l'effet de la non-confluence pour l'algorithme. Rappelons l'avantage de la confluence et son rôle dans le contexte de ce mémoire. Un ensemble de réductions confluent va, peu importe l'ordre ou la quantité d'applications, résulter en un graphe identique (à isomorphismes près). Cette propriété est intéressante pour obtenir des résultats uniques et réplicables, surtout pour les analyses de propriétés psycholinguistiques (pour lesquelles il serait préférable de conserver la même structure de graphes au travers des différentes analyses) mais pour les TCCM toujours impossibles à identifier, on peut prendre le choix d'abandonner la confluence pour tenter de réduire davantage le dictionnaire.

Un dernier rappel est que les réductions confluentes sont appliquées autant de fois que possible avant d'introduire DOMEPP ; une fois DOMEPP appliquée autant de fois que possible, on réessaye d'appliquer les réductions de l'ensemble confluent. Comme indiqué au début de cette section, l'efficacité des réductions en relation avec l'identification des TCCM est laissée à des travaux futurs, il demeure néanmoins qu'on peut évaluer l'efficacité de DOMEPP en fonction du nombre supplémentaire d'autres réductions dont elle permet l'application. Pour faciliter cette comparaison, les tableaux 5.1, 5.2 et 5.3 ont été produits.

On remarque que les graphes réduits avec des ensembles non-confluents résultent en des graphes

	WordNet	MerWeb	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
Qté restants	-45	-34	-45	-19	-7	-26	-5	-9
Qté inclus	6	3	7	4	1	4	0	0
Qté exclus	39	31	38	15	6	22	5	9
Qté LOOP	5	3	7	4	1	3	0	0
Qté INCLIQUE	22	17	18	9	2	9	3	5
Qté OUTCLIQUE	17	16	23	5	4	14	1	3
Qté SUBSET	1	0	0	0	0	1	0	0
Qté PIE	0	0	9	0	0	0	0	0
Qté arcs restants	-966	-464	-622	-228	-159	-330	-96	-308

TABLE 5.1 – Effet de DOMEPP sur les dictionnaires anglais, lorsque comparé à l'exécution de l'algorithme utilisant l'ensemble confluent. Les valeurs négatives correspondent à des nœuds et des arcs qui ont été retirés, les valeurs positives aux réductions qui ont pu être ré-appliquées.

	WordNet	MerWeb	WEDT	LDOCE	CIDE	WCDT	WLDT	WILD
Qté restants	-20	-39	-14	-16	-9	-21	-10	-32
Qté inclus	0	1	1	1	1	3	1	9
Qté exclus	20	38	13	15	8	18	9	23
Qté LOOP	0	1	1	1	0	3	1	9
Qté INCLIQUE	9	16	9	8	4	15	5	18
Qté OUTCLIQUE	9	22	4	7	4	3	4	4
Qté SUBSET	0	0	0	0	1	0	0	0
Qté PIE	0	0	0	0	0	0	0	4
Qté arcs restants	-511	-694	-298	-202	-280	-205	-73	-404

TABLE 5.2 – Effet de DOMEPP sur les dictionnaires RAS.

contenant généralement plusieurs centaines d'arcs en moins. Les effets ne sont pas trop considérables quant aux nœuds inclus ou exclus (surtout pour les RAS). Il est cependant notable que l'application de DOMEPP crée de nouvelles opportunités, dans tous les dictionnaires, pour INCLIQUE et OUTCLIQUE. Ceci implique que le retrait d'arcs dominés crée de nouvelles cliques : il serait intéressant

	WIKI	TLFI
Qté restants	-74	-34
Qté inclus	6	5
Qté exclus	68	29
Qté LOOP	6	5
Qté INCLIQUE	46	12
Qté OUTCLIQUE	22	16
Qté SUBSET	0	0
Qté PIE	0	0
Qté arcs restants	-2571	-7791

TABLE 5.3 – Effet de DOMEPP sur les dictionnaires français.

d'évaluer quelles sont ces cliques qui apparaissent lors de travaux futurs et quelles sont leurs relations avec ces arcs. Les cœurs des dictionnaires anglais étant moins denses que ceux des dictionnaires RAS, il était peut-être moins probable dans ces derniers que de nouveaux cas d'applications des réductions confluentes apparaissent.

Un résultat particulièrement intéressant est la ré-application de PIE sur le dictionnaire WEDT. Aucun autre dictionnaire, qu'il soit RAS ou français, n'a vu son cœur être fractionné en davantage de composantes fortement connexes. Ce résultat suggère que tout est à gagner en agrandissant autant que possible l'ensemble des réductions (surtout pour ce qui attrait aux TCCM dont l'identification échapperait toujours) : il est possible de fracturer en plus petites CFC même des cœurs aussi gros que celui du WEDT. Une étude plus approfondie du WEDT devrait révéler des propriétés intéressantes : quels sont les mots contenus dans ces composantes fortement connexes ? En quoi la structure du WEDT est unique de sorte à obtenir ce résultat ?

Pour conclure, considérant que certains cœurs réduits sont si gros que l'identification de leur TCCM sera probablement encore inaccessible, il serait intéressant d'améliorer l'algorithme de réductions en lui donnant accès au plus grand ensemble de réductions non-subsumables par d'autres.

5.3 Dictionnaires français et dictionnaires anglais

Le dernier élément de la discussion concerne les structures des graphes issus de dictionnaires français, décrites pour la première fois dans la littérature, et leur comparaison avec les structures des dictionnaires anglais. Une limitation de cette discussion est la quantité de dictionnaires français accessibles : présentement, seuls deux dictionnaires digitaux français ont été analysés, contrairement à huit dictionnaires anglais.

5.3.1 Différences dans les structures

Le constat le plus frappant dans la comparaison des structures est la taille des cœurs et noyaux, tel que mentionné plusieurs fois jusqu'à présent. Pourtant, la quantité de nœuds initiale des dictionnaires français n'est pas particulièrement grande : le TLFI contient à peine 50 000 définitions mais son cœur représente à lui seul 44% de ce nombre.

En comparant seulement avec les 5 plus gros dictionnaires anglais, on peut affirmer que les dictionnaires français pour usagers avancés contiennent une plus faible proportion de mots non-utilisés pour en définir d'autres et une proportion encore plus faible de mots non-définis, particulièrement pour le TLFI. Encore une fois, ces métriques témoignent du rigoureux et lent processus derrière la construction du TLFI par des experts, contrairement à la participation libre-accès du Wiktionnaire. En dehors de Wordnet, les dictionnaires français sont composés d'une proportion moins haute de satellites. Si on s'intéresse à la comparaison avec les trois derniers dictionnaires anglais, pour usagers débutants, on remarque qu'ils ont proportionnellement les plus gros cœurs parmi les dictionnaires anglais. Cependant, cette particularité n'est probablement pas explicable par des propriétés partagées avec les dictionnaires français : s'agissant de dictionnaires pour apprenants, beaucoup des mots moins fréquents sont ignorés (mots qui se retrouvent généralement filtrés au cours du processus de kernelisation, n'étant pas utilisés pour définir d'autres mots). Il serait donc intéressant de comparer lors de travaux futurs les structures de dictionnaires français pour apprenants avec ceux des dictionnaires anglais pour apprenants.

5.3.2 Mots les plus utilisés

Il est intéressant de remarquer que certains mots les plus utilisés en français le sont aussi en anglais, comme *personne/person, faire/do/make, petit/small, action*. Ceci suggère que certaines notions sont fondamentales au travers des différentes langues et qu'il serait possible de les analyser. Ce constat ouvre la porte à des analyses plus poussées lors de futurs travaux : à quel point y a-t-il une équivalence entre les mots des structures entre les différentes langues ? Est-ce que ces similarités sont partagées avec des langues qui utilisent des alphabets non latins ? Ces questions sont abordés dans la conclusion.

CONCLUSION

La conclusion du mémoire récapitule les principaux résultats du mémoire, avant d'ouvrir sur les travaux futurs.

Le point de départ de ce mémoire était la question suivante : *comment les formalismes sémantiques, comme la RAS, peuvent être utilisés pour l'identification d'ensembles d'ancrage minimaux* ? Pour évaluer cette question, le contenu de dictionnaires digitaux de langue anglaise a été plongé en RAS, un formalisme sémantique qui cherche à expliciter le *sens* d'une phrase, de telle sorte que différentes phrases *ayant le même sens* seront représentés par le même RAS. En parallèle à cet effort de recherche, la méthodologie de Vincent-Lamarre et al. a été reproduite à des fins de validation des résultats mais aussi à des fins de comparaison entre les dictionnaires anglais et leurs plongements RAS, et entre dictionnaire anglais et français, ces derniers ayant été décrits pour la première fois dans la littérature dans ce mémoire. Pour identifier les ensembles d'ancrage minimaux de ces différents dictionnaires, il est utile de réduire la tailles des graphes qui en sont produits, c'est pourquoi une autre composante importante de ce mémoire est l'implémentation d'un algorithme de réductions. L'algorithme a été utilisé avec un ensemble de réductions confluent (LOOP, INCLIQUE, OUTCLIQUE, SUBSET et PIE) et un ensemble de réductions non-confluents (les réductions précédentes auxquelles on ajoute DOMEPP).

Malheureusement, l'analyse des résultats a démontré que plonger les définitions d'un dictionnaire en RAS résulte en une proportion moins grande des définitions retenues dans le graphe produit qu'en utilisant la méthodologie de Vincent et al. (Vincent-Lamarre *et al.*, 2017). Cela signifie que retenir la première définition de chaque mot est une meilleure approximation que le plongement en RAS. Cependant, les résultats ont aussi confirmé l'intérêt à des fins de modélisation cognitive de cet effort de plongement, et des travaux futurs pour raffiner la méthodologie sont envisagés.

L'exécution de l'algorithme et l'efficacité des différentes réductions a été évaluée en fonction du dictionnaire. L'analyse de ces résultats a révélé d'importantes différences entre les dictionnaires anglais et français, surtout quant à la quantité de relations définitionnelles. Les dictionnaires français contiendraient aussi moins de mots inutilisés pour en définir d'autres et moins de mots non-définis.

Plusieurs travaux pourraient découler du présent projet. En voici quelques-uns.

Générer des dictionnaires désambiguïsés : l'échec relatif du plongement RAS pour couvrir une plus grande proportion de définitions a rendu claire pour moi une avenue plus prometteuse : utiliser des grands modèles de langue pour générer des dictionnaires désambiguïsés. Effectivement, il est possible de numéroter un mot polysémique s de s_1 à s_k (où k représente le nombre de définitions du mot s), et d'associer sa définition à chaque s_n (où n représente une valeur entre 1 et k inclusivement) de la liste précédente. Ensuite, pour chaque mot polysémique utilisé dans une définition, un *prompt* est créé contenant la liste des définitions du mot polysémique et la définition où il est utilisé : le grand modèle de langue doit alors indiquer quel s_n est ici le bon. Ce processus résulterait en des dictionnaires désambiguïsés, sur lesquels nous travaillons déjà. Un tel dictionnaire désambiguïsé pourrait ensuite être plongé en RAS dans un processus hybride, ce qui pallierait grandement la perte de définitions.

Dictionnaires d'autres langues : en parallèle à la complétion de ce mémoire, je collabore à la création de graphes à partir de dictionnaires chinois et de dictionnaires en langues des signes française et anglaise. Cependant, il serait intéressant d'étendre à un nombre encore plus grands de dictionnaires et de langues les jeux de données. Ces nouveaux jeux de données riches pourront être explorés à partir des méthodologies développées dans ce mémoire, afin d'étudier les propriétés communes aux différents ensembles d'ancrage minimaux.

Autres formalismes sémantiques : la RAS étant strictement restreinte à l'anglais, il serait intéressant de trouver des solutions pour adapter la méthodologie à d'autre langues. Bien qu'il existe de telles tentatives d'adaptation de la RAS, leur succès demeure mitigé (Martínez Lorenzo *et al.*, 2022). Il existe de nouveaux formalismes sémantiques similaires à la RAS pensés pour être des *interlinguas* (pouvant être utilisés avec plusieurs langues à la fois). Le plus prometteur pour le moment nous semble être le *BabelNet-Meaning Representation* (BMR) (Martínez Lorenzo *et al.*, 2022). Ce formalisme sémantique est d'un très grand intérêt pour ce projet pour les raisons suivantes. Le BMR est un réseau sémantique multi-lingue et multi-modal. C'est-à-dire que les cadres de chaque mot qui s'y trouvent sont liés avec les cadres des mots équivalents dans d'autres langues, le tout accompagné d'images et de sons. Identifier l'ensemble minimal d'ancrage de ce réseau sémantique aurait des implications intéressantes pour le développement d'agent : on pourrait développer un

modèle qui, étant donné un ensemble d'ancrage minimal, apprend à performer de nouvelles tâches par définitions et instructions verbales, et ce dans plusieurs langues.

Utilisation d'un plus grand nombre de réductions non-confluentes : Comme indiqué dans le chapitre 5, il serait particulièrement pertinent d'étendre autant que possible l'ensemble des réductions. Les tailles des cœurs français sont telles qu'ils n'est pas réaliste de s'attendre à identifier leur TCCM en un temps raisonnable. Pour s'aider, il faudrait donc implémenter le plus grand nombre possible de réductions non-subsumables décrites dans le tableau 3.1.

Solveur de TCCM par *MaxSAT* : Une autre avenue intéressante serait de comparer l'efficacité d'un solveur MaxSAT par opposition à la programmation linéaire en nombre entiers. Au début du projet de maîtrise, il était envisagé d'implémenter un nouveau solveur de TCCM, librement inspiré de celui ayant remporté PACE2022 (Kiesel et Schidler, 2023). Cependant, il est vite devenu apparent que deux ans auraient été insuffisants pour en plus implémenter un tout nouveau solveur de TCCM ; l'exercice est donc laissé à des travaux futurs.

RÉFÉRENCES

Abdenbi, M., Blondin Massé, A., Goupil, A. et Marcotte, O. (2024). On the confluence of directed graph reductions preserving feedback vertex set minimality. *arXiv preprint arXiv :2406.16390*.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M. et Schneider, N. (2013). Abstract meaning representation for sembanking. Dans *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*, 178–186.

Blondin Massé, A., Chicoisne, G., Gargouri, Y., Harnad, S., Picard, O. et Marcotte, O. (2008). How is meaning grounded in dictionary definitions ? *arXiv preprint arXiv :0806.3710*.

Blondin Massé, A., Harnad, S., Picard, O. et St-Louis, B. (2010). Symbol grounding and the origin of language. *Lefebvre ; Comrie ; Cohen, New Perspectives on the Origins of Language*, 279–97.

Bonial, C., Foresta, J., Fung, N. C., Hayes, C., Osteen, P., Arkin, J., Hedegaard, B. et Howard, T. (2023). Abstract meaning representation for grounded human-robot communication. Dans *Proceedings of the Fourth International Workshop on Designing Meaning Representations*, 34–44.

Cabezudo, M. A. S. et Pardo, T. (2019). Towards a general abstract meaning representation corpus for brazilian portuguese. Dans *Proceedings of the 13th Linguistic Annotation Workshop*, 236–244.

Chalmers, D. J. (1995). Facing up to the problem of consciousness. *Journal of consciousness studies*, *2*(3), 200–219.

Davidson, D. (1969). The individuation of events. In *Essays in honor of Carl G. Hempel : A tribute on the occasion of his sixty-fifth birthday* 216–234. Springer.

Diestel, R. (2024). *Graph theory*. Springer (print edition) ; Reinhard Diestel (eBooks).

Dohare, S., Karnick, H. et Gupta, V. (2017). Text summarization using abstract meaning representation. *arXiv preprint arXiv :1706.01678*.

Fellbaum, C. (1998). Wordnet : An electronic lexical database.

Fellows, M. R., Jaffke, L., Király, A. I., Rosamond, F. A. et Weller, M. (2018). *What Is Known About Vertex Cover Kernelization ?*, Dans H.-J. Böckenhauer, D. Komm, et W. Unger (dir.). *Adventures Between Lower Bounds and Higher Altitudes : Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday*, (p. 330–356). Springer International Publishing : Cham

Fomin, F. V., Gaspers, S., Pyatkin, A. V. et Razgon, I. (2008). On the minimum feedback vertex set problem : Exact and enumeration algorithms. *Algorithmica*, *52*, 293–307.

Goodman, M. W. (2020). Penman : An open-source library and tool for amr graphs. Dans *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics : System Demonstrations*, 312–319.

Harnad, S. (1990). The symbol grounding problem. *Physica D : Nonlinear Phenomena*, *42*(1-3), 335–346.

Harnad, S. (2001). Mind, machines and searle ii : What's wrong and right about searle's chinese room argument ?

Harnad, S. (2008). The annotation game : On turing (1950) on computing, machinery, and intelligence (published version bowdlerized).

Harnad, S. (2012). Alan turing and the" hard" and" easy" problem of cognition : Doing and feeling. *arXiv preprint arXiv :1206.3658*.

Harnad, S. (2017). To cognize is to categorize : cognition is categorization. In *Handbook of categorization in cognitive science* 21–54. Elsevier.

Harnad, S. (2024). Language writ large : Llms, chatgpt, grounding, meaning and understanding. *arXiv preprint arXiv :2402.02243*.

Honnibal, M., Montani, I., Van Landeghem, S. et Boyd, A. (2020). spaCy : Industrial-strength Natural Language Processing in Python. `http://dx.doi.org/10.5281/zenodo.1212303`

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L. et Weischedel, R. (2006). Ontonotes : the 90% solution. Dans *Proceedings of the human language technology conference of the NAACL, Companion Volume : Short Papers*, 57–60.

Jascob, B. (2023). Amrlib models [software]. `https://github.com/bjascob/amrlib-models`. GitHub repository.

Karp, R. M. (2010). *Reducibility among combinatorial problems*. Springer.

Kiesel, R. et Schidler, A. (2023). A dynamic maxsat-based approach to directed feedback vertex sets. Dans *2023 Proceedings of the Symposium on Algorithm Engineering and Experiments (ALENEX)*, 39–52. SIAM.

Kouris, P., Alexandridis, G. et Stafylopatis, A. (2024). Text summarization based on semantic graphs : An abstract meaning representation graph-to-text deep learning approach. *Journal of Big Data*, *11*(1), 95.

Lemaic, M. (2008). *Markov-chain-based heuristics for the feedback vertex set problem for digraphs*. (Thèse de doctorat). Universität zu Köln.

Levy, H. et Low, D. W. (1988). A contraction algorithm for finding small cycle cutsets. *Journal of algorithms*, *9*(4), 470–493.

Li, C. et Flanigan, J. (2022). Improving neural machine translation with the abstract meaning representation by combining graph and sequence transformers. Dans *Proceedings of the 2nd Workshop on Deep Learning on Graphs for Natural Language Processing*, 12–21. Association for Computational Linguistics. `http://dx.doi.org/10.18653/v1/2022.dlg4nlp-1.2`

Lin, H.-M. et Jou, J.-Y. (2000). On computing the minimum feedback vertex set of a directed graph by contraction operations. *IEEE Transactions on computer-aided design of integrated circuits and systems*, *19*(3), 295–307.

Lorenz, K. Z. (1937). The companion in the bird's world. *The Auk*, *54*(3), 245–273.

Martínez Lorenzo, A. C., Maru, M. et Navigli, R. (2022). Fully-Semantic Parsing and Generation : the BabelNet Meaning Representation. Dans S. Muresan, P. Nakov, et A. Villavicencio (dir.). *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, 1727–1741., Dublin, Ireland. Association for Computational Linguistics. `http://dx.doi.org/10.18653/v1/2022.acl-long.121`. Récupéré de `https://aclanthology.org/2022.acl-long.121`

Maru, M., Conia, S., Bevilacqua, M. et Navigli, R. (2022). Nibbling at the hard core of word sense disambiguation. Dans *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, 4724–4737.

Matthiessen, C. M. et Bateman, J. A. (1991). Text generation and systemic-functional linguistics : experiences from english and japanese. *(No Title)*.

Merriam-Webster (2003). *Merriam-Webster's Collegiate Dictionary*.

Noble, S., Curtiss, J., Pessoa, L. et Scheinost, D. (2024). The tip of the iceberg : a call to embrace anti-localizationism in human neuroscience research. *Imaging Neuroscience*, *2*, 1–10.

Oral, E., Acar, A. et Eryiğit, G. (2024). Abstract meaning representation of turkish. *Natural Language Engineering*, *30*(1), 171–200.

Palmer, M., Gildea, D. et Kingsbury, P. (2005). The proposition bank : An annotated corpus of semantic roles. *Computational Linguistics*, *31*(1), 71–106. `http://dx.doi.org/10.1162/0891201053630264`

Portelance, E., Frank, M. C. et Jurafsky, D. (2024). Learning the meanings of function words from grounded language using a visual question answering model. *Cognitive Science*, *48*(5), e13448.

Procter, P. (1978). *Longman Dictionary of Contemporary English (LDOCE)*.

Procter, P. (1995). *Cambridge International Dictionary of English (CIDE)*.

Rosen, B. K. (1976). Correctness of parallel programs : The church-rosser approach. *Theoretical Computer Science*, *2*(2), 183–207.

Rosen, K. H. et Krithivasan, K. (1999). *Discrete mathematics and its applications*, volume 6. McGraw-Hill New York.

Saussure, F. d., Bally, C., Sechehaye, A., Riedlinger, A. et Harris, R. (2020). Course in general linguistics.

Stege, U. et Fellows, M. R. (1999). An improved fixed parameter tractable algorithm for vertex cover. *Technical report/Departement Informatik, ETH Zürich*, *318*.

Takhshid, R., Shojaei, R., Azin, Z. et Bahrani, M. (2022). Persian abstract meaning representation. *arXiv preprint arXiv :2205.07712*.

Tarjan, R. (1972). Depth-first search and linear graph algorithms. *SIAM journal on computing*, *1*(2), 146–160.

Tohidi, N. et Dadkhah, C. (2022). A short review of abstract meaning representation applications. *Modeling and Simulation in Electrical and Electronics Engineering*, *2*(3), 1–9.

Turing, A. M. (2021). Computing machinery and intelligence (1950).

Valiant, L. G. (1979). The complexity of computing the permanent. *Theoretical computer science*, *8*(2), 189–201.

Van Gysel, J. E., Vigus, M., Chun, J., Lai, K., Moeller, S., Yao, J., O'Gorman, T., Cowell, A., Croft, W., Huang, C.-R. *et al.* (2021). Designing a uniform meaning representation for natural language processing. *KI-Künstliche Intelligenz*, *35*(3), 343–360.

Vaswani, A. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*.

Vincent-Lamarre, P., Blondin Massé, A., Lopes, M., Lord, M., Marcotte, O. et Harnad, S. (2016). The latent structure of dictionaries. *Topics in cognitive science*, *8*(3), 625–659.

Vincent-Lamarre, P., Lord, M., Blondin Massé, A., Marcotte, O., Lopes, M. et Harnad, S. (2017). Hidden structure and function in the lexicon. In *Cognitive approach to natural language processing* 91–108. Elsevier.

Wein, S., Donatelli, L., Ricker, E., Engstrom, C., Nelson, A. et Schneider, N. (2022). Spanish abstract meaning representation : Annotation of a general corpus. *arXiv preprint arXiv :2204.07663*.

Wordsmyth (2017). Wordsmyth. Récupéré de `https://www.wordsmyth.net`

Xiao, M. et Nagamochi, H. (2013). Confining sets and avoiding bottleneck cases : A simple maximum independent set algorithm in degree-3 graphs. *Theoretical Computer Science*, *469*, 92–104.